Problem 1 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int n = 5, numbers[4] = {10, 8, 7, 7};
   string states[3] = {"NY", "NJ", "CT"};
   double x;

   // set x to the average  8.0
   x = average(n, n, 14);                // (a)
   // print the following to screen:  x equals 8.0
   print("x equals", x);                 // (b)
   // print to screen:  The best is NY
   bestState(states, 3);                 // (c)
   if (mystery(numbers[1], states[1]))   // (d)
      x = e(e(x));                       // (e)
   return 0;
}
```

(a) Title line for **average** as called at the line marked (a).

Answer:

(b) Title line for **print** as called at the line marked (b).

Answer:

(c) Title line for **bestState** as called at the line marked (c).

Answer:

(d) Title line for **mystery** as called at the line marked (d).

Answer:

(e) Title line for **e** as called at the line marked (e).

Answer:

Problem 2 Consider the following `C++` program.

```
int f(int x, int &y) {
   x = x + 1;
   y = x - 1;
   return x + y;
}

int main() {
    int x[4] = {4, 3, 2, 1};
    int y[6] = {2, 3, 5, 7, 11, 13};
    cout << y[3] % 7 << endl;                           // line (a)
    cout << y[x[1]] << endl;                            // line (b)
    cout << f(x[0], y[0]) << endl;                      // line (c)
    cout << y[0] << x[0] << endl;                       // line (d)
    cout << f(f(x[1], y[1]), x[2]) << endl;             // line (e)
}
```

(a) What is the ouput at line (a)?
Answer:

(b) What is the ouput at line (b)?
Answer:

(c) What is the ouput at line (c)?
Answer:

(d) What is the ouput at line (d)?
Answer:

(e) What is the ouput at line (e)?
Answer:

Problem 3 Write a function called *allOdd* that returns a result of `"All Odd"` if all the elements of a 2-dimensional array are odd and returns `"Not All Odd"` otherwise. The array should have 2 columns and contain integers.

Excessively long solutions that use more than 8 lines of code may lose points. A program that uses the function *allOdd* follows.

```
int main() {
   int x[2][2] = {{1, 3}, {5, 7}};
   cout << allOdd(x, 2, 2) << endl;    // prints   All Odd
   return 0;
}
```

Answer:

Problem 4 The recursive function `rotateDigits` moves the first digit to the end of a positive integer. All other digits move one place to the left. For example `rotateDigits(12345)` returns 23451.

An implementation of this function with parts of the code covered up is given below. There is also a main program that uses it.

Some pieces of code have been replaced by `PART (a)`, `PART (b)`, and so on. To answer the 5 parts of this question you should supply the `C++` code that was replaced. Each answer must fit on a single line.

```
int rotateDigits(int x) {
   if (x < 10) PART (a);
   int y = PART (b);
   x = PART (c);
   int z = PART (d);
   return PART (e);
}

int main() {
   cout << rotateDigits(19683) << endl;  // prints 96831
   cout << rotateDigits(1024) << endl;   // prints 241 because the 0 disappears
   cout << rotateDigits(19) << endl;     // prints 91
   cout << rotateDigits(9) << endl;      // prints 9
   return 0;
}
```

(a) Give a replacement for PART (a) as the base case of recursion:

Answer:

(b) Give a replacement for PART (b) to make y store the original last digit:

Answer:

(c) Give a replacement for PART (c) to drop the last digit and make a recursive call:

Answer:

(d) Give a replacement for PART (d) to make z store the new last digit:

Answer:

(e) Give a replacement for PART (e) to use the values of x, y and z to make an answer:

Answer:

Problem 5 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int n = 5, numbers[4] = {9, 8, 7, 7}, x;
   double reals[3] = {1.9, 2.3, 3.0};
   char ch = 'b';

   // set x to the sum n + 5 + 5
   x = sum(n, 5, 5);                         // (a)
   // print the following to screen:  x equals 15
   print("x equals", x);                    // (b)
   // print to screen:  The mode is 7
   findMode(numbers, 4);                    // (c)
   if (mystery(numbers, numbers[1], reals[2]))   // (d)
      ch = e(mystery(numbers, numbers[2], reals[0]), e(false, ch)); // (e)
   return 0;
}
```

(a) Title line for **sum** as called at the line marked (a).

Answer:

(b) Title line for **print** as called at the line marked (b).

Answer:

(c) Title line for **findMode** as called at the line marked (c).

Answer:

(d) Title line for **mystery** as called at the line marked (d).

Answer:

(e) Title line for **e** as called at the line marked (e).

Answer:

Problem 6 Consider the following `C++` program.

```
int f(int &x, int y) {
   x = x + 1;
   y = y - 1;
   return x + y;
}

int main() {
    int x[4] = {1, 2, 3, 4};
    int y[6] = {2, 3, 5, 7, 11, 13};
    cout << y[3] % 9 << endl;                          // line (a)
    cout << y[x[1]] << endl;                           // line (b)
    cout << f(x[0], y[0]) << endl;                     // line (c)
    cout << x[0] << y[0] << endl;                      // line (d)
    cout << f(y[2], f(x[1], y[1])) << endl;            // line (e)
}
```

(a) What is the ouput at line (a)?

Answer:

(b) What is the ouput at line (b)?

Answer:

(c) What is the ouput at line (c)?

Answer:

(d) What is the ouput at line (d)?

Answer:

(e) What is the ouput at line (e)?

Answer:

Problem 7 Write a function called *increases* that returns a result of true if the elements of an array increase as their index increases and returns false otherwise. If the array contains 1,3,5,6 your function should return true, but if the array contains 1,3,3,5 your function should return false because the element with index 2 is not bigger than the one with index 1.

Excessively long solutions that use more than 8 lines of code may lose points. A program that uses the function *increases* follows.

```
int main() {
   int x[8] = {1, 3, 5, 7, 9};
   if (increases(x, 5))
     cout << "It increases\n";                 // prints:  It increases
   else cout << "It does not increase\n";
   return 0;
}
```

Answer:

Problem 8 The recursive function `rotateDigits` moves the last digit to the start of a positive integer. All other digits move one place to the right. For example `rotateDigits(12345)` returns 51234.

An implementation of this function with parts of the code covered up is given below. There is also a main program that uses it.

Some pieces of code have been replaced by `PART (a)`, `PART (b)`, and so on. To answer the 5 parts of this question you should supply the `C++` code that was replaced. Each answer must fit on a single line.

```
int rotateDigits(PART (a)) {
   if (x < 10) PART (b);
   int y = PART (c);
   int z = PART (d);
   x = rotateDigits((x / 100) * 10 + y);
   return PART (e);
}

int main() {
   cout << rotateDigits(19683) << endl;  // prints 31968
   cout << rotateDigits(19680) << endl;  // prints 1968 because the 0 disappears
   cout << rotateDigits(19) << endl;     // prints 91
   cout << rotateDigits(9) << endl;      // prints 9
   return 0;
}
```

(a) Give a replacement for PART (a) to declare the parameter `x`

Answer:

(b) Give a replacement for PART (b) as the base case of recursion:

Answer:

(c) Give a replacement for PART (c) to set y to the last digit in x:

Answer:

(d) Give a replacement for PART (d) to set z to the next to last digit in x

Answer:

(e) Give a replacement for PART (e) to use the values of x, y and z to make an answer:

Answer:

Problem 9 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int num[3] = {0, 1, 2};
   double rl[3] = {1.9, 2.3, 3.0};
   char ch = 'b';

   ch = a(ch, ch);                   // (a)
   rl[0] = b(num[0], ch);            // (b)
   num[2] = c(a(ch,ch));             // (c)
   d(d(ch, num[1]), 5);              // (d)
   a(e(num[0] + rl[1], rl), ch);     // (e)
   return 0;
}
```

(a) Title line for **a** as called at the line marked (a).

Answer:

(b) Title line for **b** as called at the line marked (b).

Answer:

(c) Title line for **c** as called at the line marked (c).

Answer:

(d) Title line for **d** as called at the line marked (d).

Answer:

(e) Title line for **e** as called at the line marked (e).

Answer:

Problem 10 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void a(int x[], int y) {
   x[0] = y;
   y = x[1];
}
void b(int x[], int &y) {
   x[2] = y;
   y = x[3];
}
void c(int x[], int y) {
   if (y < 2) return;
   cout << x[y + 1];
   c(x, y/2);
}

int main() {
    int x[6] = {1, 2, 3, 4, 5, 6};
    int y = 3, z = 4;
    cout << x[y % z] << endl;                              // line (a)
    a(x, y);
    cout << x[0] << y << endl;                             // line (b)
    b(x, z);
    cout << x[2] << z << endl;                             // line (c)
    c(x, 4);  cout << endl;                                // line (d)
    for (int i = 0; i < 4; i++) c(x, i); cout << endl;    // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 11 Write a function called *adjust* adds a random integer between -2 and 2 to each element of an array of integers. The biggest allowed change to an entry of the array is 2 (either up or down) and the smallest allowed change is 0. Excessively long solutions that use more than 10 lines of code may lose points. You must use the standard C++ random number function. You should give any relevant `#include` statements.

A program that uses the function *adjust* follows.

```
int main() {
   int x[5] = {3,1,4,1,5};
   adjust(x, 5);
   cout << x[1] << endl;   // prints a value that could be any of -1, 0, 1, 2, 3
   return 0;
}
```

Answer:

Problem 12 Write a function called `digitMatch`. The function has three integer parameters `first` , `second` and `third` that are positive and have the same number of digits. It returns the number of positions where their digits match. For example, if the function was applied to 17345, 97813 and 17313 it would return 1 because only the 2nd digits match. If parameters have illegal values your function can operate however you choose. Excessively long solutions that use more than 6 lines of code may lose points.

For example, a program that uses the function follows.

```
int main() {
   cout << digitMatch(168, 567, 767) << endl;     // prints 1
   cout << digitMatch(143, 243, 343) << endl;     // prints 2
   return 0;
}
```

Answer:

Problem 13 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int qq;
   double rr[3] = {0, 1.1, 2.2};
   string st[3] = {"1.9", "2.3", "3.0"};

   qq = f1(rr[2] + rr[1], rr[2]);                         // (a)
   st[0] = f2(rr[0] + rr[1], rr[0], rr[0], st[2]);    // (b)
   if (f3(st, st, 3)) cout << 2;                          // (c)
   f4(st[1], qq);                                         // (d)
   char k = f4(f5(rr[1], st), rr[1]);                     // (e)
   return 0;
}
```

(a) Title line for **f1** as called at the line marked (a).

Answer:

(b) Title line for **f2** as called at the line marked (b).

Answer:

(c) Title line for **f3** as called at the line marked (c).

Answer:

(d) Title line for **f4** as called at the line marked (d).

Answer:

(e) Title line for **f5** as called at the line marked (e).

Answer:

Problem 14 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void a(string x[], string y) {
   x[0] = y;
   y = x[1];
}
void b(string x[], string &y) {
   x[2] = y;
   y = x[3];
}
void c(string x[], int y) {
   if (y < 2) return;
   cout << x[y - 1];
   c(x, y/2);
}

int main() {
    string x[6] = {"l", "m", "n", "o", "p", "q"};
    string y = "Queens", z = "College";
    cout << x[11 % 3] << endl;                           // line (a)
    a(x, y);
    cout << x[0] << y << endl;                           // line (b)
    b(x, z);
    cout << x[2] << z << endl;                           // line (c)
    c(x, 5);  cout << endl;                              // line (d)
    for (int i = 0; i < 3; i++) c(x, i); cout << endl;   // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 15 Write a function called *advance* that moves each entry one place forward in an array and moves the last entry back to the beginning. So for example, an array storing 1,2,3,4 will be changed to store 4,1,2,3 because the first 3 entries are moved forward and the last is put back to the beginning. Excessively long solutions that use more than 10 lines of code may lose points.

A program that uses the function *advance* follows.

```
int main() {
   int x[5] = {3,1,4,1,5};
   advance(x, 5);
   cout << x[0] << x[1] << x[2] << x[3] << x[4] << endl;   // prints 53141
   return 0;
}
```

Answer:

Problem 16 Write a function called `digitsOpposite`. The function has two integer parameters `x` and `y` that are positive and have the same number of digits. It returns the number of positions where one number has an even digit and the other has an odd one. For example, if the function was applied to 17345 and 97813 it would return 2 because the third digits are 3 and 8 and the fourth ones are 4 and 1. (In both cases one of these is even and the other is odd.)

If parameters have illegal values your function can operate however you choose. Excessively long solutions that use more than 6 lines of code may lose points. A program that uses the function follows.

```
int main() {
   cout << digitsOpposite(17345, 97813) << endl;     // prints 2
   cout << digitsOpposite(13579, 24680) << endl;     // prints 5
   return 0;
}
```

Answer:

Problem 17 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   double ww[3] = {0, 1, 2};
   int nn[3] = {1, 2, 3};
   string word = "true";

   word = a(word, word);                // (a)
   nn[0] = b(ww[0], nn[1]);             // (b)
   ww[2] = c(a(word, word));            // (c)
   d(d(ww[0], ww[1]), 5);               // (d)
   a(e(ww[0] + ww[1], nn, ww), word); // (e)
   return 0;
}
```

(a) Title line for **a** as called at the line marked (a).
Answer:

(b) Title line for **b** as called at the line marked (b).
Answer:

(c) Title line for **c** as called at the line marked (c).
Answer:

(d) Title line for **d** as called at the line marked (d).
Answer:

(e) Title line for **e** as called at the line marked (e).
Answer:

Problem 18 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void a(int x[], int y) {
   x[0] = y;
   y = x[1];
}
void b(int x[], int &y) {
   x[0] = y;
   y = x[3];
}
void c(int x[], int y) {
   if (y < 2) return;
   cout << x[y -1];
   c(x, y/2);
}

int main() {
    int x[6] = {3, 4, 5, 6, 7, 8};
    int y = 3, z = 4;
    cout << x[y] % x[z] << endl;                          // line (a)
    a(x, y);
    cout << x[0] << y << endl;                            // line (b)
    b(x, x[2]);
    cout << x[2] << x[0] << endl;                         // line (c)
    c(x, 6);  cout << endl;                               // line (d)
    for (int i = 0; i < 5; i++) c(x, i); cout << endl;   // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 19 Write a function called *maxRow* that returns the number of the row in a 2-dimensional array of integers (with 5 columns) that has the maximum number of positive entries. If there is more than one row that gives the maximum, you can return any of these possibilities. Excessively long solutions that use more than 18 lines of code may lose points.

A program that uses the function *maxRow* follows.

```
int main() {
   int x[2][5] = { {-1, -2, 1, -3, 5}, {-5, -6, -4, -7, -8}};
   cout << maxRow(x, 2, 5) << endl;   // prints 0
    // because row 0 has two positive entries and row 1 has none
   return 0;
}
```

Answer:

Problem 20 The recursive function `swapFirst` is used to swap the first digits of two positive integers that have the same number of digits. For example swapping the first digits in 123 and 987 gives the numbers 923 and 187.

The parameters of the function represent two positive integers that have the same number of digits. An implementation of `swapFirst` with parts of its code covered up is given below. There is also a main program that calls the function.

Some pieces of code have been replaced by `PART (a)`, `PART (b)`, and so on. To answer the 5 parts of this question you should supply the `C++` code that was replaced. Each answer must fit on a single line.

```
void swapFirst(PART (a)) {
   if (PART (b)) {
      int temp = b;
      b = a;
      a = temp;
   }
   else {
      int aStart = a / 10, bStart = b / 10;
      PART (c)
      a = PART (d);
      b = PART (e);
   }
}

int main() {
   int x = 243, y = 357;
   swapFirst(x, y);
   cout << x << "  " << y << endl;  // prints 343 257
   return 0;
}
```

(a) Give a replacement for PART (a) to declare parameters `a` and `b`

Answer:

(b) Give a replacement for PART (b) to test for the base case of recursion:

Answer:

(c) Give a replacement for PART (c) as a useful recursive call:

Answer:

(d) Give a replacement for PART (d) with a useful expression:

Answer:

(e) Give a replacement for PART (e) with a useful expression:

Answer:

Problem 21 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int dd[3] = {0, 1, 2};
   string st[3] = {"1.9", "2.3", "3.0"};

   dd[1] = f1(dd[2] + dd[1], dd[2]);                  // (a)
   st[0] = f2(dd[0] + dd[1], dd[0], dd[0], st[2]);    // (b)
   dd[1] = f3(st, st, 3);                             // (c)
   f4(st[1], 1);                                      // (d)
   bool k = f4(f5(dd[1], st), dd[1]);                 // (e)
   return 0;
}
```

(a) Title line for **f1** as called at the line marked (a).

Answer:

(b) Title line for **f2** as called at the line marked (b).

Answer:

(c) Title line for **f3** as called at the line marked (c).

Answer:

(d) Title line for **f4** as called at the line marked (d).

Answer:

(e) Title line for **f5** as called at the line marked (e).

Answer:

Problem 22 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void a(string x[], string y) {
   x[0] = y;
   y = x[1];
}
void b(string x[], string &y) {
   x[0] = y;
   y = x[2];
}
void c(string x[], int y) {
   if (y < 2) return;
   cout << x[y];
   c(x, y/2);
}

int main() {
    string x[6] = {"z", "y", "x", "w", "v", "u"};
    string y = "CSCI", z = "111";
    cout << x[111 % 3] << endl;                         // line (a)
    a(x, x[3]);
    cout << x[0] << x[3] << endl;                       // line (b)
    b(x, z);
    cout << x[0] << z << endl;                          // line (c)
    c(x, 5);  cout << endl;                             // line (d)
    for (int i = 0; i < 2; i++) c(x, i); cout << endl;   // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 23 Write a function called *maxCol* that returns the number of the column in a 2-dimensional array of integers (with 4 columns) that has the largest sum. If more than one column gives the largest sum you can return any of these possibilities. Excessively long solutions that use more than 18 lines of code may lose points.

A program that uses the function *maxCol* follows.

```
int main() {
   int x[2][4] = { {-1, -2, 1, -3}, {-5, -6, -4, -7}};
   cout << maxCol(x, 2, 4) << endl;   // prints 2
    // because the sum of column 2 is -3 = 1 - 4 which is larger than others
   return 0;
}
```

Answer:

Problem 24 The recursive function `bigSmall` applies to two positive integers that have the same number of digits. It changes the first to a number in which the digit at any position is the larger of the digits of the two parameters at that position. Similarly it changes the second to show the smaller digits. For example, if it was applied to parameters with values 31415 and 27182, it would change them to 37485 and 21112.

The function has two integer parameters that represent the numbers being considered.

An implementation of `bigSmall` with parts of its code covered up is given below. There is also a main program that calls the function.

Some pieces of code have been replaced by `PART (a)`, `PART (b)`, and so on. To answer the 5 parts of this question you should supply the `C++` code that was replaced. Each answer must fit on a single line.

```
void bigSmall(PART (a)) {
   if (x == 0) PART (b);
   int xStart = x / 10, yStart = y / 10;
   PART (c)
   int max = x % 10, min = y % 10;
   if (max < min) {
      max = y % 10;
      min = x % 10;
   }
   x = PART (d);
   y = PART (e);
}

int main() {
   int x = 31415, y = 27182;
   bigSmall(x, y);
   cout << x << "  " << y << endl;  // prints 37485 21112
   return 0;
}
```

(a) Give a replacement for PART (a) to declare parameters `x` and `y`

Answer:

(b) Give a replacement for PART (b) as the base case of recursion:

Answer:

(c) Give a replacement for PART (c) as a useful recursive call:

Answer:

(d) Give a replacement for PART (d) with a useful expression:

Answer:

(e) Give a replacement for PART (e) with a useful expression:

Answer:

Problem 25 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string school = "Queens College CUNY";
   int array[2][3] = {{-1, -2, -3}, {3, 4, 5}};
   int data[5] = {3, 1, 4, 1, 5};
   cout << char0(school) << endl;              // (a) prints: Q
   cout << sumFirstCol(array, 2, 3) << endl;  // (b) prints: 2  (as -1 + 3).
   cout << borough(school) << endl;            // (c) prints: Queens
   randomize(array, 2, 3);           // (d) reset the array with random entries
   cout << roundUp(((double) data[0])/((double) data[2])); // (e) prints 1
                                        //  round up the ratio to an int.
   return 0;
}
```

(a) Title line for **char0** as called at the line marked (a).

Answer:

(b) Title line for **sumFirstCol** as called at the line marked (b).

Answer:

(c) Title line for **borough** as called at the line marked (c).

Answer:

(d) Title line for **randomize** as called at the line marked (d).

Answer:

(e) Title line for **roundUp** as called at the line marked (e).

Answer:

Problem 26 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int function(int x, int &y) {
   if (x == y) cout << y;
   else if (x > y) y++;
   else function(y, x);
   return x;
}

int main() {
    int a[6] = {3, 1, 4, 1, 5, 9};
    int b = 5, c = 2;
    cout << a[b] / a[c] << endl;                              // line (a)
    cout << function(a[b], a[b]) << endl;                     // line (b)
    for (int r = 3; r <= 5; r++) cout << function(r, c);      // line (c)
    cout << endl;
    cout << c << endl;                                        // line (d)
    function(a[4], a[5]);  cout << a[4] << a[5] << endl;      // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 27 Write a function called *findIndex* that returns the the first index of an array whose entry matches a given target. If the target is not present return -1. Excessively long solutions that use more than 8 lines of code may lose points.

For example, a program that uses the function *findIndex* follows.

