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.

```c++
int main() {
    int x = 248, y = 39, z[4] = {1, 3, 5, 7};

    // the function digitsUp increases all digits that are smaller than 9
    digitsUp(x); // (a)
    cout << x << endl; // prints 359

    // the function printProduct prints the product of the digits here 27
    printProduct(y); // (b)

    // the function has5 determines whether an array has 5 as an element
    if (has5(z, 4)) cout << "Yes\n"; // (c)

    // Hint: look at (e) to fully solve the mystery in (d)
    cout << mystery(z[0], 4); // (d)
    puzzle(mystery(z[0],4), puzzle(has5(z, 3), "Hello")); // (e)
    return 0;
}
```

(a) Title line for `digitsUp` as called at the line marked (a).
   Answer: `void digitsUp(int &x)`

(b) Title line for `printProduct` as called at the line marked (b).
   Answer: `void printProduct(int x)`

(c) Title line for `has5` as called at the line marked (c).
   Answer: `bool has5(int x[], int c)`

(d) Title line for `mystery` as called at the line marked (d).
   Answer: `bool mystery(int x, int y)`

(e) Title line for `puzzle` as called at the line marked (e).
   Answer: `string puzzle(bool x, string y)`
Problem 2  Consider the following C++ program.

```cpp
int a(int &x, int y) {
    x = y;
    y = x + 1;
    return y;
}

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

void c(int x, int y) {
    if (x > 2) return;
    cout << y;
    c(x + 1, y - 1);
}

int main() {
    int x[2][3] = {{1, 2, 3}, {4, 5, 6}};
    int y[3] = {7, 8, 9};
    cout << x[1][1] << endl; // line (a)
    y[0] = a(y[0], y[1]);
    cout << y[0] << y[1] << endl; // line (b)
    cout << b(x[0][2], x[1][2]) << endl; // line (c)
    cout << x[0][2] << x[1][2] << endl; // line (d)
    c(0, 4); cout << endl; // line (e)
}
```

(a) What is the output from the instruction beginning on line (a)?
Answer:

5

(b) What is the output from the instruction beginning on line (b)?
Answer:

98

(c) What is the output from the instruction beginning on line (c)?
Answer:

4

(d) What is the output from the instruction beginning on line (d)?
Answer:

46

(e) What is the output from the instruction beginning on line (e)?
Answer:

432
Problem 3  Write a function called `findRed` that counts the number of elements in an array of strings that contain the string "red". For example "Fred" would be counted but "Jack" would not.

Excessively long solutions that use more than 12 lines of code may lose points. A program that uses the function `findRed` follows.

```c++
int main() {
    string x[4] = {"Freddy", "is", "not", "bored"};
    cout << findRed(x, 4) << endl; // prints 2 since "Freddy" and "bored" count
    return 0;
}
```

Answer:

```c++
int findRed(string a[], int c) {
    int ans = 0;
    for (int i = 0; i < c ; i++) {
        int x = a[i].find("red");
        if (x >= 0 && x < a[i].length()) ans++;
    }
    return ans;
}
```
**Problem 4**  Write a function called `firstOddDigit` that returns the first odd digit in a positive integer parameter. If there is no odd digit, or the parameter is negative the function should return $-1$.

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

```cpp
int main() {
    cout << firstOddDigit(19683) << endl; // prints 1
    cout << firstOddDigit(256) << endl; // prints 5
    cout << firstOddDigit(225) << endl; // prints 5
    cout << firstOddDigit(2048) << endl; // prints -1
    cout << firstOddDigit(-19) << endl; // prints -1
    return 0;
}
```

**Answer:**

```cpp
int firstOddDigit(int x) {
    if (x <= 0) return -1;
    int ans = firstOddDigit(x / 10);
    if (ans == -1 && x % 2 != 0) ans = x % 10;
    return ans;
}
```

