QUEENS COLLEGE Department of Computer Science
CSCI 111 Second Midterm Exam Spring 2018 05.07.18

Solutions

09.05am – 09.55am, Monday, May 07, 2018

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

**Answer:**

```
int max(int a, int b)
```

(b) Title line for **maximum**.

**Answer:**

```
int maximum(int a, int b, int c, int d)
```

(c) Title line for **print**.

**Answer:**

```
void print(double a[], int b, int c)
```

(d) Title line for **addOn**.

**Answer:**

```
void addOn(int &a, int b)
```

(e) Title line for **challenge**.

**Answer:**

```
int challenge(int a, int b)
```

**Problem 2** 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:**

```
2
```

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

**Answer:**

```
5
```

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

**Answer:**

```
345
```

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

**Answer:**

```
4
```

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

**Answer:**

```
44
```

**Problem 3** 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 $\alpha$ and $\beta$ 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:**

```
void sumDiff(int one[], int two[], int c) {
   for (int i = 0; i < c; i++) {
      int temp = one[i] + two[i];
      two[i] = one[i] - two[i];
      one[i] = temp;
   }
}
```

**Problem 4** 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:**

```
void display(int n) {
   if (n <= 0) return;
   display(n / 10);
   for (int i = 1; i <= n % 10; i++) cout << i;
   cout << endl;
}
```

QUEENS COLLEGE Department of Computer Science
CSCI 111 Second Midterm Exam Spring 2018 05.07.18

Solutions

09.05am – 09.55am, Monday, May 07, 2018

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

**Answer:**

```
void max(int a, int b)
```

(b) Title line for **second**.

**Answer:**

```
int second(int a, int b, int c, int d, int e)
```

(c) Title line for **print**.

**Answer:**

```
void print(double a, int b)
```

(d) Title line for **interchange**.

**Answer:**

```
void interchange(int &a, int &b)
```

(e) Title line for **challenge**.

**Answer:**

```
double challenge(int a, double b)
```

**Problem 2** 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:**

```
5
```

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

**Answer:**

```
3
```

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

**Answer:**

```
3445
```

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

**Answer:**

```
4
```

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

**Answer:**

```
43
```

**Problem 3** 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:**

```
void parity(int one[], bool two[], int c) {
   for (int i = 0; i < c; i++) {
      two[i] = (one[i] % 2) == 0;
   }
}
```

**Problem 4** 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:**

```
void display(int n) {
   if (n <= 0) return;
   display(n / 10);
   for (int i = 9; i >= 1; i--)
      if (i <= n % 10) cout << "X";
      else cout << " ";
   cout << endl;
}
```

QUEENS COLLEGE Department of Computer Science
CSCI 111 Second Midterm Exam Spring 2018 05.07.18

Solutions

02.45pm – 03.35pm, Monday, May 07, 2018

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

**Answer:**

```
int lastDigit(int x)
```

(b) Title line for **randomEntry**.

**Answer:**

```
int randomEntry(int x[][3], int r, int c)
```

(c) Title line for **department**.

**Answer:**

```
string department(string x)
```

(d) Title line for **doubleOrNothing**.

**Answer:**

```
void doubleOrNothing(int &x)
```

(e) Title line for **odds**.

**Answer:**

```
int odds(int x[], int cap)
```

**Problem 2** 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:**

```
14
```

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

**Answer:**

```
5
```

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

**Answer:**

```
345
```

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

**Answer:**

```
4
```

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

**Answer:**

```
71
```

**Problem 3** 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:**

```
double percentTrue(bool a[], int c) {
   int count = 0;
   for (int i = 0; i < c; i++)
      if (a[i]) count++;
   return count * 100.0 / c;
}
```

**Problem 4** 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:**

```
double sumRatios(int x, int y) {
   if (x == 0) return 0;
   return sumRatios(x / 10, y/10) + ((double) (x % 10)) / (y % 10);
}
```

QUEENS COLLEGE Department of Computer Science
CSCI 111 Second Midterm Exam Spring 2018 05.07.18

Solutions

02.45pm – 03.35pm, Monday, May 07, 2018

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

**Answer:**

```
double max(double x, int y, int z)
```

(b) Title line for **min**.

**Answer:**

```
int min(int x, int y)
```

(c) Title line for **doubleIt**.

**Answer:**

```
void doubleIt(int &x)
```

(d) Title line for **printIt**.

**Answer:**

```
void printIt(int x[], int n)
```

(e) Title line for **sum**.

**Answer:**

```
int sum(int x, int y)
```

**Problem 2** 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:**

```
8
```

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

**Answer:**

```
3
```

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

**Answer:**

```
3445
```

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

**Answer:**

```
4
```

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

**Answer:**

```
77
```

**Problem 3** 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:**

```
double percentPositive(double a[][4], int r, int c) {
   int count = 0;
   for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++)
         if (a[i][j] > 0) count++;
   return count * 100.0 / (r * c);
}
```

**Problem 4** 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:**

```
int digitDifferences(int x, int y) {
   if (x == 0) return 0;
   int diff = x % 10 - y % 10;
   if (diff < 0) diff = -diff;
   return 10 * digitDifferences(x/10,y/10) + diff;
}
```