```
int main() {
   int x[8] = { 1, -1, -2, -3, -4, -4, -2, 0};
   cout << findIndex(x, 8, -3) << endl;   // prints 3 because the target -3 is at index 3
   cout << findIndex(x, 8, -2) << endl;   // prints 2 because -3 first appears at index 2
   cout << findIndex(x, 8, 2) << endl;    // prints -1 because 2 is not present
   return 0;
}
```

Answer:

Problem 28 Write a function called `startsWith`. The function has two integer parameters that are both positive. It determines whether the second number starts with the first number. For example, 19683 starts with 196 and with 19683 but it does not start with 197 or with 196830.

Your function can return any answer of your choice in case either parameter is not positive. Excessively long solutions that use more than 8 lines of code may lose points.

For example, a program that uses the function follows.

```
int main() {
   cout << startsWith(7, 747) << endl;    // prints true
   cout << startsWith(74, 74) << endl;    // prints true
   cout << startsWith(747, 74) << endl;   // prints false
   return 0;
}
```

Answer:

Problem 29 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int x = 0, y = 1, z = 2;
   double b[3] = {1.9, 2.3, 3.0};

   x = max(x + y, z);                    // (a) sets x as the max
   x = maximum(x + z, y, y, z);          // (b) sets x as the maximum
   print(b, x, y);                       // (c) print all the data
   addOn(x, y);                          // (d) add on the value of y to change x
   addOn(y, challenge(y, z));            // (e) adds on a challenge amount to y
   return 0;
}
```

(a) Title line for **max** as called at the line marked (a).

Answer:

(b) Title line for **maximum** as called at the line marked (b).

Answer:

(c) Title line for **print** as called at the line marked (c).

Answer:

(d) Title line for **addOn** as called at the line marked (d).

Answer:

(e) Title line for **challenge** as called at the line marked (e).

Answer:

Problem 30 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
   if (x == y) cout << y;
   if (x > y) y++;
   else x++;
   return x;
}

int main() {
    int a[6] = {5, 3, 1, 4, 4, 1};
    int b = 5, c = 2;
    cout << a[b] + a[c] << endl;                            // line (a)
    cout << fun(b, c) << endl;                              // line (b)
    for (int r = 3; r <= 5; r++) cout << fun(r, c);         // line (c)
    cout << endl;
    fun(a[5], a[4]);  cout << a[4] << endl;                 // line (d)
    cout << fun(a[1], a[3]); cout << a[3] << endl;          // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 31 Write a function called `sumDiff`. The function has two input array parameters `one` and `two` that have the same capacity. The capacity of the arrays is the third parameter of the function.

The function resets entries in `one` and `two` to store the sum and difference of their earlier values. So that if at index `i` the values of `one[i]` and `two[i]` are initially α and β then when the function ends they are $\alpha + \beta$ and $\alpha - \beta$.

Excessively long solutions that use more than 10 lines of code may lose points. An example of a program that calls `sumDiff` follows.

```
int main() {
   int one[4] = {7, 6, 8, 4};
   int two[4] = {2, 6, 3, 9};
   sumDiff(one, two, 4);    // one now stores {9, 12, 11, 13}
                            // and two stores {5, 0, 5, -5}
   return 0;
}
```

Answer:

Problem 32 Write a function called `display`. The function has an integer parameter that is positive. It prints a diagram with horizontal bars to display the digits of the parameter starting from the first digit at the top. Each bar should show numbers that count from 1 to the digit being displayed. If the parameter is not positive your function should not print anything. Excessively long solutions that use more than 10 lines of code may lose points.

For example, a program that uses the function follows.

```
int main() {
   display(31415);
   return 0;
}
```

This should produce the following output:

```
123
1
1234
1
12345
```

Answer:

Problem 33 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int x = 0, y = 1, z = 2;
   double b[3] = {1.9, 2.3, 3.0};

   max(x + y, z);                          // (a) prints the max
   x = second(x, y, y, z, z);              // (b) sets x as the second value
   print(sqrt(b[1]), rand());              // (c) print them all
   interchange(x, y);                      // (d) interchange them
   cout << challenge(y, challenge(y, b[0]));   // (e) a challenge function
   return 0;
}
```

(a) Title line for **max** as called at the line marked (a).
Answer:

(b) Title line for **second** as called at the line marked (b).
Answer:

(c) Title line for **print** as called at the line marked (c).
Answer:

(d) Title line for **interchange** as called at the line marked (d).
Answer:

(e) Title line for **challenge** as called at the line marked (e).
Answer:

Problem 34 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int x, int &y) {
   if (x == y) cout << y;
   if (x > y) y++;
   else x++;
   return x;
}

int main() {
    int a[6] = {5, 3, 1, 4, 4, 1};
    int b = 2, c = 3;
    cout << a[b] + a[c] << endl;                             // line (a)
    cout << fun(b, c) << endl;                               // line (b)
    for (int r = 3; r <= 5; r++) cout << fun(r, c);          // line (c)
    cout << endl;
    fun(a[4], a[5]);  cout << a[4] << endl;                  // line (d)
    cout << fun(a[1], a[3]); cout << a[1] << endl;           // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 35 Write a function called `parity`. The function has two input array parameters `int one[]` and `bool two[]` that have the same capacity. The capacity of the arrays is the third parameter of the function.

The function sets entries in `two` so that `two[i]` is true for exactly those indices for which `one[i]` is even.

Excessively long solutions that use more than 10 lines of code may lose points. An example of a program that calls `parity` follows.

```
int main() {
   int one[4] = {7, 6, 8, 4};
   bool two[4];
   parity(one, two, 4);    // two now stores {false, true, true, true}
   return 0;
}
```

Answer:

Problem 36 Write a function called `display`. The function has an integer parameter that is positive. It prints a diagram with horizontal bars to display the digits of the parameter starting from the first digit at the top. Each bar should should be 9 characters wide and should end with a number of X's that matches the digit being displayed. If the parameter is not positive your function should not print anything. Excessively long solutions that use more than 12 lines of code may lose points.

For example, a program that uses the function follows.

```
int main() {
   display(31415);
   return 0;
}
```

This should produce the following output:

```
      XXX
        X
     XXXX
        X
    XXXXX
```

Answer:

Problem 37 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string course = "CSCI 111";
   int a2[2][3] = {{-2, 4, 3}, {-3, 4, 2}};
   int a[5] = {7, 6, 5, 9, 7};
   cout << lastDigit(19683) * 2   << endl;   // (a) prints: 6 as it is 3 * 2
   cout << randomEntry(a2, 2, 3) << endl;    // (b) prints a random array entry
   cout << department(course) << endl;       // (c) prints: CSCI
   doubleOrNothing(a2[0][0]);           // (d) a2[0][0] is either doubled or made 0 (a random choice)
   cout << odds(a, 5);                  // (e) prints 4: the number of odd entries
   return 0;
}
```

(a) Title line for **lastDigit** as called at the line marked (a).
Answer:

(b) Title line for **randomEntry** as called at the line marked (b).
Answer:

(c) Title line for **department** as called at the line marked (c).
Answer:

(d) Title line for **doubleOrNothing** as called at the line marked (d).
Answer:

(e) Title line for **odds** as called at the line marked (e).
Answer:

Problem 38 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
   if (x == y) cout << y;
   if (x > y) y++;
   else x++;
   return x;
}

int main() {
    int a[6] = {1, 7, 7, 1, 4, 7};
    int b = 5, c = 2;
    cout << a[b] + a[c] << endl;                          // line (a)
    cout << fun(b, c) << endl;                            // line (b)
    for (int r = 3; r <= 5; r++) cout << fun(r, c);       // line (c)
    cout << endl;
    fun(a[5], a[4]);  cout << a[4] << endl;               // line (d)
    cout << fun(a[1], a[3]); cout << a[3] << endl;        // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 39 Write a function called *percentTrue* that returns the percentage of entries in an array that are true. Excessively long solutions that use more than 10 lines of code may lose points.

For example, a program that uses the function *percentTrue* follows.

```
int main() {
   bool x[8] = { true, false, true, false, true, false, true, true};
   cout << percentTrue(x, 8) << " percent " << endl;   // prints 62.5 percent
          // because the 5 true entries make up 62.5% of the array
   return 0;
}
```

Answer:

Problem 40 Write a function called `sumRatios`. The function has two integer parameters that are positive and have the same number of digits all of which are non-zero. It prints the sum of the ratios of corresponding digits. For instance `sumRatios(132,568)` calculates $1/5 + 3/6 + 2/8$ and returns an answer of 0.95. If any parameter has an illegal value your function can operate however you choose. Excessively long solutions that use more than 8 lines of code may lose points.

For example, a program that uses the function follows.

```
int main() {
   cout << sumRatios(132, 568) << endl;  // prints 0.95
   return 0;
}
```

Answer:

Problem 41 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int i = 2;
   int x[5] = {3, 1, 4, 1, 5};
   cout << max(2.1, i, i) << endl;                // (a) prints 2.1
   cout << min(x[2], x[3]) << endl;               // (b) prints 1
   doubleIt(i); cout << i << endl;                // (c) prints 4
   printIt(x, 3);                                 // (d) prints 314
   cout << sum(sum(2,6), sum(x[0],x[1])) << endl; // (e) prints 12
   return 0;
}
```

(a) Title line for **max** as called at the line marked (a).

Answer:

(b) Title line for **min** as called at the line marked (b).

Answer:

(c) Title line for **doubleIt** as called at the line marked (c).

Answer:

(d) Title line for **printIt** as called at the line marked (d).

Answer:

(e) Title line for **sum** as called at the line marked (e).

Answer:

Problem 42 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int x, int &y) {
   if (x == y) cout << y;
   if (x > y) y++;
   else x++;
   return x;
}

int main() {
    int a[6] = {1, 7, 7, 1, 4, 7};
    int b = 2, c = 3;
    cout << a[b] + a[c] << endl;                        // line (a)
    cout << fun(b, c) << endl;                          // line (b)
    for (int r = 3; r <= 5; r++) cout << fun(r, c);     // line (c)
    cout << endl;
    fun(a[4], a[5]);  cout << a[4] << endl;             // line (d)
    cout << fun(a[1], a[3]); cout << a[1] << endl;      // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 43 Write a function called *percentPositive* that returns the percentage of entries in a 2-dimensional array (with 4 columns) that are positive. Excessively long solutions that use more than 10 lines of code may lose points.

For example, a program that uses the function *percentPositive* follows.

```
int main() {
   double x[2][4] = { {1, -1, -2, -3}, {-4, -5, -6, -7}};
   cout << percentPositive(x, 2, 4) << " percent " << endl;   // prints 12.5 percent
          // because the 1 positive number gives 12.5%
   return 0;
}
```

Answer:

Problem 44 Write a function called `digitDifferences`. The function has two integer parameters that are positive and have the same number of digits. It prints the number formed from digits obtained as (positive) differences between corresponding digits in the parameters. For instance `digitDifferences(162,538)` forms a number from the differences $4 = 5 - 1, 3 = 6 - 3$ and $6 = 8 - 2$ getting 436. If parameters have illegal values your function can operate however you choose. Excessively long solutions that use more than 8 lines of code may lose points.

For example, a program that uses the function follows.

```
int main() {
   cout << digitDifferences(162, 538) << endl;  // prints 436
   return 0;
}
```

Answer:

Problem 45 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   double x = 32.1, a2[2][2] = {{3, 2}, {1, 0}};
   bool a[4];
   string name = "Freddy";
   setAs(a, 4, false);              // (a) sets array a to be all false
   cout << printTruth (a, 4);       // (b) prints: false false false false
   cout << mystery(a2, a, x, name); // (c) prints: Freddy is 32.1
   exchange(x, a2[0][0]);             // (d) exchanges the values
   goodDay(name);                   // (e) prints: Hello Freddy
   return 0;
}
```

(a) Title line for **setAs** as called at the line marked (a).

Answer:

(b) Title line for **printTruth** as called at the line marked (b).

Answer:

(c) Title line for **mystery** as called at the line marked (c).

Answer:

(d) Title line for **exchange** as called at the line marked (d).

Answer:

(e) Title line for **goodDay** as called at the line marked (e).

Answer:

Problem 46 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int x, int &y) {
   if (x == y) cout << y;
   if (x > y) y++;
   else x++;
   return x;
}

int main() {
    int a[6] = {3, 1, 4, 1, 5, 9};
    int b = 3, c = 4;
    cout << a[b] + a[c] << endl;                          // line (a)
    cout << fun(b, c) << endl;                            // line (b)
    for (int r = 3; r <= 5; r++) cout << fun(r, c);      // line (c)
    cout << endl;
    fun(a[5], a[4]);  cout << a[4] << endl;               // line (d)
    cout << fun(a[1], a[3]); cout << a[3] << endl;       // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 47 Write a function called *percentPositive* that returns the percentage of entries in an array that are positive. Excessively long solutions that use more than 15 lines of code may lose points.

For example, a program that uses the function *percentPositive* follows.

```
int main() {
   int x[8] = { 1, -1, -2, -3, -4, -5, -6, -7};
   cout << percentPositive(x, 8) << " percent " << endl;   // prints 12.5 percent
          // because the 1 positive number gives 12.5%
   return 0;
}
```

Answer:

Problem 48 Write a function called `lucky7`. The function has an integer parameter that is positive. It calculates an answer by turning the first 7 (from the left) in the parameter to 77.

Only one 7 gets duplicated. If there is no seven in the parameter, the answer is a copy of the parameter. If the parameter is not positive your function can return any convenient answer of your choice. Excessively long solutions that use more than 15 lines of code may lose points.

For example, a program that uses the function follows.

```
int main() {
   cout << lucky7(747) << endl;    // prints 7747
   cout << lucky7(7) << endl;      // prints 77
   cout << lucky7(1234) << endl;   // prints 1234
   cout << lucky7(172737) << endl; // prints 1772737
   return 0;
}
```

Answer:

Problem 49 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[4] = {3, 31, 314, 3141};
   int a2[2][2] = {{3, 31}, {314, 3141}};
   int b = 3, c = 1;

   cout << min(b, 4) << endl;              // (a) prints: 3
   swap(b, c);                             // (b) swaps b and c
   a[0] = max(a, 4);                       // (c) sets a[0] to 3141
   cout << second(a2, 2, 2) << endl;       // (d) prints: 314
   makeZero(a2[1][1]);                     // (e) makes it 0
   return 0;
}
```

(a) Title line for **min** as called at the line marked (a).

Answer:

(b) Title line for **swap** as called at the line marked (b).

Answer:

(c) Title line for **max** as called at the line marked (c).

Answer:

(d) Title line for **second** as called at the line marked (d).

Answer:

(e) Title line for **makeZero** as called at the line marked (e).

Answer:

Problem 50 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int x[], int c) {
   if (c == 1) cout << x[1];
   if (c < 2) return 23;
   if (c == 2) return x[1];
   return x[c] + up(x, c - 1);
}

int main() {
    int x[6] = {3, 1, 4, 1, 5, 9};
    cout << 3 + x[1] << endl;                                 // line (a)
    for (int i = 0; i < 6; i++) cout << x[i];   cout << endl; // line (b)
    cout << up(x, 1) << endl;                                 // line (c)
    cout << up(x, 2) << x[2] << endl;                         // line (d)
    cout << up(x, 4) << endl;                                 // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 51 Write a function called *averageOdd* that returns the average of all of the odd numbers in a 2-dimensional array with 3 columns. If no odd numbers are present, it should return a result of 0. Excessively long solutions that use more than 15 lines of code may lose points.

For example, a program that uses the function *averageOdd* follows.

```
int main() {
   int data[2][3] = {{3, 1, 4},{2, 7, 1}};
   cout << averageOdd(data, 2, 3) << endl;   // prints 3.0
                           // because the odd entries 3, 1, 7, 1 average to 3.0
   return 0;
}
```

Answer:

Problem 52 Write a function called *interlaceDigits* that uses two positive integer parameters with the same number of digits and returns an integer that begins with the first digit of the first parameter, then the first digit of the second parameter, then the second digits of the parameters, and so on until all digits are used. If a negative parameter is given, or if parameters with unequal numbers of digits are given your function can return any result of your choosing. Excessively long solutions that use more than 10 lines of code may lose points.

For example, a program that uses the function *interlaceDigits* follows.

```
int main() {
   cout << interlaceDigits(1, 2) << endl;              // prints 12
   cout << interlaceDigits(117, 302) << endl;          // prints 131072
   cout << interlaceDigits(1357, 2468) << endl;        // prints 12345678
   return 0;
}
```

Answer:

Problem 53 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int i = 123, arr1 [3] = {1, 2, 3}, arr2 [2][2] = {{1, 0}, {2, 4}};
   double d1 = 1.23, d2 = 12.3;
   printLine (arr2, 2, 2);          // (a) prints: 1 0 2 4
   printFancy (arr1, 3);            // (b) prints: 1 * 2 ** 3 ***
   cout << doNothing (i, (int) d1); // (c) prints: This is a useless function
   switchValues (d1, d2); // (d) switches the values: now, d1 = 12.3, d2 = 1.23
   goodDayWishes ();                // (e) prints: Have a good day
   return 0;
}
```

(a) Title line for **printLine** as called at the line marked (a).
Answer:
(b) Title line for **printFancy** as called at the line marked (b).
Answer:
(c) Title line for **doNothing** as called at the line marked (c).
Answer:
(d) Title line for **switchValues** as called at the line marked (d).
Answer:
(e) Title line for **goodDayWishes** as called at the line marked (e).
Answer:

Problem 54 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int x[], int c) {
   if (c == 1) cout << x[1];
   if (c < 2) return 47;
   if (c == 2) return x[1];
   return x[c] + up(x, c - 1);
}

int main() {
    int x[6] = {2, 7, 1, 8, 2, 8};
    cout << 3 + x[1] << endl;                                 // line (a)
    for (int i = 0; i < 6; i++) cout << x[i];   cout << endl; // line (b)
    cout << up(x, 1) << endl;                                 // line (c)
    cout << up(x, 2) << x[2] << endl;                         // line (d)
    cout << up(x, 4) << endl;                                 // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 55 Write a function called *numberNegative* that returns the number of negative elements in a 2-dimensional array with 3 columns. Excessively long solutions that use more than 12 lines of code may lose points.

For example, a program that uses the function *numberNegative* follows.

```
int main() {
   double data[2][3] = {{-3.0, 1, 4.5},{-2.2, 7, 1.4}};
   cout << numberNegative(data, 2, 3) << endl;   // prints 2
                            // because there are 2 negatives -3.0 and -2.2
   return 0;
}
```

Answer:

Problem 56 Write a function called *interweaveDigits* that uses two positive integer parameters with the same number of digits and returns an integer that begins with the first digit of the second parameter, then the first digit of the first parameter, then the second digits of the parameters, and so on until all digits are used. If a negative parameter is given, or if parameters with unequal numbers of digits are given your function can return any result of your choosing. Excessively long solutions that use more than 10 lines of code may lose points.

