MATH CSI 32 - Programming II

TEST 1. Time allowed: two hours. Professor Luis Fernández

NAME: _____

PART 1:

Write the answer to the following exercises. You may not use the computer in this part, but you can look at the book. Justify your answer briefly when indicated.

---

[10] **1.** Answer the following short questions.

a) What `#include` statement do you put at the top of a program that lets you use `cin` or `cout`?

b) What `using` statement do you always put at the top of your programs (to get standard naming conventions)?

c) Which are the three iteration (or *looping*) statements in C++?

d) What does the keyword `continue` do?

.

e) What does the keyword `break` do?

---

[10] **2.** What is the output of this program?

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

int X{12};

void value1()//Definition of function value1
{
  cout << X << endl;
}

void value2(int val)//Definition of function value2
{
  cout << val << endl;
}

int main()
{
  int X{45};

  value1();

  value2(X);

  {
    int X{23};
    value2(X);
  }

  value1();
}
```

Thus the output is

12

45

23

12

[10] **3.** Hand-write a `switch` statement that prints out the name of the month given the number of the month. So for example, if the variable `numMon` is the number of the month (between 1 and 12), the `switch` statement should print `January` if `numMon = 1`, `February` if `numMon = 2`, etc.

[10] **4.** Hand-write the definition of a function with void output that receives a string as input **by reference** and prints out the string.

[10] **5.** What does this program do?

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

int mystery(int, int); // function prototype

int main()
{
   cout << "Enter two integers: ";
   int x{0};
   int y{0};
   cin >> x >> y;
   cout << "The result is " << mystery(x, y) << endl;
}

int mystery(int a, int b) //Function definition
{
   if (1 == b)// base case
   {
      return a;
   }
   else// recursion step
   {
      return a + mystery(a, b - 1);
   }
}
```

PART 2:

Write 5 (five) of the following programs. If you finish the 5, do the rest for extra credit. You may use the book, but **you cannot check the internet**. Make sure it runs, although partial credit may be given even if it does not.

When you are finished, **upload your files to the Dropbox link I sent you today**.

---

[10] **1.** Write a program that writes out your name 100 times, each time in a new line.

---

[10] **2.** Write a program that contains a function `sum` that finds the sum of all the integers from 1 to a positive number given by the user (so for example, if the user enters 5, the output should be 15, because $1+2+3+4+5 = 15$).

---

[10] **3.** Write a program that simulates tossing two coins 100,000 times and counts the results. There are 3 possibilities: two heads, one head and one tail, two tails. The output should look something like this (probably with different numbers!):

```
Number of two heads: 249312
Number of one head and one tail: 499829
Number of two tails: 250859
```

---

[10] **4.** Write a program that prints out the even numbers from 2 to 100. Print the numbers out distributing them in 10 rows of 5 numbers each. You can use the command `setw()` from `iomanip` to format the output, or you can also tabs (`"\t"`). The output should look something like

```
  2   4   6   8  10
 12  14  16  18  20
 22  24  26  28  30
 32  34  36  38  40
 42  44  46  48  50
 52  54  56  58  60
 62  64  66  68  70
 72  74  76  78  80
 82  84  86  88  90
 92  94  96  98 100
```

---

[10] **5.** Write a program that contains a **recursive** function `power(base, exponent)` that returns $\text{base}^{\text{exponent}}$. In the `main` part of the program, ask the user to enter base and exponent and use the function to return $\text{base}^{\text{exponent}}$. NOTE: The function in this exercise **has to** be defined recursively. You cannot use the library `cmath`.

---

[10] **6.** Write a program that reads five positive integers and, for each number, displays that number of asterisks. For example, if the numbers entered are 3, 5, 9, 2 and 6, the program should output:

```
***
*****
*********
**
******
```

[10] **7.** Consider the following sequence of numbers: starting at any integer, say a_1, to calculate the next member of the sequence do the following:

- If a_1 is even, then divide it by 2 (that is, the next element of the sequence will be $a_1/2$).
- If a_1 is odd, then multiply it by 3 and add 1 (that is, the next element of the sequence will be $3a_1 + 1$).

Repeat this for ever, or until you reach 1. For example, starting with 5 we would get

$$5, 16, 8, 4, 2, 1,$$

(5 is odd, so we do $3 \cdot 5 + 1 = 16$;
16 is even, so we do 16/2=8;
8 is even, so we do 8/2=4;
4 is even, so we do 4/2=2;
2 is even so we do 2/2=1.)

Or, for example, starting with 17 we would get

$$17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1.$$

Misteriously, it seems that no matter the starting number, eventually we get to 1. It is an **open problem in Mathematics** whether this is true for *every* starting number.

Write a C++ program that implements this. In other words, ask the user for a starting value `firstElement`, write a function `nextElement(x)` that gives $x/2$ if x is even and $3x + 1$ if x is odd, and use a while loop to iterate the function (that is, if the current element of the sequence is called `element`, do `element = nextElement(element)` to find the next element), and print the elements of the sequence until 1 is reached.