Alternative solution (with a loop instead of recursion):

```cpp
int firstOddDigit(int x) {
    int ans = -1;
    while (x > 0)
        if (x % 2 != 0) ans = x % 10;
    return ans;
}
```
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.

```c++
int main() {
    int x = 248, y = 39, z[4] = {1, 3, 5, 7};
    // the function swapLast swaps the last digits of two numbers
    swapLast(x, y); // (a)
    cout << x << " " << y << endl; // prints 249 38
    // the function product returns the product of the elements here 105
    cout << product(z, 4) << endl; // (b)
    // the function hasEven determines whether any parameter is even
    if (hasEven(z[1], z[2], z[0] + z[3])) cout << "Yes\n"; // (c)
    // Hint: look at (e) to fully solve the mystery in (d)
    cout << mystery(z, z); // (d)
    puzzle(mystery(z,z), puzzle(hasEven(z[1], z[2], z[3]), x + y)); // (e)
    return 0;
}
```

(a) Title line for `swapLast` as called at the line marked (a).
**Answer:** `void swapLast(int &x, int &y)`

(b) Title line for `product` as called at the line marked (b).
**Answer:** `int product(int x[], int cap)`

(c) Title line for `hasEven` as called at the line marked (c).
**Answer:** `bool hasEven(int x, int y, int z)`

(d) Title line for `mystery` as called at the line marked (d).
**Answer:** `bool mystery(int x[], int y[])`

(e) Title line for `puzzle` as called at the line marked (e).
**Answer:** `int puzzle(bool x, int y)`
Problem 2 Consider the following C++ program.

```cpp
int a(int &x, int y) {
    x = y + 1;
    y = x + 1;
    return y;
}

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

void c(int x, int y) {
    if (x > 2) return;
    cout << y;
    c(x + 1, y + 2);
}

int main() {
    int x[2][3] = {{3, 2, 1}, {6, 5, 4}};
    int y[3] = {9, 8, 7};
    cout << x[1][1] << endl; // line (a)
    y[0] = a(y[0], y[1]);
    cout << y[0] << y[1] << endl; // line (b)
    cout << b(x[0][2], x[1][2]) << endl; // line (c)
    cout << x[0][2] << x[1][2] << endl; // line (d)
    c(0, 4); cout << endl; // line (e)
}
```

(a) What is the output from the instruction beginning on line (a)?

**Answer:**

5

(b) What is the output from the instruction beginning on line (b)?

**Answer:**

108

(c) What is the output from the instruction beginning on line (c)?

**Answer:**

2

(d) What is the output from the instruction beginning on line (d)?

**Answer:**

34

(e) What is the output from the instruction beginning on line (e)?

**Answer:**

468
Problem 3  Write a function called `seesRed` that finds the index of the first element that contains the word "red" in an array of strings. For example "Fred" contains "red" but "Jack" does not. Your function should return \(-1\) in case no element of the array contains red.

Excessively long solutions that use more than 12 lines of code may lose points. A program that uses the function `seesRed` follows.

```cpp
int main() {
    cout << seesRed(x, 4) << endl;  // prints 1 since "Freddy" is the first element containing "red"
    return 0;
}
```

Answer:

```cpp
int seesRed(string a[], int c) {
    for (int i = 0; i < c ; i++) {
        int x = a[i].find("red");
        if (x >= 0 && x < a[i].size()) return i;  // size() can be used in place of length()
    }
    return -1;
}
```
Problem 4  Write a function called `hasOddDigit` that tests whether a positive integer parameter has an odd digit. If there is no odd digit, or the parameter is negative the function should return `false`.

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

```cpp
int main() {
    if (hasOddDigit(19683)) cout << "Yes\n"; // prints \em Yes
    if (hasOddDigit(256)) cout << "Yes\n"; // prints \em Yes
    if (hasOddDigit(2048)) cout << "Yes\n"; // does not print
    if (hasOddDigit(-19)) cout << "Yes\n"; // does not print
    return 0;
}

Answer:

```cpp
bool hasOddDigit(int x) {
    if (x <= 0) return false;
    return hasOddDigit(x / 10) || x % 2 != 0;
}
```

Alternative solution (with a loop instead of recursion):

```cpp
bool hasOddDigit(int x) {
    while (x > 0)
        if (x % 2 != 0) return true;
        else x = x / 10;
    return false;
}
```