For example, a program that uses the function *interweaveDigits* follows.

```
int main() {
   cout << interweaveDigits(2, 1) << endl;              // prints 12
   cout << interweaveDigits(302, 117) << endl;          // prints 131072
   cout << interweaveDigits(2468, 1357) << endl;        // prints 12345678
   return 0;
}
```

Answer:

Problem 57 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[4] = {3, 31, 314, 3141};
   int a2[2][2] = {{3, 31}, {314, 3141}};
   int b = 3, c = 1;

   cout << min(b, 4) << endl;              // (a) prints: 3
   swap(b, c);                             // (b) swaps b and c
   a[0] = max(a, 4);                       // (c) sets a[0] to 3141
   cout << second(a2, 2, 2) << endl;       // (d) prints: 314
   makeZero(a2[1][1]);                     // (e) makes it 0
   return 0;
}
```

(a) Title line for **min** as called at the line marked (a).
Answer:

(b) Title line for **swap** as called at the line marked (b).
Answer:

(c) Title line for **max** as called at the line marked (c).
Answer:

(d) Title line for **second** as called at the line marked (d).
Answer:

(e) Title line for **makeZero** as called at the line marked (e).
Answer:

Problem 58 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void up(int x[][3], int rows, int cols) {
   for (int c = 0; c < cols; c++) for (int r = 0; r < rows; r++)
      cout << 10 + x[r][c];
   cout << endl;
}

int main() {
    int x[3][3] = {{3, 1, 4}, {1, 5, 9}, {2, 6, 5}};
    cout << x[2][2] << endl;                             // line (a)
    cout << x[x[2][0]][x[2][0]] << endl;                 // line (b)
    for (int r = 0; r < 2; r++) cout << x[2][r] << endl; // line (c)
    up(x, 1, 1);                                         // line (d)
    up(x, 2, 2);                                         // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 59 Write a function called *sum3* that returns the sum of all of the 3-digit numbers in an array. Excessively long solutions that use more than 12 lines of code may lose points.

For example, a program that uses the function *sum3* follows.

```
int main() {
   int x[6] = {3, 31, 314, 111, 4000, 100};
   cout << sum3(x, 6) << endl;   // prints 525
          // because the 3-digit numbers 314, 111, 100 add to 525
   return 0;
}
```

Answer:

Problem 60 Write a function called *gcb* that uses two positive integer parameters and returns the greatest common beginning to the two numbers. For example, the greatest common beginning of 1235 and 1248 is 12. If the two parameters begin differently the function should return 0. If a negative parameter is given your function can return any result of your choosing. Excessively long solutions that use more than 10 lines of code may lose points.

For example, a program that uses the function *gcb* follows.

```
int main() {
   cout << gcb(123, 223) << endl;          // prints 0
   cout << gcb(117, 119) << endl;          // prints 11
   cout << gcb(1357, 136578) << endl;      // prints 13
   return 0;
}
```

Answer:

Problem 61 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int i = 123, arr1 [3] = {1, 2, 3}, arr2 [2][2] = {{1, 0}, {2, 4}};
   double d1 = 1.23, d2 = 12.3;
   printLine (arr2, 2, 2);          // (a) prints: 1 0 2 4
   printFancy (arr1, 3);            // (b) prints: 1 * 2 ** 3 ***
   cout << doNothing (i, (int) d1); // (c) prints: This is a useless function
   switchValues (d1, d2); // (d) switches the values: now, d1 = 12.3, d2 = 1.23
   goodDayWishes ();                // (e) prints: Have a good day
   return 0;
}
```

(a) Title line for **printLine** as called at the line marked (a).

Answer:

(b) Title line for **printFancy** as called at the line marked (b).

Answer:

(c) Title line for **doNothing** as called at the line marked (c).

Answer:

(d) Title line for **switchValues** as called at the line marked (d).

Answer:

(e) Title line for **goodDayWishes** as called at the line marked (e).

Answer:

Problem 62 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void up(int x[][3], int rows, int cols) {
   for (int c = 0; c < cols; c++) for (int r = 0; r < rows; r++)
      cout << x[r][c] - 7;
   cout << endl;
}

int main() {
    int x[3][3] = {{2, 7, 1}, {8, 2, 8}, {1, 8, 2}};
    cout << x[2][2] << endl;                                // line (a)
    cout << x[x[2][0]][x[2][0]] << endl;                    // line (b)
    for (int r = 0; r < 2; r++) cout << x[2][r] << endl; // line (c)
    up(x, 1, 1);                                            // line (d)
    up(x, 2, 2);                                            // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 63 Write a function called *numberFreddy* that returns the number of entries of an array equal to `"Freddy"`. Excessively long solutions that use more than 12 lines of code may lose points.

For example, a program that uses the function *numberFreddy* follows.

```
int main() {
   string data[5] = {"Kelly", "Jack", "Freddy", "Arthur", "Freddy"};
   cout << numberFreddy(data, 5) << endl;   // prints 2
   return 0;
}
```

Answer:

Problem 64 Write a function called *gce* that uses two positive integer parameters and returns the greatest common ending to the two numbers. For example, the greatest common ending of 1234 and 134 is 34. If the two parameters end differently the function should return 0. If a negative parameter is given your function can return any result of your choosing. Excessively long solutions that use more than 10 lines of code may lose points.

For example, a program that uses the function *gce* follows.

```
int main() {
   cout << gce(123, 123) << endl;          // prints 123
   cout << gce(123, 223) << endl;          // prints 23
   cout << gce(117, 119) << endl;          // prints 0
   cout << gce(1357, 13657) << endl;       // prints 57
   return 0;
}
```

Answer:

Problem 65 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   char x = 'a', y = 'b', z = 'c';
   string a[3] = {"A", "B", "Freddy"};
   bool b[2][2] = {{true, false},{true,true}};
   int c = 0;

   c = subtract(z, y);                    // (a) sets c to the difference 1
   welcomeUser(a[2]);                     // (b) print out "Hello Freddy"
   deFred(a[2]);                          // (c) change it to "Anon"
   reset(b, 2, 2, 2 == 2);                // (d) sets the array to be all true
   cout << addOn(addOn(a[2],x),y);        // (e) function adds on a character
   return 0;
}
```

(a) Title line for **subtract** as called at the line marked (a).

Answer:

(b) Title line for **welcomeUser** as called at the line marked (b).

Answer:

(c) Title line for **deFred** as called at the line marked (c).

Answer:

(d) Title line for **reset** as called at the line marked (d).

Answer:

(e) Title line for **addOn** as called at the line marked (e).

Answer:

Problem 66 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void up(int x[][3], int rows, int cols) {
   for (int c = 0; c < cols; c++) for (int r = 0; r < rows; r++)
      cout << (char) ('A' + x[r][c]);
   cout << endl;
}

void recursive(int x[][3], int r) {
    if (r == 0) {
       cout << endl;
       return;
    }
    cout << x[r - 1][r - 1];
    recursive(x, r - 1);
}

int main() {
    int x[3][3] = {{3, 1, 4}, {1, 5, 9}, {2, 6, 5}};
    cout << x[1][1] << x[0][2] << endl;                  // line (a)
    cout << x[x[1][0]][x[1][0]] << endl;                 // line (b)
    for (int c = 0; c < 3; c++) cout << x[2][c] << endl; // line (c)
    up(x, 2, 2);                                         // line (d)
    recursive(x,3);                                      // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 67 Write a function called *goodStudent* that gives the name of a student who scores at least 8 points on a quiz. The function uses three parameters: an array of names, an array of scores and a count of students. If more than one student scores at least 8, the first name in the array with a score of at least 8 is returned. If no student does well a result of "Nobody" is returned.

For example, a program that uses the function *goodStudent* follows.

```
int main() {
   string students[4] = {"Freddy", "Kelly", "Arthur", "Jack"};
   int scores[4] = {0, 8, 7, 10};
   int hardQuiz[4] = {0, 1, 1, 2};
   cout << goodStudent(students, scores, 4) << endl;   // prints Kelly
   cout << goodStudent(students, hardQuiz, 4) << endl; // prints Nobody
   return 0;
}
```

Answer:

Problem 68 Write a function called *biggerDigits* that uses two positive integer parameters with the same number of digits and returns an integer whose digit in each position is the bigger of the two digits in that position in the input parameters. If a negative parameter is given, or if parameters with unequal numbers of digits are given your function can return any result of your choosing.

For example, a program that uses the function *biggerDigits* follows.

```
int main() {
   cout << biggerDigits(567, 765) << endl;          // prints 767
   cout << biggerDigits(123456, 444444) << endl;    // prints 444456
   cout << biggerDigits(999, 111) << endl;          // prints 999
   return 0;
}
```

Answer:

Problem 69 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string x = "a", y = "b", z = "c";
   char a[3] = {'A', 'B', 'C'};
   int b[2][2] = {{1,0},{1, 1}};
   bool c = false;

   c = sameLength(x, y, "z");             // (a) sets c to true
   courseName(a[2]);                      // (b) print out "A course about C"
   cout << plusplus(a, 2);                // (c) print "A++ B++ C++"
   reset(b, 2, 2, a[2] - a[0]);           // (d) sets all array entries to 2
   cout << addOn(addOn(z,a[0]),a[0]);     // (e) function adds on a character
   return 0;
}
```

(a) Title line for **sameLength** as called at the line marked (a).
Answer:

(b) Title line for **courseName** as called at the line marked (b).
Answer:

(c) Title line for **plusplus** as called at the line marked (c).
Answer:

(d) Title line for **reset** as called at the line marked (d).
Answer:

(e) Title line for **addOn** as called at the line marked (e).
Answer:

Problem 70 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive(int x[][3], int r) {
    if (r <= -1) return 1;
    return x[r][r] + recursive(x, r - 1);
}

int main() {
    int x[3][3] = {{2, 7, 1}, {8, 2, 8}, {1, 8, 2}};
    cout << x[1][2] << x[2][1] << endl;                  // line (a)
    cout << x[x[1][1]][x[0][0]] << endl;                 // line (b)
    for (int c = 0; c < 3; c++) cout << x[c][c] << endl; // line (c)
    cout << recursive(x, -1) << endl;                    // line (d)
    cout << recursive(x, 1) << endl;                     // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 71 Write a function called *bestStudents* that prints the names of all students that get the highest score on a quiz. The function uses three parameters: an array of names, an array of scores and a count of students.

For example, a program that uses the function *bestStudents* follows.

```
int main() {
   string students[4] = {"Freddy", "Kelly", "Arthur", "Jack"};
   int scores[4] = {0, 1, 1, 1};
   bestStudents(4, scores, students);   // prints Kelly Arthur Jack
   return 0;
}
```

Answer:

Problem 72 Write a function called *digitDifference* that uses two positive integer parameters with the same number of digits and returns an integer whose digit in each position is the (positive) difference between the two digits in that position in the input parameters. If a negative parameter is given, or if parameters with unequal numbers of digits are given your function can return any result of your choosing.

For example, a program that uses the function *digitDifference* follows.

```
int main() {
   cout << digitDifference(567, 765) << endl;          // prints 202
   cout << digitDifference(123456, 444444) << endl;    // prints 321012
   cout << digitDifference(999, 111) << endl;          // prints 888
   cout << digitDifference(999, 987) << endl;          // prints  12
   return 0;
}
```

Answer:

Problem 73 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string name = "Freddy Next Door";
   int a2[2][3] = {{-2, 4, 3}, {-3, 4, 2}};
   int a[5] = {7, 6, 5, 9, 7};
   cout << firstLetters(name, name) << endl;      // (a) prints: F F
   cout << sumAll(a, 5, a, 5) << endl;            // (b) prints: 68 by summing twice
   cout << middleInitial(name) << endl;           // (c) prints: N
   makeRandom(a2, 2, 3);                          // (d) reset the array with random entries
   if (countIt(name, countIt(middleInitial(name), 5.0)) > 0) // (e) mystery
      cout << "Yes\n";
   return 0;
}
```

(a) Title line for **firstLetters** as called at the line marked (a).
Answer:

(b) Title line for **sumAll** as called at the line marked (b).
Answer:

(c) Title line for **middleInitial** as called at the line marked (c).
Answer:

(d) Title line for **makeRandom** as called at the line marked (d).
Answer:

(e) Title line for **countIt** as called at the line marked (e).
Answer:

Problem 74 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1}, x = 10, y = 1000;
 // (a) Finds the cube, here -27
   cout << cube(-3) << endl;
 // (b) Finds a random number between 1 and x
   cout << random(x) << endl;
 // (c) Prints the ratio as a percentage, here 12.5% for 1/8
   cout << percentage(1, 8) << "%" << endl;
 // (d) reverse print the array here 1413  (no spaces)
   reversePrint(a, 4);
 // (e) determine whether x or y has more digits, assume x and y both positive
   if (hasMore(x,y)) cout << "x is longer\n";
   return 0;
}
```

(a) `int cube(int x)`

Answer:

(b) `int random(int x)`

Answer:

(c) `double percentage(int x, int y)`

Answer:

(d) `void reversePrint(int x[], int cap)`

Answer:

(e) `bool hasMore(int x, int y)`

Answer:

Problem 75 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int xy(int x, string &y) {
  if (x > 0) y = "error";
  else y = "fine";
  if (x <= 0) return 3;
  return x % 10 + 10 * xy(x/10, y);
}

int main() {
    int c = 9, x = 10;
    string y;
    if ((x % c) >= (c % x)) cout << c;        // line (a)
    cout << endl;
    for(c = 8; c > x - c; c--) cout << c;     // line (b)
    cout << endl;
    cout << xy(-2, y) << endl;                // line (c)
    cout << y << endl;                        // line (d)
    cout << xy(3145, y) << endl;              // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 76 Write a function called *toNumber* that uses an array of integers each entry of which is between 0 and 9 and returns an integer formed by using the entries as its digits. If input array entries are out of range, you can return any answer of your choosing. Your function should not use more than 5 lines of code.

For example, a program that uses the function *toNumber* follows.

```
int main() {
   int a[6] = {3,1,4,1,5,9};
   cout << toNumber(a, 6) << endl;          // prints 314159
   cout << toNumber(a, 6) + 1 << endl;      // prints 314160
   return 0;
}
```

Answer:

Problem 77 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int i = 2;
   int x[5] = {3, 1, 4, 1, 5};
   cout << max(2.1, i, 1.5) << endl;                        // (a) prints 2.1
   cout << min(x[2], x[3]) << endl;                         // (b) prints 1
   negateIt(i); cout << i + 1 << endl;                      // (c) prints -1
   printArray(x, 5);                                        // (d) prints 31415
   if (sum(sum(2.1, 6), 1) > 0) cout << "big\n";            // (e) prints big
   return 0;
}
```

(a) Title line for **max** as called at the line marked (a).
Answer:

(b) Title line for **min** as called at the line marked (b).
Answer:

(c) Title line for **negateIt** as called at the line marked (c).
Answer:

(d) Title line for **printArray** as called at the line marked (d).
Answer:

(e) Title line for **sum** as called at the line marked (e).
Answer:

Problem 78 Consider the following C++ program.

```
#include <iostream>
using namespace std;

double sum(int x[], int cap, int jump) {
  double ans = 0.0;
  for (int i = 0; i < cap; i+= jump)
     ans += x[i];
  return ans / 10.0;
}

int main() {
    int x[6] = {2, 1, 3, 0, 4, 9};
    cout << x[2] << endl;                     // line (a)
    cout << x[5/3] << endl;                   // line (b)
    cout << x[x[2]] << endl;                  // line (c)
    cout <<  sum(x, 6, 1) << endl;            // line (d)
    cout <<  sum(x, 4, 2) << endl;            // line (e)
    return 0;
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 79 Write a function called *maxGap* that calculates the largest gap between adjacent entries of an array. (A gap between two numbers is the absolute value of their difference.)

For example, a program that uses the function *maxGap* follows.

```
int main() {
   int x[5] = {2, 9, 1, 6, 3};
   cout << maxGap(x, 5) << endl;   // prints 8 corresponding to the gap from 1 to 9.
   return 0;
}
```

Answer:

Problem 80 Write a function called *secondDown* that returns the result of decreasing the second digit of a positive integer parameter by 1. (If the second digit is already 0, then the value of the parameter is returned. If the parameter is less than 10, then the function can return any answer of your choice.)

For example, a program that uses the function *secondDown* follows.

```
int main() {
   cout << secondDown(243)  << endl;       // prints 233
   cout << secondDown(2048) << endl;       // prints 2048
   cout << secondDown(1234) + 1 << endl;   // prints 1135
   return 0;
}
```

Answer:

Problem 81 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int i = 3;
   string s = "Hello";
   int x[5] = {2, 7, 1, 8, 2};
   cout << min(i, 2.1, x[0]) << endl;             // (a) prints: 2.1
   cout << max(x[2], 3) << endl;                  // (b) prints: 3
   cout << doubleIt(i) << endl;                   // (c) prints:  2 x 3
   hi(s);  cout << s << endl;                     // (d) prints: Hi
   cout << sum(sum(2,6,i), i, i) << endl;         // (e) prints: 17
   return 0;
}
```

(a) Title line for **min** as called at the line marked (a).
Answer:

(b) Title line for **max** as called at the line marked (b).
Answer:

(c) Title line for **doubleIt** as called at the line marked (c).
Answer:

(d) Title line for **hi** as called at the line marked (d).
Answer:

(e) Title line for **sum** as called at the line marked (e).
Answer:

Problem 82 Consider the following C++ program.

```
#include <iostream>
using namespace std;

double sum(int x[], int cap, int jump) {
  double ans = 0.0;
  for (int i = 0; i < cap; i+= jump)
     ans += x[i];
  return ans / 5.0;
}

int main() {
    int x[6] = {5, 4, 3, 2, 1, 9};
    cout << x[3] << endl;                     // line (a)
    cout << x[5/3] << endl;                   // line (b)
    cout << x[x[3]] << endl;                  // line (c)
    cout << sum(x, 6, 1) << endl;             // line (d)
    cout << sum(x, 5, 2) << endl;             // line (e)
    return 0;
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 83 Write a function called *sumGaps* that calculates the sum of the gaps between adjacent entries of an array. (A gap between two numbers is the absolute value of their difference.)

For example, a program that uses the function *sumGaps* follows.

```
int main() {
   int x[5] = {3, 1, 4, 1, 5};
   cout << sumGaps(x, 5) << endl;   // prints 12 corresponding to the sum of gaps 2 + 3 + 3 + 4.
   return 0;
}
```

Answer:

Problem 84 Write a function called *thirdDown* that returns the result of decreasing the third digit of a positive integer parameter by 1. (If the third digit is already 0, then the value of the parameter is returned. If the parameter is less than 100, then the function can return any answer of your choice.)

For example, a program that uses the function *thirdDown* follows.

```
int main() {
   cout << thirdDown(1243)  << endl;       // prints 1233
   cout << thirdDown(12048) << endl;       // prints 12048
   cout << thirdDown(11234) + 1 << endl;   // prints 11135
   return 0;
}
```

Answer:

Problem 85 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int i = 2;
   double x[5] = {3.0, 1.1, 4.2, 1.3, 5.4};
   cout << max(4.1, x[i] / 10, i) << endl;          // (a) prints 4.1
   cout << min(x[2], x[3]) << endl;                 // (b) prints 1.3
   squareIt(i); cout << i << endl;                  // (c) prints 4
   squareAll(x, 5); cout << x[0] << endl;           // (d) prints 9.0
   if (f(f(x[0])) > 2) cout << "+" << endl;         // (e) prints +
   return 0;
}
```

(a) Title line for **max** as called at the line marked (a).

Answer:

(b) Title line for **min** as called at the line marked (b).

Answer:

(c) Title line for **squareIt** as called at the line marked (c).

Answer:

(d) Title line for **squareAll** as called at the line marked (d).

Answer:

(e) Title line for **f** as called at the line marked (e).

Answer:

Problem 86 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void down(int x[], int cap, int gap) {
  for (int i = 0; i < cap; i+= gap)
     x[i] -= gap;
}

int main() {
    int x[6] = {3, 1, 4, 1, 5, 9};
    cout << x[5] / 4 << endl;                 // line (a)
    cout << x[5/4] << endl;                   // line (b)
    cout << x[x[5]/4] << endl;                // line (c)
    down(x, 6, 1); cout << x[1] << endl;      // line (d)
    down(x, 6, 3); cout << x[1] << endl;      // line (e)
    return 0;
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 87 Write a function called *evenSum* that calculates the sum of those entries in an array that are even numbers.

For example, a program that uses the function *evenSum* follows.

```
int main() {
   int x[8] = {3, 1, 4, 1, 5, 9, 2, 6};
   cout << evenSum(x, 8) << endl;   // prints 12
   // The even entries are 4, 2, 6 and these add to 12
   return 0;
}
```

Answer:

Problem 88 Write a function called *allEven* that reports whether all the digits in a positive integer parameter are even.

For example, a program that uses the function *allEven* follows.

```
int main() {
   int x;
   cout << "Enter a number: ";
   cin >> x;
   if (allEven(x)) cout << "All digits are even." << endl;
   else cout << "Not all digits are even." << endl;
   return 0;
}
```

If the user entered any of 2, 242 or 2048, the program would print *All digits are even.* But if the user entered any of 1, 21, 1248 or 555, the program would print *Not all digits are even.*

Answer:

Problem 89 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   double x = 0.0, y = 1.1, z = 2.5;
   int array[5] = {3,1,4,1,5};
   string s = "Hello";

   z = average(x, y, z);                  // (a) sets z to average 1.2
   addStar(s);                            // (b) replaces s by "Hello*"
   cout << bigger(average(x,y,z), 7.5);   // (c) prints 7.5 because it is bigger
   cout << endl;
   printArray(array, 5);                  // (d) prints array: 3 1 4 1 5
   subtract(array[0], array, 5);          // (e) subtracts array[0] from other elements
   printArray(array, 5);                  //   output will now be 0 -2 1 -2 2
   return 0;
}
```

(a) Title line for **average** as called at the line marked (a).

Answer:

(b) Title line for **addStar** as called at the line marked (b).

Answer:

(c) Title line for **bigger** as called at the line marked (c).

Answer:

(d) Title line for **printArray** as called at the line marked (d).

Answer:

(e) Title line for **subtract** as called at the line marked (e).

Answer:

Problem 90 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int x, int &y) {
  if (x < 0) y = -x;
  if (x <= 0) return 0;
  return x % 10 + 2 * fun(x/100, y);
}

int main() {
    int c, x = 1, y = 5;
    if ((x % y) > (y % x)) cout << x;         // line (a)
    cout << endl;
    for(c = x; c < y; c++) cout << c;         // line (b)
    cout << endl;
    cout << fun(-2, y) << endl;               // line (c)
    cout << y << endl;                        // line (d)
    cout << fun(31459, y) << endl;            // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 91 Write a function called *subtractFirst* that subtracts the value of the first element from every element in an array.

For example, a program that uses the function *subtractFirst* follows.

```
int main() {
   int array[6] = {3,1,4,1,5,9};
   subtractFirst(array, 6);
   for (int i = 0; i < 6; i++)
      cout << array[i] << " ";   //   Output will be 0 -2 1 -2 2 6
   return 0;
}
```

Answer:

Problem 92 Write a function called *cutAfter7* that cuts a positive integer parameter after the first digit 7 that it contains. Parameters that are not positive should be returned without any change.

For example, a program that uses the function *cutAfter7* follows.

```
int main() {
   cout << cutAfter7(765) << endl;     // prints 7
   cout << cutAfter7(765765) << endl;  // prints 7
   cout << cutAfter7(666) << endl;     // prints 666
   cout << cutAfter7(107) << endl;     // prints 107
   cout << cutAfter7(107007) << endl;  // prints 107
   return 0;
}
```

Answer:

Problem 93 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   double z = 2.5;
   int array[5] = {3,1,4,1,5};
   string s = "Hello";

   z = average(array, 5);                 // (a) sets z to average 2.8
   addTwice(s,"**");                      // (b) replaces s by "Hello**Hello**"
   cout << sum(average(array, 5), 1.2);   // (c) 4.0 the sum of 1.2 and the average
   cout << endl;
   cout << someArray(array, 3);           // (d) prints 3 elements: 3 1 4
   count(array[1], array, 5);             // (e) print count of copies of array[1] in array
   return 0;
}
```

(a) Title line for **average** as called at the line marked (a).
Answer:

(b) Title line for **addTwice** as called at the line marked (b).
Answer:

(c) Title line for **sum** as called at the line marked (c).
Answer:

(d) Title line for **someArray** as called at the line marked (d).
Answer:

(e) Title line for **count** as called at the line marked (e).
Answer:

Problem 94 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int xy(int x, string &y) {
  if (x < 0) y = "error";
  else y = "ok";
  if (x <= 0) return 5;
  return x % 10 + 10 * xy(x/100, y);
}

int main() {
    int c = 4, x = 1;
    string y;
    if ((x % c) == (c % x)) cout << c;        // line (a)
    cout << endl;
    for(c = 5; c > x; c--) cout << c;         // line (b)
    cout << endl;
    cout << xy(-2, y) << endl;               // line (c)
    cout << y << endl;                        // line (d)
    cout << xy(31459, y) << endl;            // line (e)
}
```

(a) What is the output at line (a)?
Answer:

(b) What is the output at line (b)?
Answer:

(c) What is the output at line (c)?
Answer:

(d) What is the output at line (d)?
Answer:

(e) What is the output at line (e)?
Answer:

Problem 95 Write a function called *subtractAverage* that subtracts the average value of an array from every element in an array.

For example, a program that uses the function *subtractAverage* follows.

```
int main() {
   double array[6] = {3,1,4,1,5};   // has average 2.8
   subtractAverage(array, 5);
   for (int i = 0; i < 5; i++)
      cout << array[i] << " ";   //   Output will be 0.2 -1.8 1.2 -1.8 2.2
   return 0;
}
```

Answer:

Problem 96 Write a function called *cutBefore7* that cuts a positive integer parameter before the first digit 7 that it contains. Parameters that are not positive should be returned without any change.

For example, a program that uses the function *cutBefore7* follows.

```
int main() {
   cout << cutBefore7(667) << endl;     // prints 66
   cout << cutBefore7(677) << endl;     // prints 6
   cout << cutBefore7(666) << endl;     // prints 666
   cout << cutBefore7(766) << endl;     // prints 0
   cout << cutBefore7(567567) << endl;  // prints 56
   return 0;
}
```

Answer:

Problem 97 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string s; char c = 'A'; double d = 1.1;
   int a[4] = {3, 1, 4, 2};
   bool b[2][3] = {{true, false, true}, {false, true, true}};

   s = asString(c); cout << s << endl;      // (a) prints: A
   doubleIt(d); cout << d << endl;          // (b) prints: 2.2
   doubleThem(a, 4);  cout << a[0] << endl; // (c) prints 6
   printArray(b, 2, 3);                     // (d) prints TFT FTT
   c = randomLetter(); cout << c << endl;   // (e) prints a random letter eg Z
   return 0;
}
```

(a) Title line for **asString** as called at the line marked (a).
Answer:

(b) Title line for **doubleIt** as called at the line marked (b).
Answer:

(c) Title line for **doubleThem** as called at the line marked (c).
Answer:

(d) Title line for **printArray** as called at the line marked (d).
Answer:

(e) Title line for **randomLetter** as called at the line marked (e).
Answer:

Problem 98 Consider the following C++ program.

```
#include <iostream>
using namespace std;

double down(int x[], int cap, int gap) {
  double ans = 0.0;
  for (int i = 0; i < cap; i+= gap)
     ans += x[i];
  return ans / 10;
}

int main() {
    int x[4] = {3, 1, 4, 1};
    cout << x[2] << endl;                     // line (a)
    cout << x[5/3] << endl;                   // line (b)
    cout << down(x, 4, 1) << endl;            // line (c)
    cout << down(x, 4, 3) << endl;            // line (d)
    cout << down(x, x[0], x[x[1]]) << endl;   // line (e)
}
```

(a) What is the output at line (a)?
Answer:

(b) What is the output at line (b)?
Answer:

(c) What is the output at line (c)?
Answer:

(d) What is the output at line (d)?
Answer:

(e) What is the output at line (e)?
Answer:

Problem 99 Write a function called *diff2* that returns the absolute value of the difference of the first two digits of a positive integer parameter. If the parameter has just one digit, that digit should be returned.

For example, a program that uses the function *diff2* follows.

```
int main() {
   cout << diff2(7070);        // prints 7
   cout << endl;
   cout << diff2(7907);        // prints 2
   cout << endl;
   cout << diff2(7);           // prints 7
   cout << endl;
   return 0;
}
```

Answer:

Problem 100 Write a function called *evenLessOdd* that returns the sum of the even valued entries minus the sum of the odd valued entries in an array of integers.

For example, a program that uses the function *evenLessOdd* follows. The first output is $2 = 8 - 1 - 5$ and the second is $-10 = -1 - 1 - 5 - 3$.

```
int main() {
   int x[3] = {8, 1, 5};
   int y[4] = {1, 1, 5, 3};
   cout << evenLessOdd(x, 3) << endl;       // prints 2
   cout << evenLessOdd(y, 4) << endl;       // prints -10
   return 0;
}
```

Answer:

Problem 101 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string s; char c = 'A'; double d = 1.1;
   int a[4] = {3, 1, 4, 2};
   bool b[2][3] = {{true, false, true}, {false, true, true}};

   d = randomNumber(); cout << d << endl; // (a) prints a random number eg 1.5
   printThem(a, 4);                                         // (b) prints 3142
   b[1][0] = majority(b, 2, 3); if (b[1][0]) cout << "true\n"; // (c) prints true
   doubleIt(a[1]); cout << a[1] << endl;                    // (d) prints: 2
   s = asString(b[0][0]); cout << s << endl;                // (e) prints: True
   return 0;
}
```

(a) Title line for **randomNumber** as called at the line marked (a).
Answer:

(b) Title line for **printThem** as called at the line marked (b).
Answer:

(c) Title line for **majority** as called at the line marked (c).
Answer:

(d) Title line for **doubleIt** as called at the line marked (d).
Answer:

(e) Title line for **asString** as called at the line marked (e).
Answer:

Problem 102 Consider the following C++ program.

```
#include <iostream>
using namespace std;

double down(int x[], int cap, int &gap) {
  double ans = 0.0;
  for (int i = 0; i < cap; i+= gap)
     ans += x[i];
  gap += 2;
  return ans / 10;
}

int main() {
    int x[4] = {3, 2, 1, 8};
    int a = 4, b = 1;
    cout << x[7/3] << endl;                    // line (a)
    cout << down(x, a, b) << endl;             // line (b)
    cout << down(x, a, b) << endl;             // line (c)
    cout << down(x, x[0], x[x[2]]) << endl;    // line (d)
    cout << x[2] << endl;                      // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 103 Write a function called *unlucky* that returns an answer of *true* if the first two digits of a positive integer parameter add to 13. Otherwise it returns *false*. (It returns *false* if the parameter has fewer than 2 digits.) For example, a program that uses the function *unlucky* follows.

```
int main() {
   if (unlucky(6789)) cout << "Unlucky!\n";  // prints Unlucky!
   if (unlucky(6889)) cout << "Unlucky!\n";  // prints
   if (unlucky(6)) cout <<    "Unlucky!\n";  // prints
   if (unlucky(49)) cout <<   "Unlucky!\n";  // prints Unlucky!
   return 0;
}
```

Answer:

Problem 104 Write a function called *lastOdd* that returns the last odd valued entry in an array or returns 0 if there is no odd value.

For example,

```
int main() {
   int x[3] = {8, 1, 7};
   int y[5] = {1, 2, 5, 4, 6};
   int z[2] = {2, 2};
   cout << lastOdd(x, 3) << endl;       // prints 7
   cout << lastOdd(y, 5) << endl;       // prints 5
   cout << lastOdd(z, 2) << endl;       // prints 0
   return 0;
}
```

Answer:

Problem 105 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string s; char c = 'A'; double d = 4.0;
   int a[4] = {3, 1, 4, 2};
   bool b[2][3] = {{true, false, true}, {false, true, true}};

   printThem(b, 2, 3);                           // (a) prints TFT FTT
   fixLies(b, 2, 3); printThem(b, 2, 3);         // (b) prints FTF TFF
   d = cubeIt(d); cout << d << endl;             // (c) prints: 64.0
   cubeInt(a[2]); cout << a[2] << endl;          // (d) prints: 64
   a[1] = reverseDigits(a[2]); cout << a[1] << endl; // (e) prints: 1
   return 0;
}
```

(a) Title line for **printThem** as called at the line marked (a).
Answer:

(b) Title line for **fixLies** as called at the line marked (b).
Answer:

(c) Title line for **cubeIt** as called at the line marked (c).
Answer:

(d) Title line for **cubeInt** as called at the line marked (d).
Answer:

(e) Title line for **reverseDigits** as called at the line marked (e).
Answer:

Problem 106 Consider the following C++ program.

```
#include <iostream>
using namespace std;

double down(int x[], int cap, int &gap) {
  double ans = 0.0;
  for (int i = 0; i < cap; i+= gap)
     ans += x[i];
  gap += 2;
  return ans / 10;
}

int main() {
    int x[4] = {9, 1, 3, 2};
    int a = 4, b = 2;
    cout << x[9/3] << endl;                    // line (a)
    cout << down(x, a, b) << endl;             // line (b)
    cout << down(x, a, b) << endl;             // line (c)
    cout << down(x, x[2], x[x[2]]) << endl;    // line (d)
    cout << x[3] << endl;                      // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 107 Write a function called *add7* that returns an answer found by putting a 7 in front of the first digit of a positive integer.

For example, a program that uses the function *add7* follows.

```
int main() {
   cout << add7(1) << "\n";    // prints 71
   cout << add7(17) << "\n";   // prints 717
   cout << add7(456) << "\n";  // prints 7456
   return 0;
}
```

Answer:

Problem 108 Write a function called *indexFirstOdd* that returns the index of the first odd valued entry in an array or returns -1 if there is no odd value. (The index of an entry is its position in the array.)

For example,

```
int main() {
   int x[3] = {8, 8, 7};
   int y[5] = {7, 2, 5, 1, 9};
   int z[2] = {2, 2};
   cout << indexFirstOdd(x, 3) << endl;       // prints 2
   cout << indexFirstOdd(y, 5) << endl;       // prints 0
   cout << indexFirstOdd(z, 2) << endl;       // prints -1
   return 0;
}
```

Answer:

Problem 109 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string fullName = "Freddy Next Door";
   int a2[2][3] = {{-2, 4, 3}, {-3, 4, 2}};
   int a[5] = {7, 6, 5, 9, 7};
   cout << middleDigit(19683) + 1   << endl;   // (a) prints: 7 as 6 + 1
   cout << random(a2, 2, 3) << endl;           // (b) prints random entry eg 4
   cout << initials(fullName) << endl;         // (c) prints: F.N.D.
   makePositive(a2[0][0]);                     // (d) make a2[0][0] positive
   cout << number7s(a, 5);                     // (e) prints 2: the number of 7s
   return 0;
}
```

(a) Title line for **middleDigit** as called at the line marked (a).
Answer:

(b) Title line for **random** as called at the line marked (b).
Answer:

(c) Title line for **initials** as called at the line marked (c).
Answer:

(d) Title line for **makePositive** as called at the line marked (d).
Answer:

(e) Title line for **number7s** as called at the line marked (e).
Answer:

Problem 110 Write the best **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   string fullName = "Freddy Next Door";
   int a2[2][3] = {{-2, 4, 3}, {-3, 4, 2}};
   int a[5] = {7, 6, 5, 9, 7};
   cout << firstLetter(fullName) << endl;      // (a) prints: F
   cout << sumFirstCol(a2, 2, 3) << endl;      // (b) prints: -5  (as -2 + - 3).
   cout << middleName(fullName) << endl;       // (c) prints: Next
   makeRandom(a2, 2, 3);                       // (d) reset the array with random entries
   cout << round(((double) a[0])/((double) a[1])); // (e) prints 1
                                         //  the nearest integer to the ratio.
   return 0;
}
```

(a) Title line for **firstLetter** as called at the line marked (a).
Answer:

(b) Title line for **sumFirstCol** as called at the line marked (b).
Answer:

(c) Title line for **middleName** as called at the line marked (c).
Answer:

(d) Title line for **makeRandom** as called at the line marked (d).
Answer:

(e) Title line for **round** as called at the line marked (e).
Answer:

Problem 111 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
    x = x + 1;
    y = y - 1;
    return y;
}

int main() {
    int x = 2, y = 7, z = 10;   string s = "007";
    cout << ((double) y) / x << endl;         // line (a)
    if (!((x > y) && (y > 5))) s = "008";
    cout << s << endl;                        // line (b)
    z %= y; cout << z << endl;                // line (c)
    cout << fun(z, y) << endl;                // line (d)
    fun(x, y); cout << y - x * 2 << endl;     // line (e)
}
```

(a) What is the output at line (a)?
Answer:

(b) What is the output at line (b)?
Answer:

(c) What is the output at line (c)?
Answer:

(d) What is the output at line (d)?
Answer:

(e) What is the output at line (e)?
Answer:

Problem 112 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int x, int &y) {
    x = x + 1;
    y = y - 1;
    return y;
}

int main() {
    int x = 3, y = 9, z = 10;   string s = "Yes";
    cout << ((double) x) / z << endl;          // line (a)
    if (!((x > y) || (y > 5))) s = "No";
    cout << s << endl;                         // line (b)
    z %= y; cout << z << endl;                 // line (c)
    cout << fun(z, y) << endl;                 // line (d)
    fun(x, y); cout << y - x % 2 << endl;      // line (e)
}
```

(a) What is the output at line (a)?
Answer:

(b) What is the output at line (b)?
Answer:

(c) What is the output at line (c)?
Answer:

(d) What is the output at line (d)?
Answer:

(e) What is the output at line (e)?
Answer:

Problem 113 Write a function called *removeLast0* that prints an integer parameter without its rightmost 0. If there is no 0, print the number itself. If the number is 0, print nothing.

For example, a program that uses the function *removeLast0* follows.

```
int main() {
   removeLast0(7070);         // prints 707
   cout << endl;
   removeLast0(7007);         // prints 707
   cout << endl;
   removeLast0(777);          // prints 777
   cout << endl;
   return 0;
}
```

Answer:

Problem 114 Write a function called *removeLast7* that removes the rightmost 7 from an integer parameter. If there is no 7, it makes no change.

For example, a program that uses the function *removeLast7* follows.

```
int main() {
   cout << removeLast7(777)  << endl;       // prints 77
   cout << removeLast7(1727) << endl;       // prints 172
   cout << removeLast7(1234) << endl;       // prints 1234
   return 0;
}
```

Answer:

Problem 115 Write a function called *largestGap* that returns the largest gap between two adjacent elements of an array.

For example, a program that uses the function *largestGap* follows, it prints 7 since the largest gap is between the 9 and the 2.

```
int main() {
   int x[] = {3, 1, 4, 1, 5, 9, 2, 6};
   cout << largestGap(x, 8) << endl;  // prints 7
   return 0;
}
```

Answer:

Problem 116 Write a function called *smallestProduct* that returns the smallest product formed by two adjacent elements of an array.

For example, a program that uses the function *smallestProduct* follows, it prints 3 since the smallest product is between the 3 and the 1.

```
int main() {
   int x[] = {3, 1, 4, 1, 5, 9, 2, 6};
   cout << smallestProduct(x, 8) << endl;  // prints 3
   return 0;
}
```

Answer:

Problem 117 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int x = 0, y = 1, z = 2;
   double b[3] = {1.9, 2.3, 3.0};

   x = larger(x + y, z);                // (a) sets x as the larger
   x = largest(x, y, y, z);             // (b) sets x as the largest
   printAll(b, x, y);                   // (c) print them all
   boost(x, y);                         // (d) increase x by the value of y
   boost(y, mystery(y, z));             // (e) boosts y by a mystery amount
   return 0;
}
```

(a) Title line for **larger** as called at the line marked (a).

Answer:

(b) Title line for **largest** as called at the line marked (b).

Answer:

(c) Title line for **printAll** as called at the line marked (c).

Answer:

(d) Title line for **boost** as called at the line marked (d).

Answer:

(e) Title line for **mystery** as called at the line marked (e).

Answer:

Problem 118 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int x = 0, y = 1, z = 2;
   double b[3] = {1.9, 2.3, 3.0};

   larger(x + y, z);                       // (a) prints the larger
   x = middle(x, y, y, z, z);              // (b) sets x as the middle value
   printAll(sqrt(b[1]), rand());           // (c) print them all
   swap(x, y);                             // (d) swap them
   cout << mystery(y, mystery(y, b[0]));   // (e) a mystery function
   return 0;
}
```

(a) Title line for **larger** as called at the line marked (a).

Answer:

(b) Title line for **middle** as called at the line marked (b).

Answer:

(c) Title line for **printAll** as called at the line marked (c).

Answer:

(d) Title line for **swap** as called at the line marked (d).

Answer:

(e) Title line for **mystery** as called at the line marked (e).

Answer:

Problem 119 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the sum of 3 things, here 6
   cout << sum3(1,3,c) << endl;
 // (b) Prints decimal form of fraction b/c, here 0.5
   cout << fraction(b, c) << endl;
 // (c) Fill array with random integers
   randomFill(a, 4);
 // (d) Print array backwards, with entries separated by spaces
   backPrint(a, 4);
 // (e) Print the first digit, assume argument is positive.  Here 1.
   firstDigit(19683);
   cout << endl;
   return 0;
}
```

(a) `int sum3(int x, int y, int z)`

Answer:

(b) `double fraction (int x, int y)`

Answer:

(c) `void randomFill(int x[], int cap)`

Answer:

(d) `void backPrint(int x[], int cap)`

Answer:

(e) `void firstDigit(int x)`

Answer:

Problem 120 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the average of 3 things, here 2.0
   cout << average3(1,3,c) << endl;
 // (b) Print the fraction b/c as a percentage, here 50.0%
   cout << percentage(b, c) << "%" << endl;
 // (c) Fill array with random positive single digit integers
   randomFill(a, 4);
 // (d) Print array, with entries separated by spaces
   print(a, 4);
 // (e) Print the second digit, assume argument is at least 10.  Here print 9.
   cout << secondDigit(19683) << endl;
   return 0;
}
```

(a) `double average3(int x, int y, int z)`

Answer:

(b) `double percentage(int x, int y)`

Answer:

(c) `void randomFill(int x[], int cap)`

Answer:

(d) `void print(int x[], int cap)`

Answer:

(e) `int secondDigit(int x)`

Answer:

Problem 121 Write a function called *gcd* that returns the greatest common divisor of two positive integers. For example, a program that uses the function *gcd* follows.

```
int main() {
   cout << gcd(10, 15) << endl;       // prints 5
   cout << gcd(11, 15) << endl;       // prints 1
   cout << gcd(0, 15) << endl;        // prints 15
   return 0;
}
```

Answer:

Problem 122 Write a function called *removeFirst* that removes the first digit of a positive integer and returns the result (or returns 0 if the integer has only one digit).

For example, a program that uses the function *removeFirst* follows.

```
int main() {
   cout << removeFirst(19683) << endl;    // prints 9683
   cout << removeFirst(11) << endl;       // prints 1
   cout << removeFirst(1) << endl;        // prints 0
   return 0;
}
```

Answer:

Problem 123 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter to enter 250 integers.
2. It computes the average of the 250 integers that the user supplies.
3. It prints out exactly those numbers entered by the user that differ from the average by no more than 10.

Answer:

Problem 124 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter to enter 250 integers.
2. It prints out exactly the negative numbers entered by the user in the reverse of their order of input.

Answer:

Problem 125 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[4] = {314, 315, 265, 358};
   int b = 1, c = 4;

   cout << max(a, 4) << endl;              // (a) prints: 358
   reverse(a, 4);                          // (b) prints: 358 265 315 314
   b = add(b, c);                          // (c) b becomes 5
   cout << difference(a[0], a[1]) << endl; // (d) prints: 1
   a[0] = sum(a[1], c);                    // (e) a[0] becomes 319
   return 0;
}
```

(a) Title line for **max** as called at the line marked (a).

Answer:

(b) Title line for **reverse** as called at the line marked (b).

Answer:

(c) Title line for **add** as called at the line marked (c).

Answer:

(d) Title line for **difference** as called at the line marked (d).

Answer:

(e) Title line for **sum** as called at the line marked (e).

Answer:

Problem 126 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[4] = {314, 315, 265, 358};
   int b = 1, c = 4, capacity = 4;

   swap(b, c);                             // (a) swaps values of b & c
   b = last(a, 4);                         // (b) b becomes 358
   c = add(a[1], a[0]);                    // (c) c becomes 629
   cout << max(a[1], 1) << endl;           // (d) prints: 315
   cout << max(a, capacity, 700) << endl;  // (e) prints 700
   return 0;
}
```

(a) Title line for **swap** as called at the line marked (a).

Answer:

(b) Title line for **last** as called at the line marked (b).

Answer:

(c) Title line for **add** as called at the line marked (c).

Answer:

(d) Title line for **max** as called at the line marked (d).

Answer:

(e) Title line for **max** as called at the line marked (e).

Answer:

Problem 127 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[4] = {314, 315, 265, 358};
   int b = 1, c = 4;

   cout << max(4, a) << endl;               // (a) prints: 358
   reverse(a, 4);                           // (b) a becomes 358,265,315,314
   b = add(b, b, c);                        // (c) b becomes 6
   cout << difference(a[1], 300) << endl;  // (d) prints: 15
   addOn(a[1], c);                          // (e) a[1] changes to 319
   return 0;
}
```

(a) Title line for **max** as called at the line marked (a).

Answer:

(b) Title line for **reverse** as called at the line marked (b).

Answer:

(c) Title line for **add** as called at the line marked (c).

Answer:

(d) Title line for **difference** as called at the line marked (d).

Answer:

(e) Title line for **addOn** as called at the line marked (e).

Answer:

Problem 128 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[4] = {314, 315, 265, 358};
   int b = 1, c = 4, capacity = 4;

   swap(a[3], c);                           // (a) swaps values of a[3] & c
   b = first(a);                            // (b) b becomes 314
   a[3] = add(a[1], a[0]);                  // (c) a[3] becomes 629
   cout << min(a, capacity) << endl;        // (d) prints: 265
   printMin(a, capacity);                   // (e) prints: 265
   return 0;
}
```

(a) Title line for **swap** as called at the line marked (a).

Answer:

(b) Title line for **first** as called at the line marked (b).

Answer:

(c) Title line for **add** as called at the line marked (c).

Answer:

(d) Title line for **min** as called at the line marked (d).

Answer:

(e) Title line for **printMin** as called at the line marked (e).

Answer:

Problem 129 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[2][2] = {{314, 315}, {265, 358}};
   int b = 1, c = 4;

   cout << max(a, 2, 2) << endl;                 // (a) prints: 358
   reverse(a, 2, 2);                             // (b) prints: 358 265 315 314
   b = add(b, c);                                // (c) b becomes 5
   cout << difference(a[0][0], a[0][1]) << endl; // (d) prints: 1
   a[0][0] = sum(a[0][1], c);                    // (e) a[0][0] becomes 319
   return 0;
}
```

(a) Title line for **max** as called at the line marked (a).

Answer:

(b) Title line for **reverse** as called at the line marked (b).

Answer:

(c) Title line for **add** as called at the line marked (c).

Answer:

(d) Title line for **difference** as called at the line marked (d).

Answer:

(e) Title line for **sum** as called at the line marked (e).

Answer:

Problem 130 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[2][2] = {{314, 315}, {265, 358}};
   int b = 1, c = 4, rows = 2, cols = 2;

   swap(b, c);                                // (a) swaps values of b & c
   b = last(a, rows, cols);                   // (b) b becomes 358
   c = add(a[0][1], a[0][0]);                 // (c) c becomes 629
   cout << max(a[0][1], 1) << endl;           // (d) prints: 314
   cout << max(a, rows, cols, 700) << endl;   // (e) prints 700
   return 0;
}
```

(a) Title line for **swap** as called at the line marked (a).

Answer:

(b) Title line for **last** as called at the line marked (b).

Answer:

(c) Title line for **add** as called at the line marked (c).

Answer:

(d) Title line for **max** as called at the line marked (d).

Answer:

(e) Title line for **max** as called at the line marked (e).

Answer:

Problem 131 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[2][2] = {{314, 315}, {265, 358}};
   int b = 1, c = 4;

   cout << max(2, 2, a) << endl;              // (a) prints: 358
   reverse(a, 2, 2);                          // (b) a becomes 358,265,315,314
   b = add(b, b, c);                          // (c) b becomes 6
   cout << difference(a[0][1], 300) << endl;  // (d) prints: 15
   addOn(a[0][1], c);                         // (e) a[0][1] changes to 319
   return 0;
}
```

(a) Title line for **max** as called at the line marked (a).

Answer:

(b) Title line for **reverse** as called at the line marked (b).

Answer:

(c) Title line for **add** as called at the line marked (c).

Answer:

(d) Title line for **difference** as called at the line marked (d).

Answer:

(e) Title line for **addOn** as called at the line marked (e).

Answer:

Problem 132 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[2][2] = {{314, 315}, {265, 358}};
   int b = 1, c = 4, row = 2, col = 2;

   swap(a[1][1], c);                          // (a) swaps values of a[1][1] & c
   b = first(a);                              // (b) b becomes 314
   a[1][1] = add(a[0][1], a[0][0]);           // (c) a[1][1] becomes 629
   cout << min(a, row, col) << endl;          // (d) prints: 265
   printMin(a, row, col);                     // (e) prints: 265
   return 0;
}
```

(a) Title line for **swap** as called at the line marked (a).

Answer:

(b) Title line for **first** as called at the line marked (b).

Answer:

(c) Title line for **add** as called at the line marked (c).

Answer:

(d) Title line for **min** as called at the line marked (d).

Answer:

(e) Title line for **printMin** as called at the line marked (e).

Answer:

Problem 133 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the difference (ignoring sign), here 1
   cout << absoluteDifference(7,8) << endl;
 // (b) Prints random integer in range from b to c, assume b < c
   cout << random(b, c) << endl;
 // (c) Print square root of sum of squares of arguments, here 5.0
   cout << hyp(3, 4) << endl;
 // (d) Print array backwards, here 1413
   backPrint(a, 4);
 // (e) Print the last digit, assume argument is positive.  Here 3.
   lastDigit(19683);
   return 0;
}
```

(a) `int absoluteDifference(int x, int y)`

Answer:

(b) `int random(int x, int y)`

Answer:

(c) `double hyp(int x, int y)`

Answer:

(d) `void backPrint(int x[], int cap)`

Answer:

(e) `void lastDigit(int x)`

Answer:

Problem 134 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the max, here 8
   cout << max(7,8) << endl;
 // (b) Swaps values
   swap(b, c);
 // (c) Print ratio, here 0.75
   cout << ratio(3, 4) << endl;
 // (d) Print number of even entries, here 1
   cout << countEven(a, 4) << endl;
 // (e) Print the first digit, assume argument is positive.  Here 1.
   firstDigit(19683);
   return 0;
}
```

(a) `int max(int x, int y)`
Answer:

(b) `void swap(int &x, int &y)`
Answer:

(c) `double ratio(int x, int y)`
Answer:

(d) `int countEven(int x[], int cap)`
Answer:

(e) `void firstDigit(int x)`
Answer:

Problem 135 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Prints the absolute value (ignore sign), here 7
   cout << absolute(-7) << endl;
 // (b) Prints a random id number with the given length, here 007 may be printed
   random(3);
 // (c) Prints the ratio as a percentage, here 12.5% for 1/8
   cout << percentage(1, 8) << "%" << endl;
 // (d) Print every second entry of the array here 34
   skipPrint(a, 4);
 // (e) Print the last two digit, assume argument is at least 10.  Here 83.
   lastTwoDigits(19683);
   return 0;
}
```

(a) `int absolute(int x)`
Answer:

(b) `void random(int x)`
Answer:

(c) `double percentage(int x, int y)`
Answer:

(d) `void skipPrint(int x[], int cap)`
Answer:

(e) `void lastTwoDigits(int x)`
Answer:

Problem 136 Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```
int main() {
   int b = 1, c = 2, a[4] = {3, 1, 4, 1};
 // (a) Print the number of odd arguments, here 1
   cout << numberOdd(7,8) << endl;
 // (b) Reorder arguments so that they increase, here swap them
   sort(c, b);
 // (c) Print closest integer here 4
   cout << closest(3.75) << endl;
 // (d) Print maximum entry, here 4
   cout << max(a, 4) << endl;
 // (e) Print the first digit, assume argument is positive.  Here 1.
   cout << firstDigit(19683) << endl;
   return 0;
}
```

(a) `int numberOdd(int x, int y)`

Answer:

(b) `void sort(int &x, int &y)`

Answer:

(c) `int closest(double x)`

Answer:

(d) `int max(int x[], int cap)`

Answer:

(e) `int firstDigit(int x)`

Answer:

Problem 137 Write a function called *numEven* that returns the number of even digits in a positive integer parameter.

For example, a program that uses the function *numEven* follows.

```
int main() {
   cout << numEven(23) << endl;       // prints 1
   cout << numEven(1212) << endl;     // prints 2
   cout << numEven(777) << endl;      // prints 0
   return 0;
}
```

Answer:

Problem 138 Write a function called *lastEven* that returns the last even digit in a positive integer parameter. It should return 0 if there are no even digits.

For example, a program that uses the function *lastEven* follows.

```
int main() {
   cout << lastEven(23) << endl;       // prints 2
   cout << lastEven(1214) << endl;     // prints 4
   cout << lastEven(777) << endl;      // prints 0
   return 0;
}
```

Answer:

Problem 139 Write a function called *sumEven* that returns the sum of the even digits in a positive integer parameter.

For example, a program that uses the function *sumEven* follows.

```
int main() {
   cout << sumEven(23) << endl;       // prints 2
   cout << sumEven(1212) << endl;     // prints 4
   cout << sumEven(777) << endl;      // prints 0, because there are none
   return 0;
}
```

Answer:

Problem 140 Write a function called *lastOdd* that returns the last odd digit in a positive integer parameter. It should return 0 if there are no odd digits.

For example, a program that uses the function *lastOdd* follows.

```
int main() {
   cout << lastOdd(23) << endl;       // prints 3
   cout << lastOdd(1254) << endl;     // prints 5
   cout << lastOdd(666) << endl;      // prints 0
   return 0;
}
```

Answer:

Problem 141 Write a function called *firstEven* that returns the first even digit in a positive integer parameter. It should return -1 if there are no even digits.

For example, a program that uses the function *firstEven* follows.

```
int main() {
   cout << firstEven(23) << endl;       // prints 2
   cout << firstEven(1416) << endl;     // prints 4
   cout << firstEven(777) << endl;      // prints -1
   return 0;
}
```

Answer:

Problem 142 Write a function called *evenLessOdd* that returns the sum of the even valued digit minus the sum of the odd valued digits in a positive integer parameter.

For example, a program that uses the function *evenLessOdd* follows.

```
int main() {
   cout << evenLessOdd(43) << endl;       // prints 1
   cout << evenLessOdd(9876) << endl;     // prints -2
   cout << evenLessOdd(777) << endl;      // prints -21
   return 0;
}
```

Answer:

Problem 143 Write a function called *firstOdd* that returns the first odd digit in a positive integer parameter. It should return -1 if there are no odd digits.

For example, a program that uses the function *firstOdd* follows.

```
int main() {
   cout << firstOdd(21) << endl;       // prints 1
   cout << firstOdd(3456) << endl;     // prints 3
   cout << firstOdd(666) << endl;      // prints -1
   return 0;
}
```

Answer:

Problem 144 Write a function called *oddLessEven* that returns the sum of the odd valued digits minus the sum of the even valued digits in a positive integer parameter.

For example, a program that uses the function *oddLessEven* follows.

```
int main() {
   cout << oddLessEven(23) << endl;       // prints 1
   cout << oddLessEven(1234) << endl;     // prints -2
   cout << oddLessEven(777) << endl;      // prints 21
   return 0;
}
```

Answer:

Problem 145 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][3], int x, int y) {
    if (a[x][y] % 2 == 0) cout << a[x][y] << endl;
    a[x][y]++;
    return a[x][y];
}

int main() {
    int x[2][3] = {{1,2,3}, {3,4,5}};
    cout << x[1][1] << endl;                              // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;   // line (b)
    cout << x[x[0][0]][x[0][1]] << endl;                  // line (c)
    up(x,1,1);                                            // line (d)
    cout << up(x,1,2) << endl;                            // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 146 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][3], int x, int y) {
    if (y < 2) return a[x][y+1];
    cout << a[x][y] << endl;
    return a[x][y];
}

int main() {
    int x[2][3] = {{3,2,1}, {0,3,6}}, a = 0;
    cout << x[a][a] << endl;                                   // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][2 - i] << endl;   // line (b)
    cout << x[x[x[0][2]][0]][0] << endl;                       // line (c)
    up(x,1,1);                                                 // line (d)
    cout << up(x,1,2) << endl;                                 // line (e)
}
```

(a) What is the output at line (a)?
Answer:

(b) What is the output at line (b)?
Answer:

(c) What is the output at line (c)?
Answer:

(d) What is the output at line (d)?
Answer:

(e) What is the output at line (e)?
Answer:

Problem 147 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][3], int x, int y) {
    if (a[x][y] % 2 == 1) cout << a[x][y] << endl;
    a[x][y]++;
    return a[x][y];
}

int main() {
    int x[2][3] = {{0,1,2}, {4,5,6}}, a = 0;
    cout << x[1][1] << endl;                                // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;    // line (b)
    cout << x[x[0][0]][x[0][1]] << endl;                    // line (c)
    cout << up(x,1,1) << endl;                              // line (d)
    up(x,1,2);                                              // line (e)
}
```

(a) What is the output at line (a)?
Answer:

(b) What is the output at line (b)?
Answer:

(c) What is the output at line (c)?
Answer:

(d) What is the output at line (d)?
Answer:

(e) What is the output at line (e)?
Answer:

Problem 148 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][3], int x, int y) {
    if (y < 2) return a[1-x][y+1];
    cout << a[x][y] << endl;
    return a[x][y];
}

int main() {
    int x[2][3] = {{2,1,0}, {0,4,8}}, a = 0;
    cout << x[a][2*a] << endl;                              // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;    // line (b)
    cout << x[0][x[x[0][1]][0]]<< endl;                     // line (c)
    up(x,1,2);                                              // line (d)
    cout << up(x,1,1) << endl;                              // line (e)
}
```

(a) What is the output at line (a)?
Answer:

(b) What is the output at line (b)?
Answer:

(c) What is the output at line (c)?
Answer:

(d) What is the output at line (d)?
Answer:

(e) What is the output at line (e)?
Answer:

Problem 149 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][2], int x, int y) {
    if (a[x][y] % 2 == 0) cout << a[x][y] << endl;
    a[x][y]++;
    return a[x][y];
}

int main() {
    int x[3][2] = {{1,2}, {3,3}, {4,5}};
    cout << x[1][1] << endl;                             // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;   // line (b)
    cout << x[x[0][1]][x[0][0]] << endl;                 // line (c)
    up(x,1,1);                                           // line (d)
    cout << up(x,2,1) << endl;                           // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 150 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][2], int x, int y) {
    if (y < 1) return a[x][y+1];
    cout << a[x][y] << endl;
    return a[x][y];
}

int main() {
    int x[3][2] = {{3,2},{4,5},{0,1}}, a = 0;
    cout << x[a][a] << endl;                                 // line (a)
    for (int i = 0; i < 2; i++) cout << x[2 - i][i] << endl;   // line (b)
    cout << x[x[x[2][0]][0]][0] << endl;                     // line (c)
    up(x,1,1);                                               // line (d)
    cout << up(x,2,1) << endl;                               // line (e)
}
```

(a) What is the output at line (a)?
Answer:

(b) What is the output at line (b)?
Answer:

(c) What is the output at line (c)?
Answer:

(d) What is the output at line (d)?
Answer:

(e) What is the output at line (e)?
Answer:

Problem 151 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][2], int x, int y) {
    if (a[x][y] % 2 == 0) cout << a[x][y] << endl;
    a[x][y]++;
    return a[x][y];
}

int main() {
    int x[3][2] = {{0,1}, {3,4}, {5,7}};
    cout << x[1][1] << endl;                             // line (a)
    for (int i = 0; i < 2; i++) cout << x[i][i] << endl;   // line (b)
    cout << x[x[0][1]][x[0][0]] << endl;                 // line (c)
    up(x,1,1);                                           // line (d)
    cout << up(x,2,1) << endl;                           // line (e)
}
```

(a) What is the output at line (a)?
Answer:

(b) What is the output at line (b)?
Answer:

(c) What is the output at line (c)?
Answer:

(d) What is the output at line (d)?
Answer:

(e) What is the output at line (e)?
Answer:

Problem 152 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int up(int a[][2], int x, int y) {
    if (y < 1) return a[x][y+1];
    cout << a[x][y] << endl;
    return a[x][y];
}

int main() {
    int x[3][2] = {{2,3},{0,4},{1,5}}, a = 0;
    cout << x[a][a] << endl;                                   // line (a)
    for (int i = 0; i < 2; i++) cout << x[2 - i][i] << endl;   // line (b)
    cout << x[x[x[2][0]][0]][0] << endl;                       // line (c)
    up(x,1,1);                                                 // line (d)
    cout << up(x,2,1) << endl;                                 // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 153 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   int a[4] = {314, 159, 265, 358};
   cout << sqrt("FFrreedd") << endl;      // prints: Fred
   cout << firstLetter("Freddy") << endl; // prints: F
   sort(a, 4);                            // prints: 159 265 314 358
   oddElements(a, 4);                     // prints: odd: 159 265
   a[0] = sum(a[1], a[2]);                // adds elements
   return 0;
}
```

(a) Title line for **sqrt**.

Answer:

(b) Title line for **firstLetter**.

Answer:

(c) Title line for **sort**.

Answer:

(d) Title line for **oddElements**.

Answer:

(e) Title line for **sum**.

Answer:

Problem 154 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int &y) {
   if (y <= 0) return x;
   x = x + 2;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 5, y = -1;
  cout << fun(x, y) << endl;   // line a
  fun(y, x);                   // line b
  fun(x, y);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, y) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

(b) line b:

(c) line c:

(d) line d:

(e) line e:

Problem 155 Write a function called *addThrees* that inserts a 3 after each digit of a positive integer parameter. For example, a program that uses the function *addThrees* follows.

```
int main() {
   cout << addThrees(3)  << endl;       // prints 33
   cout << addThrees(1313) << endl;     // prints 13331333
   cout << addThrees(777) << endl;      // prints 737373
   return 0;
}
```

Answer:

Problem 156 Write a C++ function called *halfs* that divides each element of a 2-dimensional array (with two columns) by 2.

It should be possible to use your function in the following program.

```
main() {
   double data[2][2] = {{1, 2}, {3, 4}};
   halfs (data, 2, 2);
   for (int i = 0; i < 2; i++)
     cout << data[1][i] << " ";   // prints 1.5 2.0
}
```

Answer:

Problem 157 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   int a[4] = {314, 159, 265, 358};
   sqrt("FFrreedd");                          // prints: Fred
   firstLetter("Freddy");                     // prints: F
   sort(a, 4);                                // prints: 159 265 314 358
   cout << oddElements(a, 4);                 // prints: odd: 159 265
   swap(a[1], a[2]);                          // swaps elements
   return 0;
}
```

(a) Title line for **sqrt**.

Answer:

(b) Title line for **firstLetter**.

Answer:

(c) Title line for **sort**.

Answer:

(d) Title line for **oddElements**.

Answer:

(e) Title line for **swap**.

Answer:

Problem 158 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int &y) {
   if (y <= 0) return x;
   x = x + 2;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 4, y = 0;
  cout << fun(x, y) << endl;   // line a
  fun(y, x);                   // line b
  fun(x, y);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, y) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

(b) line b:

(c) line c:

(d) line d:

(e) line e:

Problem 159 Write a function called *addThrees* that inserts a 3 before each digit of a positive integer parameter. For example, a program that uses the function *addThrees* follows.

```
int main() {
   cout << addThrees(3)  << endl;       // prints 33
   cout << addThrees(1313) << endl;     // prints 31333133
   cout << addThrees(777) << endl;      // prints 373737
   return 0;
}
```

Answer:

Problem 160 Write a C++ function called *roots* that replaces each element of an array by its root. It should be possible to use your function in the following program.

```
main() {
   double data[3] = {1.0, 4.0, 9.0};
   roots (data, 3);
   for (int i = 0; i < 3; i++)
     cout << data[i] << " ";   // prints 1 2 3
}
```

Answer:

Problem 161 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   int a[4] = {314, 159, 265, 358};
   cout << firstLetter("Freddy") << endl; // prints: F
   cout << sqrt("FFrreedd") << endl;      // prints: Fred
   oddElements(a, 4);                     // prints: odd: 159 265
   sort(a, 4);                            // prints: 159 265 314 358
   a[0] = sum(a[1], a[2]);                // adds elements
   return 0;
}
```

(a) Title line for **firstLetter**.
Answer:

(b) Title line for **sqrt**.
Answer:

(c) Title line for **oddElements**.
Answer:

(d) Title line for **sort**.
Answer:

(e) Title line for **sum**.
Answer:

Problem 162 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int &y) {
   if (y <= 0) return x;
   x = x + 2;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 3, y = -1;
  cout << fun(x, y) << endl;   // line a
  fun(y, x);                   // line b
  fun(x, y);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, y) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

(b) line b:

(c) line c:

(d) line d:

(e) line e:

Problem 163 Write a function called *addTwos* that inserts a 2 after each digit of a positive integer parameter. For example, a program that uses the function *addTwos* follows.

```
int main() {
   cout << addTwos(3)  << endl;       // prints 32
   cout << addTwos(1212) << endl;     // prints 12221222
   cout << addTwos(777) << endl;      // prints 727272
   return 0;
}
```

Answer:

Problem 164 Write a C++ function called *squares* that replaces each element of a 2-dimensional array (with two columns) by its square.

It should be possible to use your function in the following program.

```
main() {
   int data[2][2] = {{1, 2}, {3, 4}};
   squares (data, 2, 2);
   for (int i = 0; i < 2; i++)
     cout << data[1][i] << " ";   // prints 9 16
}
```

Answer:

Problem 165 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   int a[4] = {314, 159, 265, 358};
   firstLetter("Freddy");                 // prints: F
   sqrt("FFrreedd");                      // prints: Fred
   cout << oddElements(a, 4);             // prints: odd: 159 265
   sort(a, 4);                            // prints: 159 265 314 358
   swap(a[1], a[2]);                      // swaps elements
   return 0;
}
```

(a) Title line for **firstLetter**.

Answer:

(b) Title line for **sqrt**.

Answer:

(c) Title line for **oddElements**.

Answer:

(d) Title line for **sort**.

Answer:

(e) Title line for **swap**.

Answer:

Problem 166 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int &y) {
   if (y <= 0) return x;
   x = x + 2;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 2, y = 0;
  cout << fun(x, y) << endl;   // line a
  fun(y, x);                   // line b
  fun(x, y);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, y) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

(b) line b:

(c) line c:

(d) line d:

(e) line e:

Problem 167 Write a function called *addTwos* that inserts a 2 before each digit of a positive integer parameter. For example, a program that uses the function *addTwos* follows.

```
int main() {
   cout << addTwos(3)  << endl;       // prints 23
   cout << addTwos(1212) << endl;     // prints 21222122
   cout << addTwos(777) << endl;      // prints 272727
   return 0;
}
```

Answer:

Problem 168 Write a C++ function called *cubes* that replaces each element of an array by its cube. It should be possible to use your function in the following program.

```
main() {
   int data[3] = {1, 2, 3};
   cubes (data, 3);
   for (int i = 0; i < 3; i++)
     cout << data[i] << " ";   // prints 1 8 27
}
```

Answer:

Problem 169 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   string a[4] = {"Freddy", "Max", "Kelly", "Jack"};
   undouble(11223344);                   // prints: 1234
   firstDigit(65536);                    // prints: Six
   printSorted(a, 4);                    // prints: Freddy Jack Kelly Max
   cout << join(a[1], a[3]) << endl;     // prints: MaxJack
   randomWords(a, 4);                    // assigns new random values to array
   return 0;
}
```

(a) Title line for **undouble**.
Answer:

(b) Title line for **firstDigit**.
Answer:

(c) Title line for **printSorted**.
Answer:

(d) Title line for **join**.
Answer:

(e) Title line for **randomWords**.
Answer:

Problem 170 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
   if (y <= 0) return x;
   x = x + 1;
   y = y + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 5, y = -1;
  cout << fun(x, y) << endl;   // line a
  fun(x, 1);                   // line b
  fun(y, 1);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, 2) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

(b) line b:

(c) line c:

(d) line d:

(e) line e:

Problem 171 Write a function called *killTwos* that deletes all digits that are multiples of 2 from a positive integer parameter.

For example, a program that uses the function *killTwos* follows.

```
int main() {
   cout << killTwos(11) << endl;       // prints 11
   cout << killTwos(1212) << endl;     // prints 11
   cout << killTwos(2400) << endl;     // prints 0, because no digits are left
   return 0;
}
```

Answer:

Problem 172 Write a C++ function called *numOdd* that returns the number of odd elements in a 2-dimensional array (with 4 columns).

It should be possible to use your function in the following program. (The output from this program is 2 because only the two 11s are odd).

```
main() {
   int data[2][4] = {{11, 12, 14, 0}, {32, 12, 132, 11}};
   int x;
   x = numOdd (data, 2, 4);
     //   data is the 2-d array, 2 and 4 are its capacities
   cout << "The number of odds is: " << x  << endl;
}
```

Answer:

Problem 173 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   string a[4] = {"Freddy", "Max", "Kelly", "Jack"};
   cout << undouble(11223344);            // prints: 1234
   cout << firstDigit(65536) << endl;     // prints: Six
   sort(a, 4);                            // prints: Freddy Jack Kelly Max
   cout << halfString(a[0]) << endl;      // prints: Fre
   a[1] = randomWord();                   // assigns a random value
   return 0;
}
```

(a) Title line for **undouble**.

Answer:

(b) Title line for **firstDigit**.

Answer:

(c) Title line for **sort**.

Answer:

(d) Title line for **halfString**.

Answer:

(e) Title line for **randomWord**.

Answer:

Problem 174 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
   if (y <= 0) return x;
   x = x + 1;
   y = y + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 4, y = 0;
  cout << fun(x, y) << endl;   // line a
  fun(x, 1);                   // line b
  fun(y, 1);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, 2) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

(b) line b:

(c) line c:

(d) line d:

(e) line e:

Problem 175 Write a function called *twos* that deletes all digits that are not multiples of 2 from a positive integer parameter.

For example, a program that uses the function *twos* follows.

```
int main() {
   cout << twos(23) << endl;       // prints 2
   cout << twos(1212) << endl;     // prints 22
   cout << twos(777) << endl;      // prints 0, because nothing is left
   return 0;
}
```

Answer:

Problem 176 Write a C++ function called *range* that returns the difference between the largest and smallest elements in a 2-dimensional array (with 4 columns).

It should be possible to use your function in the following program. (The output from this program is 10 because the difference between the largest element 13 and the smallest element 3 is $13 - 3 = 10$).

```
main() {
   int data[2][4] = {{11, 12, 11, 5}, {6, 3, 12, 13}};
   int x;
   x = range (data, 2, 4);
     //   data is the 2-d array, 2 and 4 are its capacities
   cout << "The range is: " << x  << endl;
}
```

Answer:

Problem 177 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   string a[4] = {"Freddy", "Max", "Kelly", "Jack"};
   firstDigit(65536);                     // prints: Six
   undouble(11223344);                    // prints: 1234
   cout << join(a[1], a[3]) << endl;      // prints: MaxJack
   printSorted(a, 4);                     // prints: Freddy Jack Kelly Max
   randomWords(a, 4);                     // assigns new random values to array
   return 0;
}
```

(a) Title line for **firstDigit**.

Answer:

(b) Title line for **undouble**.

Answer:

(c) Title line for **join**.

Answer:

(d) Title line for **printSorted**.

Answer:

(e) Title line for **randomWords**.

Answer:

Problem 178 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
   if (y <= 0) return x;
   x = x + 1;
   y = y + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 3, y = -1;
  cout << fun(x, y) << endl;   // line a
  fun(x, 1);                   // line b
  fun(y, 1);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, 2) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

(b) line b:

(c) line c:

(d) line d:

(e) line e:

Problem 179 Write a function called *killTwos* that deletes all digits that are equal to 2 from a positive integer parameter.

For example, a program that uses the function *killTwos* follows.

```
int main() {
   cout << killTwos(11) << endl;       // prints 11
   cout << killTwos(1212) << endl;     // prints 11
   cout << killTwos(222) << endl;      // prints 0, because no digits are left
   return 0;
}
```

Answer:

Problem 180 Write a C++ function called *numEven* that returns the number of even elements in a 2-dimensional array (with 3 columns).

It should be possible to use your function in the following program. (The output from this program is 2 because only the two 12s are even).

```
main() {
   int data[2][3] = {{11, 12, 11}, {3, 12, 13}};
   int x;
   x = numEven (data, 2, 3);
     //   data is the 2-d array, 2 and 3 are its capacities
   cout << "The number of evens is: " << x  << endl;
}
```

Answer:

Problem 181 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.** Your title lines must allow for any indicated types of output.

```
int main() {
   string a[4] = {"Freddy", "Max", "Kelly", "Jack"};
   cout << firstDigit(65536) << endl;     // prints: Six
   cout << undouble(11223344);            // prints: 1234
   cout << halfString(a[0]) << endl;      // prints: Fre
   sort(a, 4);                            // prints: Freddy Jack Kelly Max
   a[1] = randomWord();                   // assigns a random value
   return 0;
}
```

(a) Title line for **firstDigit**.

Answer:

(b) Title line for **undouble**.

Answer:

(c) Title line for **halfString**.

Answer:

(d) Title line for **sort**.

Answer:

(e) Title line for **randomWord**.

Answer:

Problem 182 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
   if (y <= 0) return x;
   x = x + 1;
   y = y + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 2, y = 0;
  cout << fun(x, y) << endl;   // line a
  fun(x, 1);                   // line b
  fun(y, 1);                   // line c
  fun(y, x);                   // line d
  cout << fun(x, 2) << endl;   // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

(b) line b:

(c) line c:

(d) line d:

(e) line e:

Problem 183 Write a function called *twos* that deletes all digits that are not equal to 2 from a positive integer parameter.

For example, a program that uses the function *twos* follows.

```
int main() {
   cout << twos(23) << endl;       // prints 2
   cout << twos(1212) << endl;     // prints 22
   cout << twos(777) << endl;      // prints 0, because nothing is left
   return 0;
}
```

Answer:

Problem 184 Write a C++ function called *range* that returns the difference between the largest and smallest elements in a 2-dimensional array (with 3 columns).

It should be possible to use your function in the following program. (The output from this program is 10 because the difference between the largest element 13 and the smallest element 3 is $13 - 3 = 10$).

```
main() {
   int data[2][3] = {{11, 12, 11}, {3, 12, 13}};
   int x;
   x = range (data, 2, 3);
     //   data is the 2-d array, 2 and 3 are its capacities
   cout << "The range is: " << x  << endl;
}
```

Answer:

Problem 185 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[5] = {3,1,4,1,5};
   int x[2][3] = {{0,1,3},{2,4,5}};
   string s= "Hello";
   string t;

   cout << average(a, 5) << endl;           // prints the average:  2.8
   t = reverse(s);  cout << t << endl;      // prints:  olleH
   reverseRows(x, 2, 3);                    // prints:  2 4 5, 0 1 3
   if (hasRepeat(a, 5)) cout << "Has repeat" << endl;
                                            // prints:  Has repeat
   t = entries(a, 5); cout << t << endl;    // prints:  3,1,4,1,5
   return 0;
}
```

(a) Title line for **average**

Answer:

(b) Title line for **reverse**

Answer:

(c) Title line for **reverseRows**

Answer:

(d) Title line for **hasRepeat**

Answer:

(e) Title line for **entries**

Answer:

Problem 186 Consider the following C++ program.

```
#include <iostream>
using namespace std;

char f(string s, int n) {
   if (n >= s.length()) return 'A';
   return s[n];
}

int mystery (int x) {
   if (x > 5) return 0;
   cout << -x;
   return x;
}

int main () {
   cout << f("Hello", 20) << endl;     //line A
   cout << f("Hello", 1)  << endl;     //line B
   cout << mystery(19683) << endl;     //line C
   cout << mystery(2) << endl;         //line D
   mystery(-5);                        //line E
   cout << endl;
   return 0;
}
```

(a) What is the output at line A?

Answer:

(b) What is the output at line B?

Answer:

(c) What is the output at line C?

Answer:

(d) What is the output at line D?

Answer:

(e) What is the output at line E?

Answer:

Problem 187 Write a function called *extraOne* that places an initial 1 at the start of an integer parameter. (Assume that the input parameter is not negative.)

For example, a program that uses the function *extraOne* follows.

```
int main() {
   int x = extraOne(729);
   cout << x << endl;    // prints   1729
   return 0;
}
```

Answer:

Problem 188 Write a function called *dropDimension* that copies the entries from a 2-dimensional array row by row as the entries of a 1-dimensional array. Assume that the 1-dimensional array has more than enough capacity for these entries. (The function should use capacities of the 2-dimensional array but not the 1-dimensional array as input parameters.)

For example, a program that uses the function follows.

```
int main() {
   int x[100];
   int y[2][3] = {{3,1,4}, {1,5,9}};
   int yrows = 2, ycols = 3;
   dropDimension(y, yrows, ycols, x);
   for (int i = 0; i <= 5; i++) cout << x[i];
     // 314159   is printed
   cout << endl;
   return 0;
}
```

Answer:

Problem 189 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[5] = {3,1,4,1,5};
   int x[2][3] = {{0,1,3},{2,4,8}};
   string s= "Hello";
   string t;

   cout << average(x, 2, 3) << endl;        // prints the average:  3.0
   t = doubleIt(s);  cout << t << endl;     // prints:  HelloHello
   reverseCols(x, 2, 3);                    // prints:  3 0 1, 8 4 2
   if (isPositive(a[0])) cout << "Positive" << endl;
                                            // prints:  Positive
   cout << midEntry(a, 5) << endl;          // prints:  4
   return 0;
}
```

(a) Title line for **average**
Answer:

(b) Title line for **doubleIt**
Answer:

(c) Title line for **reverseCols**
Answer:

(d) Title line for **isPositive**
Answer:

(e) Title line for **midEntry**
Answer:

Problem 190 Consider the following C++ program.

```
#include <iostream>
using namespace std;

string f(string s, int n) {
   if (n >= s.length()) return "XYZ";
   return s.substr(n);
}

int mystery (int x) {
   if (x > 5) return 0;
   return x;
}

int main () {
   cout << mystery(19683) << endl;     //line A
   cout << mystery(2) << endl;         //line B
   cout << f("Hello", 20) << endl;     //line C
   cout << f("Hello", 1)  << endl;     //line D
   mystery(-5);                        //line E
   return 0;
}
```

(a) What is the output at line A?

Answer:

(b) What is the output at line B?

Answer:

(c) What is the output at line C?

Answer:

(d) What is the output at line D?

Answer:

(e) What is the output at line E?

Answer:

Problem 191 Write a function called *doubleEight* that places an extra digit 8 after the last 8 in an integer parameter. If there is no 8 present, nothing is done. (Assume that the input parameter is not negative.)

For example, a program that uses the function *doubleEight* follows.

```
int main() {
   int x = doubleEight(19683);
   cout << x << endl;                     // prints   196883
   cout << doubleEight(271828) << endl;   // prints   2718288
   cout << doubleEight(314159) << endl;   // prints   314159
   return 0;
}
```

Answer:

Problem 192 Write a function called *dropDimension* that copies the entries from a 2-dimensional array column by column as the entries of a 1-dimensional array. Assume that the 1-dimensional array has more than enough capacity for these entries. (The function should use capacities of the 2-dimensional array but not the 1-dimensional array as input parameters.)

For example, a program that uses the function follows.

```
int main() {
   int x[100];
   int y[2][3] = {{3,4,5}, {1,1,9}};
   int yrows = 2, ycols = 3;
   dropDimension(y, yrows, ycols, x);
   for (int i = 0; i <= 5; i++) cout << x[i];
     // 314159   is printed
   cout << endl;
   return 0;
}
```

Answer:

Problem 193 Write a function called *extraTwo* that inserts an extra digit 2 as the second digit of an integer parameter. (Assume that the input parameter is positive.)

For example, a program that uses the function *extraTwo* follows.

```
int main() {
   int x = extraTwo(79);
   cout << x << endl;             // prints   729
   cout << extraTwo(1) << endl;   // prints   12
   return 0;
}
```

Answer:

Problem 194 Write a function called *fill2D* that fills the entries of a 2-dimensional array column by column from the entries of a 1-dimensional array. Assume that the 1-dimensional array has more than enough capacity for these entries. (The function should use capacities of the 2-dimensional array but not the 1-dimensional array as input parameters.)

For example, a program that uses the function follows.

```
int main() {
   int x[11] = {3,1,4,1,5,9,2,6,5,3,5};
   int y[2][3];
   int yrows = 2, ycols = 3;
   fill2D(y, yrows, ycols, x);
   for (int i = 0; i < yrows; i++) {
      for (int j = 0; j < ycols; j++) cout << y[i][j];
      cout << endl;
   }
     // 345   is printed
     // 119
   return 0;
}
```

Answer:

Problem 195 Write a function called *doubleFour* that places an extra copy of the 4th digit right after that digit in an integer parameter. If there is no 4th digit, nothing is done. (Assume that the input parameter is not negative.)

For example, a program that uses the function *doubleFour* follows.

```
int main() {
   int x = doubleFour(19683);
   cout << x << endl;             // prints    196883
   cout << doubleFour(271828);    // prints   2718828
   cout << doubleFour(314159);    // prints   3141159
   return 0;
}
```

Answer:

Problem 196 Write a function called *fill2D* that fills the entries of a 2-dimensional array row by row from the entries of a 1-dimensional array. Assume that the 1-dimensional array has more than enough capacity for these entries. (The function should use capacities of the 2-dimensional array but not the 1-dimensional array as input parameters.)

For example, a program that uses the function follows.

```
int main() {
   int x[11] = {3,1,4,1,5,9,2,6,5,3,5};
   int y[2][3];
   int yrows = 2, ycols = 3;
   fill2D(y, yrows, ycols, x);
   for (int i = 0; i < yrows; i++) {
      for (int j = 0; j < ycols; j++) cout << y[i][j];
      cout << endl;
   }
     // 314   is printed
     // 159
   return 0;
}
```

Answer:

Problem 197 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {

   int i = 3, j = 5;
   int a[9] = {3,1,4,1,5,9,2,6,5};
   int x[3][2] = {{0,1},{3,2},{4,5}};

   cout << min(i, j) << endl;              // prints minimum
   printArray(x, 3, 2);                    // prints array
   cout << average(a, 9) << endl;          // prints average
   swap(a, 3, 5);                          // swap elements 3 and 5
   reverse(a[1]);                          // reverse the digits in a[1]
   return 0;
}
```

(a) Title line for **min**

Answer:

(b) Title line for **printArray**

Answer:

(c) Title line for **average**

Answer:

(d) Title line for **swap**

Answer:

(e) Title line for **reverse**

Answer:

Problem 198 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive (int n) {
   if (n < 10) return n;
   return 100 * recursive (n / 100) + 10 * (n % 10);
}

int mystery (int x) {
   cout << x << "54321";
   return x;
}

int main () {
   cout << recursive (7) << endl;     //line A
   cout << recursive (135) << endl;   //line B
   cout << recursive (19683) << endl; //line C
   cout << mystery (2) << endl;       //line D
   mystery (2);                       //line E
   return 0;
}
```

(a) What is the output at line A?

Answer:

(b) What is the output at line B?

Answer:

(c) What is the output at line C?

Answer:

(d) What is the output at line D?

Answer:

(e) What is the output at line E?

Answer:

Problem 199 Write a function called *smallestDigit* that finds the smallest digit in an integer parameter. (Assume that the input parameter is not negative.)

For example, a program that uses the function *smallestDigit* follows.

```
int main() {
   cout << smallestDigit(29) << endl;    // prints    2
   cout << smallestDigit(31415) << endl; // prints    1
   cout << smallestDigit(7) << endl;     // prints    7
   return 0;
}
```

Answer:

Problem 200 Write a function called *lastIndex* that finds the largest index of an entry in an array of integers that matches a given target. If the target is not present the function should return an answer of -1.

For example, a program that uses the function follows.

```
int main() {
   int x[6] = {3, 1, 4, 1, 5, 9};
   int capacity = 6;
   int target = 5;
   cout << lastIndex(x, capacity, target) << endl;
     // prints  4  because the target 5 is found as element number 4
   cout << lastIndex(x, capacity, 1) << endl;
     // prints  3  because the target 1 is last found as element number 3
   cout << lastIndex(x, capacity, 8) << endl;
     // prints  -1  because the target 8 is not found.
   return 0;
}
```

Answer:

Problem 201 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {

   int i = 3, j = 5;
   int a[9] = {3,1,4,1,5,9,2,6,5};
   int x[3][2] = {{0,1},{3,2},{4,5}};

   cout << average(i, j) << endl;          // prints average
   printArray(a, 9);                       // prints array
   cout << min(x, 3, 2) << endl;           // prints minimal element
   reverse(a, 9);                          // reverse the order of elements
   swap(a[1], a[2]);                       // swap two values
   return 0;
}
```

(a) Title line for **average**
Answer:

(b) Title line for **printArray**
Answer:

(c) Title line for **min**
Answer:

(d) Title line for **reverse**
Answer:

(e) Title line for **swap**
Answer:

Problem 202 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive (int n) {
   if (n < 10) return n;
   return 100 * recursive (n / 100) + 11 * (n % 10);
}

int mystery (int x) {
   cout << x << "12345";
   return x;
}

int main () {
   cout << recursive (7) << endl;     //line A
   cout << recursive (135) << endl;   //line B
   cout << recursive (19683) << endl; //line C
   cout << mystery (2) << endl;       //line D
   mystery (2);                       //line E
   return 0;
}
```

(a) What is the output at line A?

Answer:

(b) What is the output at line B?

Answer:

(c) What is the output at line C?

Answer:

(d) What is the output at line D?

Answer:

(e) What is the output at line E?

Answer:

Problem 203 Write a function called *biggestDigit* that finds the biggest digit in an integer parameter. (Assume that the input parameter is not negative.)

For example, a program that uses the function *biggestDigit* follows.

```
int main() {
   cout << biggestDigit(29) << endl;    // prints    9
   cout << biggestDigit(31415) << endl; // prints    5
   cout << biggestDigit(7) << endl;     // prints    7
   return 0;
}
```

Answer:

Problem 204 Write a function called *firstIndex* that finds the smallest index of an entry in an array of integers that matches a given target. If the target is not present the function should return an answer of -1.

For example, a program that uses the function follows.

```
int main() {
   int x[6] = {3, 1, 4, 1, 5, 9};
   int capacity = 6;
   int target = 5;
   cout << firstIndex(x, capacity, target) << endl;
     // prints  4  because the target 5 is found as element number 4
   cout << firstIndex(x, capacity, 1) << endl;
     // prints  1  because the target 1 is first found as element number 1
   cout << firstIndex(x, capacity, 8) << endl;
     // prints  -1  because the target 8 is not found.
   return 0;
}
```

Answer:

Problem 205 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {

   int a[4] = {3,1,4,1}, i = 3, j = 5, k = 4;
   int b[4] = {2,7,1,8};
   int x[2][2] = {{0,1},{3,2}};

   cout << max(i, j, k) << endl;          // prints maximum
   printMax(a, 4);                        // prints maximum
   cout << max2d(x, 2, 2) << endl;        // prints maximum
   swap(i, j);                            // swap
   swapArrays(a, b, 4);                   // swap first 4 elements in arrays
   return 0;
}
```

(a) Title line for **max**
Answer:

(b) Title line for **printMax**
Answer:

(c) Title line for **max2d**
Answer:

(d) Title line for **swap**
Answer:

(e) Title line for **swapArrays**
Answer:

Problem 206 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int main() {
  int x;
  cout << "Enter an integer:";
  cin >> x;

  if (x > 0) cout << "Goodbye" << endl;
  if (x < -10) {
     cout << x + 2 << endl;
     return 0;
  }
  else if (x % 2 != 0) cout << "odd" << endl;

  for (int i = 1; i < x; i++) cout << i;
  cout << endl;
  for (int i = 1; i <= -x; i++) {
       for (int j = 1; j <= 3; j++) cout << "*";
       cout << endl;
  }

  return 0;
}
```

(a) What is the output if the user enters *-729*?

Answer:

(b) What is the output if the user enters *4*?

Answer:

(c) What is the output if the user enters *-5*?

Answer:

(d) What is the output if the user enters *-4*?

Answer:

(e) What is the output if the user enters *3*?

Answer:

Problem 207 Write a function called *doubleFirst* that places an extra copy of the first digit at the start of a number.

For example, a program that uses the function *doubleFirst* follows.

```
int main() {
   cout << doubleFirst(29) << endl;    // prints    229
   cout << doubleFirst(19683) << endl; // prints 119683
   cout << doubleFirst(9) << endl;     // prints     99
   return 0;
}
```

Answer:

Problem 208 Write a function called *findLargest* that finds the largest possibility for the sum of the entries in a row of a 2-dimensional array of integers. The array and the capacities are parameters.

For example, a program that uses the function follows.

```
int main() {
   int d[2][3] = {{2,4,6}, {1,3,5}};
   cout << findLargest(d, 2, 3) << endl;
     // prints    12, because the sum 12 = 2+4+6 is larger than 1+3+5
   return 0;
}
```

Answer:

Problem 209 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {

   int a[4] = {3,1,4,1}, i = 3, j = 5, k = 4;
   int x[2][2] = {{0,1},{3,2}};

   cout << average(i, j, k) << endl;          // prints average
   printAverage(a, 4);                        // prints average
   cout << average2d(x, 2, 2) << endl;        // prints average
   sort(i, j ,k );                            // sort into order
   sort3(a, 4);                               // sort into order
   return 0;
}
```

(a) Title line for **average**
Answer:

(b) Title line for **printAverage**
Answer:

(c) Title line for **average2d**
Answer:

(d) Title line for **sort**
Answer:

(e) Title line for **sort3**
Answer:

Problem 210 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int main() {
  int x;
  cout << "Enter an integer:";
  cin >> x;

  if (x < 0) cout << "Goodbye" << endl;
  if (x > 10) {
     cout << x % 10 << endl;
     return 0;
  }
  else if (x % 2 != 0) cout << "odd" << endl;

  for (int i = 1; i <= x; i++) cout << i;
  cout << endl;
  for (int i = 1; i < -x; i++) {
       for (int j = 1; j < 3; j++) cout << "*";
       cout << endl;
  }

  return 0;
}
```

(a) What is the output if the user enters *729*?

Answer:

(b) What is the output if the user enters *9*?

Answer:

(c) What is the output if the user enters *5*?

Answer:

(d) What is the output if the user enters *4*?

Answer:

(e) What is the output if the user enters *-3*?

Answer:

Problem 211 Write a function called *dropSecond* that removes the second digit of an integer parameter. (Assume that the input parameter is not negative. If the parameter has just one digit, return that digit.)

For example, a program that uses the function *dropSecond* follows.

```
int main() {
   cout << dropSecond(29) << endl;    // prints    2, the 9 dropped
   cout << dropSecond(19683) << endl; // prints 1683, the 9 dropped
   cout << dropSecond(9) << endl;     // prints    9
   return 0;
}
```

Answer:

Problem 212 Write a function called *findLargest* that finds the largest entry in a specified column of a 2-dimensional array of integers. The array, the capacities, and the specified column are parameters.

For example, a program that uses the function follows.

```
int main() {
   int d[2][3] = {{2,4,6}, {1,3,5}};
   cout << findLargest(d, 2, 3, 0) << endl;
     // prints    2, because this is the largest entry in column 0
   return 0;
}
```

Answer:

Problem 213 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **num7s** which returns the number of digits equal to 7 in an input integer.

Answer:

(b) A function called **num7s** which returns the number of elements equal to 7 in an input array of integers.

Answer:

(c) A function called **num7s** which returns the number of characters equal to 7 in an input string.

Answer:

(d) A function called **num7s** which changes an integer parameter to be the number of 7's in its decimal expansion. (For example if the input is 777111 it would be changed to 3 because it has 3 digits equal to 7.)

Answer:

(e) A function called **num7s** which returns the number of elements equal to 7 in a 2-dimensional array of integers with size 7×7.

Answer:

Problem 214 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int x) {
   if (x <= 0) return 0;
   if (x >= 9 && x % 2 == 1) return x - 1;
   if (x >= 9 || x % 3 == 0) return x - 2;
   return 7;
}

int rec(int x) {
  if (x < 10) return fun(x);
  return rec(x / 10) + rec(x % 10);
}

int main() {
    cout << fun(3) << endl;    // line (a)
    cout << fun(30) << endl;   // line (b)
    cout << fun(33) << endl;   // line (c)
    cout << rec(33) << endl;   // line (d)
    cout << rec(999) << endl;  // line (e)
}
```

(a) What is the output at line (a)?
Answer:

(b) What is the output at line (b)?
Answer:

(c) What is the output at line (c)?
Answer:

(d) What is the output at line (d)?
Answer:

(e) What is the output at line (e)?
Answer:

Problem 215 Write a function called *startBinary* that returns a number giving the first 2 digits in the binary expansion of an integer parameter. (Assume that the input parameter is not negative. If the parameter has just one binary digit, return that digit.)

For example, a program that uses the function *startBinary* follows.

```
int main() {
   int x = startBinary(6);
   cout << x << endl;                // prints 11 because 6 in binary is 110
   cout << startBinary(23) << endl; // prints 10 because 23 is 10111 in binary
   cout << startBinary( 3) << endl; // prints 11 because  3 is    11 in binary
   cout << startBinary( 1) << endl; // prints  1 because  1 is     1 in binary
   return 0;
}
```

Answer:

Problem 216 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter a positive integer n that is less than 100. If the user enters an incorrect value, the program terminates. The program next asks the user to enter n^2 strings to be stored in a 2-dimensional array with size $n \times n$. The program then reports the maximum number of times that it can find the string *Kamil* in any row or column of the array.

For example, if the user enters 4 for n and then enters the 16 strings:

```
Kamil Peter  Dustin Kamil
Kamil Andrew Carl   Phil
Rat   Rat    Rat    Rat
Kamil Peter  Dustin Kamil
```

The final output would be 3 because Kamil appears three times in the first column, and no more than three times in any row or column.

Answer:

Problem 217 Write header lines (prototypes) for the following functions. **Do not attempt to supply the blocks for the functions**.

(a) A function called **isNegative** that tests whether a decimal number is negative.

Answer:

(b) A function called **thirdChar** which uses a string as input and returns the third character in the string.

Answer:

(c) A function called **swapLast2** which modifies an array of integers. The task of the function is to swap the last two elements of the array.

Answer:

(d) A function called **printPic** which uses as input an 6×6 array of characters that represents a picture. The task of the function is to print the picture.

Answer:

(e) A function called **reverseArray** which is to reverse the order of elements in an array of integers.

Answer:

Problem 218 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void mystery(int data[], int p, int q) {
  data[p] = data[q];
  data[q] = data[p];
}

void m2(int &p, int q) {
  int temp = p;
  p = q;
  q = temp;
}

void print(int data[], int p) {
  for (int i = 0; i < p; i++)
     cout << data[i] << " ";
  cout << endl;
}

main() {
  int x[8] = {0, 1, 2, 3, 4, 5, 6, 7};
  int y[7] = {0, 1, 2, 3, 4, 5, 6};
  int a = 3, b = 4;

  print(x, 3);                          // line (a)
  mystery(x, 1, 2);  print(x, 5);       // line (b)
  for (int i = 1; i <= 7; i++) mystery(x, 0 ,i);
  print(x, 8);                          // line (c)
  m2(a, b);    cout << a << b << endl;  // line (d)
  m2(y[3], 7);       print(y, 6);       // line (e)
}
```

(a) What is the output at line (a)?

Answer:

(b) What is the output at line (b)?

Answer:

(c) What is the output at line (c)?

Answer:

(d) What is the output at line (d)?

Answer:

(e) What is the output at line (e)?

Answer:

Problem 219 Write a function called *doubleDigit* that makes each digit of an input parameter repeat twice.

For example, a program that uses the function *doubleDigit* follows.

```
int main() {
   cout << doubleDigit(9)   << endl;          // prints 99
   cout << doubleDigit(81)  << endl;          // prints 8811
   cout << doubleDigit(243) << endl;          // prints 224433
   cout << doubleDigit(244) << endl;          // prints 224444
   return 0;
}
```

Answer:

Problem 220 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter 1000 single digit integers. It then outputs the digit or digits that appears least often.

For example, if the user enters $3, 1, 4, 1, 5, 9, \ldots, 9, 8$ where 0 appears 93 times, 1 appears 116 times, 2 appears 103 times, 3 appears 103 times, 4 appears 93 times, 5 appears 97 times, 6 appears 94 times, 7 appears 95 times, 8 appears 101 times, 9 appears 105 times the output would be:

```
The digits 0 and 4 are least frequent.
```

Answer:

Problem 221 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **detectAge** which returns a user's age (by asking for input and rejecting negative values).

Answer:

(b) A function called **sortString** that sorts an array of strings into alphabetical order.

Answer:

(c) A function called **sort4** that sorts 4 integer parameters into increasing order.

Answer:

(d) A function called **printCode** that prints the ASCII code for a character.

Answer:

(e) A function called **delete7** which alters an integer parameter by deleting every occurrence of the digit 7.

Answer:

Problem 222 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void mystery(int x[][4], int a, int b, int k) {
  for (int r = a; r <= b; r++) for (int c = a; c <= b; c++)
      x[r][c] = k;
}

void print(int x[][4], int s) {
  for (int r = 0; r < s; r++) {
     for (int c = 0; c < s; c++) cout << x[r][c];
     cout << endl;
  }
  cout << endl;
}

int main() {
  int x[4][4] = {{0,0,0,0}, {0,0,0,0}, {0,0,0,0}, {0,0,0,0}};
  mystery(x, 3, 2, 1); print(x, 4);  // line (a)
  mystery(x, 0, 1, 2); print(x, 4);  // line (b)
  mystery(x, 1, 2, 3); print(x, 4);  // line (c)
  mystery(x, 1, 3, 4); print(x, 4);  // line (d)
  mystery(x, 0, 3, 5); print(x, 2);  // line (e)
  return 0;
}
```

(a) What is the output at line (a)?
Answer:

(b) What is the output at line (b)?
Answer:

(c) What is the output at line (c)?
Answer:

(d) What is the output at line (d)?
Answer:

(e) What is the output at line (e)?
Answer:

Problem 223 Write a function called *cutNine* that prints the part of a number that follows its last 9 digit. (If there is no 9 digit, the whole number is printed. If the last digit is a 9, nothing is printed.)

For example, a program that uses the function *cutNine* follows.

```
int main() {
   cutNine(770);         cout << endl;          // prints 770
   cutNine(135792468);   cout << endl;          // prints 2468
   cutNine(1991991);     cout << endl;          // prints 1
   cutNine(1991999);     cout << endl;          // prints
   return 0;
}
```

Answer:

Problem 224 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter 1000 single digit integers. It then outputs the number of times that each digit was seen.

For example, if the user enters $3, 1, 4, 1, 5, 9, \ldots, 9, 8$ where 0 appears 93 times, 1 appears 116 times, …, 9 appears 105 times, the output would be:

```
0 count 93, 1 count 116, 2 count 103, 3 count 103, 4 count 93,
5 count 97, 6 count 94, 7 count 95, 8 count 101, 9 count 105.
```

Answer:

Problem 225 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **add3** which returns the sum of three double parameters.

Answer:

(b) A function called **reverseIt** that returns the number obtained by reversing the digits in an integer parameter.

Answer:

(c) A function called **randomArray** that sets the values in an array of doubles to have random values.

Answer:

(d) A function called **add5** that adds 5 to every entry in a 2-dimensional array each of whose rows has 35 columns.

Answer:

(e) A function called **deleteX** which alters a string parameter by deleting every occurence of the letter X.

Answer:

Problem 226 Consider the following C++ program.

```
#include <iostream>
using namespace std;

string fun(string x[], int y) {
   if (y <= 0) return x[1];
   if (y == 1) return x[0] + x[2];
   if (y == 2) return "illegal";
   if (y <= 4) return " 4";
   return "X" + fun(x, y - 6);

}

int main() {
  string array[3] = { "1", "2", "3"};
  cout << fun(array,0) << endl;           // line a
  cout << fun(array,1) << endl;           // line b
  cout << fun(array,2) << endl;           // line c
  cout << fun(array,4) << endl;           // line d
  cout << fun(array,12) << endl;          // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

(b) line b:

(c) line c:

(d) line d:

(e) line e:

Problem 227 Write a function called *makeOne* that returns the result of turning every odd valued digit in an integer parameter to a 1.

For example, a program that uses the function *makeOne* follows.

```
int main() {
   cout << makeOne(770)   << endl;    // prints 110
   cout << makeOne(13579) << endl;    // prints 11111
   cout << makeOne(1000)  << endl;    // prints 1000
   return 0;
}
```

Answer:

Problem 228 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter 3 positive integers. It then outputs the least frequently encountered digit or digits in those 3 numbers.

For example, if the user enters the integers 123, 45678, and 200 the program should output 9 which occurs less often than any other digit in these numbers.

Answer:

Problem 229 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **add3** which returns the sum of three integer parameters.

Answer:

(b) A function called **reverseString** that returns the reverse of a string.

Answer:

(c) A function called **randomArray** that sets the values in an array of integers to have random values.

Answer:

(d) A function called **add3** that adds 3 to every entry in a 2-dimensional array each of whose rows has 25 columns.

Answer:

(e) A function called **deleteX** which alters a string parameter by deleting every occurence of the letter X.

Answer:

Problem 230 Consider the following C++ program.

```
#include <iostream>
using namespace std;

string fun(string x[], int y) {
   if (y <= 0) return x[0];
   if (y == 1) return x[1] + x[2];
   if (y == 2) return "illegal";
   if (y <= 4) return " <= 4";
   return "X" + fun(x, y - 5);

}

int main() {
  string array[3] = { "1", "2", "3"};
  cout << fun(array,0) << endl;            // line a
  cout << fun(array,1) << endl;            // line b
  cout << fun(array,2) << endl;            // line c
  cout << fun(array,4) << endl;            // line d
  cout << fun(array,12) << endl;           // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

(b) line b:

(c) line c:

(d) line d:

(e) line e:

Problem 231 Write a function called *makeOne* that returns the result of turning every non-zero digit in an integer parameter to a 1.

For example, a program that uses the function *makeOne* follows.

```
int main() {
   cout << makeOne(770)   << endl;    // prints 110
   cout << makeOne(13579) << endl;    // prints 11111
   cout << makeOne(1000)  << endl;    // prints 1000
   return 0;
}
```

Answer:

Problem 232 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

The program asks the user to enter 3 positive integers. It then outputs the most frequently encountered digit or digits in those 3 numbers.

For example, if the user enters the integers 737, 13579, and 246 the program should output 7 which occurs more often than any other digit in these numbers.

Answer:

Problem 233 Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```
int main() {
   int a[4] = {3,1,4,1}, b[5] = {2,7,1,8,1}, i = 3, j = 5, k = 4;
   int x[2][2] = {{0,1},{3,2}};
   cout << max(x, 2 , 2); // outputs:  3
   printArray(a, 4);  // outputs:  3,1,4,1
   reverse(a, 0, 3);  // changes a to 1,4,1,3
   sort1(b, 5);
   printArray(b, 5);  // outputs:  1,1,2,7,8
   sort2(i, j, k);
   cout << i << j << k << endl;  // outputs: 345
   return 0;
}
```

(a) Title line for **max**

Answer:

(b) Title line for **printArray**

Answer:

(c) Title line for **reverse**

Answer:

(d) Title line for **sort1**

Answer:

(e) Title line for **sort2**

Answer:

Problem 234   Consider the following C++ program.

```
#include <iostream>
using namespace std;

void rec(int a[], int start, int stop) {
   if (stop <= start) return;
   a[start] = a[stop];
   rec(a, start + 1, stop -1);
}

void printA(int a[], int cap) {
  for (int c = cap - 1; c >= 0; c--) cout << a[c] << " ";
  cout << endl;
}

int main() {
  int x[6] = {0, 1, 2, 3, 4, 5};

  printA(x, 6);                   // line (a)
  printA(x, 4);                   // line (b)
  rec(x, 3, 3); printA(x, 4);     // line (c)
  rec(x, 3, 4); printA(x, 6);     // line (d)
  rec(x, 0, 5); printA(x, 6);     // line (e)

  return 0;
}
```

What is the output at each of the following lines?

(a) line (a)

(b) line (b)

(c) line (c)

(d) line (d)

(e) line (e)

Problem 235 Write a function called *maxMid* that determines the maximum value in the middle column of a 2-dimensional array of numbers of type double. (You should assume that the 2-dimensional array has an odd number of columns.)

For example, a program that uses the function *maxMid* follows. Your function must complete this program.

```
int main() {
   double x[4][5] = {{0,1,2,3,4}, {1,2,3,4,5}, {2,3,4,5,6}, {5,6,7,8,9}};
   cout << maxMid(x, 4, 5) << endl;  // prints 7.0
   return 0;
}
```

Answer:

Problem 236 Write a complete C++ program that does the following. (In your program, you do not need to check whether the user enters legal input.)

1. It asks the user to enter a positive integer n that is at most 100.

2. The program reads n single digit integers entered by the user. (A single digit integer is an integer n with $0 \leq n \leq 9$.)

3. The program prints a list of all single digit integers that were not entered at all by the user.

For example, the following represents one run of the program.

```
Enter a positive integer (at most 100):    11
Enter 11 single digit integers:
1 1 7 3 3 2 0 3 7 7 7
The following were not entered: 4 5 6 8 9
```

Answer:

Problem 237 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **welcome** which prints the word ”Hello” to the screen.

Answer:

(b) A function called **addTwo** that adds 2 to every entry in an array of integers.

Answer:

(c) A function called **randomTruth** which determines and returns a random true/false result.

Answer:

(d) A function called **numberPrimes** which returns the number of prime numbers that lie between a specified pair of input values.

Answer:

(e) A function called **biggerAverage** which determines which of two arrays of integers has the bigger average. It should return the value of this bigger average.

Answer:

Problem 238 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int fun(int &x, int y) {
   x = y + 1;
   y = x + 1;
   cout << x << y << endl;
   return x * y;
}

int main() {
  int x = 2, y = 0;
  fun(x, 8);                    // line a
  fun(x, y);                    // line b
  fun(y, x);                    // line c
  fun(y, x);                    // line d
  cout << fun(x, 3) << endl;    // line e
  return 0;
}
```

What is the output from the program at each of the following lines:

(a) line a:

(b) line b:

(c) line c:

(d) line d:

(e) line e:

Problem 239 Write a function called *alternates* that prints every second digit of an integer parameter, starting from the right.

For example, a program that uses the function *alternates* follows.

```
int main() {
   alternates(10);  cout << endl;     // prints 0
   alternates(123456); cout << endl;  // prints 642
   alternates(1000); cout << endl;    // prints 00
   return 0;
}
```

Answer:

Problem 240 Write a complete C++ program that does the following. (Programs that correctly carry out some of the tasks will receive partial credit.)

1. It asks the user to enter a positive integer that is between 1 and 26.

2. The program reads a value n entered by the user. If the value is not legal, the program exits.

3. The program prints an $n \times n$ pattern of characters, in which the top left character is an 'A'. The top left 2×2 block is completed by three 'B' characters. The top left 3×3 block is completed by five 'C' characters, and so on.

For example, if the user enters 5 for n the program should print the following picture.

```
ABCDE
BBCDE
CCCDE
DDDDE
EEEEE
```

Answer:

Problem 241 Write title lines (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **firstDigit** which returns the first digit of an integer.

Answer:

(b) A function called **sqrt** that returns the square root of a double precision parameter.

Answer:

(c) A function called **oddString** which returns a string made up of the characters in odd position of an input string.

Answer:

(d) A function called **randomWord** which is to create and return a random word.

Answer:

(e) A function called **sort** which is to sort an array of strings into alphabetical order.

Answer:

Problem 242 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive(int n) {
   if (n < 10) return n;
   if (n < 100) return n/10;
   return 10 * recursive(n / 100) + n % 10;
}

main() {
  int x;
  cout << "Enter an integer: ";
  cin >> x;
  cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters 5 for x.

(b) The user enters 16 for x.

(c) The user enters 123 for x.

(d) The user enters 1234 for x.

(e) The user enters 19683 for x.

Problem 243 Write a function called *evens* that deletes all odd digits from a positive integer parameter.

For example, a program that uses the function *evens* follows.

```
int main() {
   cout << evens(16) << endl;       // prints 6
   cout << evens(666) << endl;      // prints 666
   cout << evens(777) << endl;      // prints 0
   return 0;
}
```

Answer:

Problem 244 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 100.
2. The program reads in a 2-dimensional array with n rows and n columns of integers entered by the user.
3. The program prints out the average of the entries for each column of the array.

For example, the following represents one run of the program.

```
Enter a positive integer (at most 100):     3
Enter 3 rows of 3 integers:
 3 -1    4
10 30 -100
 2 -2   99
The averages of the 3 columns are: 5.0 9.0 1.0
```

Answer:

Problem 245 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Include declarations for any variable that you use.

(i) Print the remainder when 101 is divided by 17 to the file *out.txt*.

(ii) Print a random lower case letter to the screen. (The random letter should be determined by using an appropriate C++ function.)

(iii) Read a line of text from the user and print the word *Yes* if it contains the character 7.

(iv) Print the middle character of the string s. Assume that the string has odd ength.

(v) Swap the values of integer variables called x and y.

Problem 246 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive(int n) {
   if (n < 10) return n;
   return 100 * recursive(n / 100) + 11* (n % 10);
}

main() {
  int x;
  cout << "Enter an integer: ";
  cin >> x;
  cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters 5 for x.

(b) The user enters -10 for x.

(c) The user enters 65 for x.

(d) The user enters 123 for x.

(e) The user enters 19683 for x.

Problem 247 Write a function called *twoPart* that returns the largest power of 2 that divides a positive integer parameter.

For example, a program that uses the function *twoPart* follows.

```
int main() {
   cout << twoPart(16) << endl;   // prints 16
   cout << twoPart(666) << endl;   // prints 2
   cout << twoPart(777) << endl;   // prints 1
   return 0;
}
```

Answer:

Problem 248 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 100.
2. The program reads in a 2-dimensional array with n rows and n columns of integers entered by the user.
3. The program prints out the maximum entry found for each row of the array.

For example, the following represents one run of the program.

```
Enter a positive integer (at most 100):     3
Enter 3 rows of 3 integers:
3 -1 4
10 30 -100
0 0 0
The maximum entries in the 3 rows are: 4 30 0
```

Answer:

Problem 249 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part:

```
  int x[10], z[10][10], r, c;
  string s;
```

(i) Print the remainder when r is divided by c.

(ii) Set r to be a random integer between 1 and 10. (The random integer should be determined by an appropriate C++ function.)

(iii) Print the sum of all 10 entries of the array x.

(iv) Print the last character of the string s.

(v) Swap row number 0 with row number 4 in the 2-dimensional array z.

Problem 250 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void x1(int a[][6], int n) {
   for (int i = 0; i < 5; i++) cout << a[n][i];
   cout << endl;
}

void x2(int b[][6], int n) {
   for (int i = 0; i < n; i++)
      cout << b[i][i] << " ";
   x1(b, n);
}

main() {
  int x[6][6], a[6][6], b[6][6];
  for (int i = 0; i < 6; i++) for (int j = 0; j < 6; j++) {
     x[i][j] = i + j;
     a[i][j] = i * j;
     b[i][j] = (i + 1) / (j + 1);
  }
  cout << "Part a: " << x[5][4] << endl;
  cout << "Part b: " << a[5][4] << endl;
  cout << "Part c: "; x1(x, 5);
  cout << "Part d: "; x2(x, 5);
  cout << "Part e: "; x2(b, 3);
  return 0;
}
```

Complete the line of output that begins:

Part a:

Part b:

Part c:

Part d:

Part e:

Problem 251 Write a function called *sixCount* that returns a count of the number of digits that are equal to 6 in its positive integer parameter.

For example, a program that uses the function *sixCount* follows.

```
int main() {
   cout << sixCount(16) << endl;   // prints 1
   cout << sixCount(666) << endl;   // prints 3
   cout << sixCount(777) << endl;   // prints 0
   return 0;
}
```

Answer:

Problem 252 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer n that is at most 100.
2. The program reads in an array n integers entered by the user.
3. The program prints the negative entries from the array, in order.
4. The program prints the positive entries from the array in reverse order.

For example, the following represents one run of the program.

```
Enter a positive integer (at most 100):     8
Enter 8 integers:  3 -1 4 -10 17 18 19 -11
-1 -10 -11
19 18 17 4 3
```

Answer:

Problem 253 Write C++ statements to carry out the following tasks. **Do not write complete programs**, just give a single line, or a few lines of C++ instructions. Assume that the following variables have been declared, and if necessary have values, for each part:

```
  int x[10], y[10], z[10][10], r, c;
```

(i) Read 10 integers into the array x.

(ii) Set all the entries of the array z so that the entry in row r and column c stores the product of r and c.

(iii) Print the smallest value in the array x.

(iv) Print the word *Divides* if r divides exactly into c otherwise do nothing.

(v) Swap each entry of the array x with the correpsonding entry of array y.

Problem 254 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive(int n) {
   if (n < 100) return n%10;
   return 10 * recursive(n / 100) + n % 10;
}

main() {
  int x;
  cout << "Enter an integer: ";
  cin >> x;
  cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters -10 for x.

(b) The user enters 5 for x.

(c) The user enters 55 for x.

(d) The user enters 123 for x.

(e) The user enters 19683 for x.

Problem 255 Write a function called *toTen* that calculates how many entries of an array need to be added to make a sum of 10 or more. (Start adding from index 0.)

For example, a program that uses the function *toTen* follows.

```
int main() {
   int x[8] = {5, 3, 1, 6, 10, 1, -30, -100};
   cout << toTen(x, 8) << endl;
   return 0;
}
```

The output from this program would be 4, because the sum of the first 4 entries $5 + 3 + 1 + 6$ is the first sum that exceeds 10.

Answer:

Problem 256 Write a complete C++ program that does the following.

1. It asks the user to enter their name as a string *name*.
2. The program reads the name entered by the user.
3. The program converts all letters in the name to uppercase and prints the name.
4. The program prints the uppercase characters of the name in reverse.

For example, the following represents one run of the program.

```
What is your name:     Freddy
FREDDY
YDDERF
```

Answer:

Problem 257 Write header lines (prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **sumDigits** which returns the sum of the digits of an integer.

Answer:

(b) A function called **isSmall** that returns an answer of true if a double precision parameter has a value between 0 and 0.001. (It returns false otherwise.)

Answer:

(c) A function called **randomLetter** which generates and returns a random letter of the alphabet. (The output is to be a single character between 'A' and 'Z'.)

Answer:

(d) A function called **sort3** which is to change a collection of three input values so that they appear in increasing order.

Answer:

(e) A function called **total** which is to determine the sum of all the entries in an array.

Answer:

Problem 258 Consider the following C++ program.

```
#include <iostream>
using namespace std;

int recursive(int n) {
   if (n < 10) return n;
   return n % 10 - recursive(n/10);
}

main() {
  int x;
  cout << "Enter a positive integer: ";
  cin >> x;
  if (x <= 0) cout << "Error" << endl;
  else cout << recursive(x) << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters 0 for x.

(b) The user enters 5 for x.

(c) The user enters 55 for x.

(d) The user enters 555 for x.

(e) The user enters 19683 for x.

Problem 259 Write a function called *quadratic* that calculates the value of a quadratic function $ax^2 + bx + c$. For example, a program that uses the function *quadratic* follows.

```
int main() {
   double a = 1.0, b = 2.2, c = 1.21, x = 0.1;
   cout << quadratic(a, b, c, x) << endl;
   return 0;
}
```

Answer:

Problem 260 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer value, n.

2. The program reads a value entered by the user. If the value is not positive, the program should terminate.

3. The program should consider every number x between 1 and n and print out any value of x that divides exactly into n.

The printed values should all appear on a single line, separated by spaces.

For example, the following represents one run of the program. (The user chooses the number 28.)

```
Enter a positive integer:     28
1 2 4 7 14 28
```

Answer:

Problem 261 Write header lines (prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **sum** which returns the sum of 4 double precision values.

Answer:

(b) A function called **midDigit** that is used to return the middle digit of an integer.

Answer:

(c) A function called **isPositive** which is to return an answer of true if the sum of the entries of an array of double precision data is positive (and return false otherwise).

Answer:

(d) A function called **average2DArray** which is to print (to cout) the average of the entries in a 2-dimensional array (the array stores integers and has 10 rows and 15 columns).

Answer:

(e) A function called **makeZero** which is to use two integer input variables and change their values to zero. (After the function ends, the input variables must be zero.)

Answer:

Problem 262 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void mystery(int n) {
   cout << n % 100;
   if (n < 1000) return;
   mystery(n/10);
}

main() {
  int x;
  cout << "Enter an integer: ";
  cin >> x;
  mystery(x);
  cout << endl;
  return 0;
}
```

What is the output from the program in response to the following user inputs.

(a) The user enters 5 for x.

(b) The user enters 512 for x.

(c) The user enters 4370 for x.

(d) The user enters 175560 for x.

Problem 263 Write a function called *sum2D* that returns the sum of all elements in a 2-dimensional array that has 4 columns of integer entries.

For example, a program that uses the function *sum2D* follows.

```
int main() {
   int array[3][4] = {{1,2,3,4},{1,2,3,4},{1,2,3,4}};
   cout << sum2D(array, 3, 4) << endl;
   return 0;
}
```

The input values 3 and 4 specify the number of rows and columns in the array. The program should print an answer of 30 (since this is the sum of $1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3$, and 4).

Answer:

Problem 264 Write a complete C++ program that does the following.

1. It asks the user to enter a 5-digit integer value, n.
2. The program reads a value entered by the user. If the value is not in the right range, the program should terminate.
3. The program calulates and stores the 5 individual digits of n.
4. The program outputs a "bar code" made of 5 lines of stars that represent the digits of the number n.

For example, the following represents one run of the program. (The user chooses the number 16384.)

```
Enter a 5 digit integer:     16384
*
******
***
********
****
```

Answer:

Problem 265 Write header lines (prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **lastDigit** that is used to find the last digit of an integer.

Answer:

(b) A function called **average** which determines the average of 3 integer values.

Answer:

(c) A function called **largest** which is used to find the largest value in an array of double precision data.

Answer:

(d) A function called **print2DArray** which is to print out all of the data in a 2-dimensional array (the array has 100 columns).

Answer:

(e) A function called **sort** which is to sort an array of strings into alphabetical order.

Answer:

Problem 266 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void mystery(int data[], int p, int q) {
  data[p] = data[q];
  data[q] = data[p];
}

void m2(int p, int q) {
  int temp = p;
  q = p;
  p = temp;
}

void print(int data[], int p) {
  for (int i = 0; i < p; i++)
     cout << data[i] << " ";
  cout << endl;
}

main() {
  int scores[8] = {3, 1, 4, 1, 5, 9, 2, 6};
  int quiz[7] = {0, 1, 2, 3, 4, 5, 6};
  print(scores, 3);
  print(quiz, 4);
  mystery(scores, 1, 2);
  print(scores, 5);
  for (int i = 0; i < 3; i++)
     m2(quiz[i], quiz[i+ 1]);
  print(quiz, 6);
}
```

What is the output from the program?

Problem 267 Write a function called *countChange* that uses four parameters q, d, n, and p and converts the value of q auarters, d dimes, n nickels, and p cents into dollars.

For example, a program that uses the function *countChange* follows.

```
int main() {
   int q = 10, d = 5, n = 1, p = 2;
   double x = countChange(q, d, n, p);
   cout << "You have $" << x << endl;
}
```

It should print:

```
You have $3.07
```

Answer:

Problem 268 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer value, r that is at most 100.

2. The program reads a value entered by the user. If the value is not in the right range, the program should terminate.

3. The program reads and stores r integers from the user and then prints a pattern of r rows of stars, the lengths of which are the other integers entered by the user.

For example, the following represents one run of the program.

```
How many rows? 4
Enter 4 row lengths:  2 7 1 5
**
*******
*
*****
```

Answer:

Problem 269 Write a C++ program that asks a user how many times it should say hello and then says hello the required number of times. For example, a run of the program might produce the following output:

```
How many hellos do you want: 6
Hello Hello Hello Hello Hello Hello
```

Problem 270 Two numbers are considered as very different if they differ by more than 10. Write a C++ function called areVeryDifferent that determines whether two integers are very different.

For example, your function could be used in the following program.

```
int main() {
   int x = 4, y = 10, z = -4;
   if (areVeryDifferent(x, y)) cout << "x and y are very different" << endl;
   if (areVeryDifferent(x, z)) cout << "x and z are very different" << endl;
   if (areVeryDifferent(y, z)) cout << "y and z are very different" << endl;
   return 0;
}
```

The output from this program would be:

```
y and z are very different
```

Problem 271 Write a complete C++ program that does the following.

1. It asks the user to enter a positive integer value, x that is at most 100.
2. The program reads a value entered by the user. If the value is not in the right range, the program should terminate.
3. The program reads and stores x words from the user and then prints them in reverse order.

For example, the following represents one run of the program.

```
How many words? 5
Freddy and Max were absent
absent were Max and Freddy
```

Answer:

Problem 272 Consider the following C++ program.

```
#include <iostream>
using namespace std;

void mystery(int data[], int p, int q) {
  data[p] = data[q] + data[p];
  data[q] = 0;
}

void print(int data[], int p) {
  for (int i = 0; i < p; i++)
     cout << data[i] << " ";
  cout << endl;
}

main() {
  int scores[8] = {3, 1, 4, 1, 5, 9, 2, 6};
  int quiz[7] = {0, 1, 2, 3, 4, 5, 6};
  print(quiz, 7);
  print(scores, 8);
  mystery(scores, 3, 4);
  print(scores, 8);
  for (int i = 1; i < 7; i++)
     mystery(quiz, 0, i);
  print(quiz, 7);
}
```

What is the output from the program?