CSCI 13500 Midterm S22 v1 EMPLID | | | | | | | | |

NAME: FIRST LAST

SEAT ROW NUMBER

1. (30 points) Answer the following questions.

(1) Declare an array of ints called values with size 5. Initialize the elements, from the first one to the last one, to be 1, 6, 5, 9, 8.

```
int values[] = {1, 6, 5, 9, 8};
Or
int values[5] = {1, 6, 5, 9, 8};
Or
int values[5];
values[0] = 1;
values[1] = 6;
values[2] = 5;
values[3] = 9;
values[4] = 8;
```

(2) Declare function foo whose input parameter is a string (pass by value) and return is an int. You just need to write the function header, no implementation is needed.

```
int foo(string str);
```

(3) Write code to generate a random floating point number in [-2, 1].

(1) To generate a random floating number in [0, 1], use 1.0 * rand() / RAND_MAX. Note that 1.0 * rand() converts rand() to double type, otherwise rand() / RAND_MAX returns either integer 0 or integer 1.
(2) To generate a random floating number in [0, 3], since the distance between -2 and 1 is 3, is 1.0 * rand() / RAND_MAX * 3, or simplified as 3.0 * rand() / RAND_MAX.
(3) To generate a random floating number in [-2, 1] is to shift the result in (2) by -2, that is, -2 + 3.0 * rand() / RAND_MAX.
(4) Complete code is double num = -2 + 3.0 * rand() / RAND_MAX;

(4) Given array of strings as follows
string greetings[] = {"Hi", "Hello", "Great!"};
What is the value for greetings[2][5]?

greetings[2] is string "Great!", greetings[2][5] is the letter indexed at 5, which is the sixth letter (the first letter is indexed at 0), so the answer is letter ‘!’.

(5) How to run the default compiled runnable file generated by g++ on a correct cpp file?

```
./a.out
```

(6) What is the output of the following code?

```
int value = 1;
for (int i = 2; i <= 6; i += 3)
{
    value += i;
    cout << value << endl;
}
```

In the very beginning value is 1.

variable i	condition i <= 6	value += i	cout << value << endl	i += 3
2	Yes	value becomes 3	print 3 in a line	i becomes 5
5	Yes	value becomes 8	Print 8 in a line	i becomes 8
8	No			

Output

```
3
8
```

(7) Write code to calculate the square root of a, where a is a positive double number. Hints: you may use sqrt function.

```
double value = sqrt(a);
```

(8) If m is 3 and n is 1, what is the value of 1 / 2.0 * m * n?

The answer is 0.5 * 3 * 1, which is 1.5
1 / 2.0 returns 0.5

(9) Suppose n is an int >= 10, write code to extract its last two digits. For example, suppose n is 21, after your code, you get 21; if n is 1265, then your code gets 65.

```
int lastTwoDigits = n % 100;
```

(10) What is the output of the following code?

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

void foo(int size);

int main()
{
   foo(4);
   return 0;
}

void foo(int size)
{
  for (int numAsts = size; numAsts >= 1; numAsts--)
  {
     for (int i = 0; i < size - numAsts; i++)
        cout << "#";
     for (int i = 0; i < numAsts; i++)
        cout << "*";
     cout << endl;
  }
}
```

Here is an analysis when size is 4.

numAsts	numAsts >= 1	for (int i = 0; i < size - numAsts; i++) cout << "#"; print "#" (size-numAsts) times	for (int i = 0; i < numAsts; i++) cout << "*"; print "*" numAsts times	cout << endl;	numAsts--
4	yes	no # is printed	Print four *'s, that is, ****	Print a new line, the screen looks like ****	3
3	yes	Print one #, that is, #	Print three *'s, that is, ***	Print a new line, the screen	2

				looks like **** #***	
2	yes	Print two #'s, that is, ##	Print two *'s, that is, **	Print a new line, the screen look like **** #*** ##**	1
1	yes	Print three #'s, that is, ###	Print one *'s, that is, ###*	Print a new line, the screen look like **** #*** ##** ###*	0
0	no				

The output is

```
****
#***
##**
###*
```

2. (45 points) Short programming exercises
   (2.1) Write code in main that generate 20 random integers in [10, 100], tally how many of grades are >= 70, and how many grades are between 60 (included) and 70 (not included), and how many grades are < 60.

```
#include <iostream>
#include <cstdlib> //rand,
#include <ctime> //use time
using namespace std;

//Write code in main that generate 20 random integers
//in [10, 100], tally how many of grades are >= 70,
//and how many grades are between 60 (included) and 70
//(not included), and how many grades are < 60.

//keys:
//(1) generate 20 random integers
//for (int i = 0; i < 20; i++)
//    generate a random int in [10, 100]
//(2) generate a random int in [10, 100] is
//    score = rand() % (100 - 10 + 1) + 10;
//    assume that score is declared as an int.
//    There are a total of 100 - 10 + 1 = 91 integers
//    in [10, 100],
//    expression rand() % (100 - 10 + 1) returns
//    a random int in [0, 90].
//    expression rand() % (100 - 10 + 1) + 10 returns
//    a random int in [10, 100].
//(3) Before the loop, set number of each category
//    to be 0.
//    Use nested if-else statement to
//    tally each category.

int main()
{
    srand(time(NULL));
       //Use the current time as a seed
       //to generate random int.

    int score;
    int numGE70 = 0; //score >= 70
    int num60To70 = 0; //score >= 60 && score < 70
    int numLT60 = 0; //score < 60
    for (int i = 0; i < 20; i++)
    {
        //Generate a random int in [10, 100].
        score = rand() % (100 - 10 + 1) + 10;
        if (score >= 70)
           numGE70++;
        else if (score >= 60)
                numGE60To70++;
             else numLT60++;
    }

    //print out tallied result (optional)
    cout << "score >= 70: " << numGE70 << endl;
    cout << "score >= 60 and score < 70: "
         << num60To70 << endl;
    cout << "score < 60: " << numLT60 << endl;
    return 0;
}
```

(2.2) Define a function, sort two given strings according to their lengths, return type is void. For example, suppose string a is “seller” (six letters) and string b is “buyer” (five letters), after calling this function, a is changed to “buyer” and b is changed to “seller”. Suppose a is “good” and b is “better”, then after the function, a and b do not change.

```
#include <iostream>
#include <cassert> //assert
using namespace std;

//Define a function, sort two given strings
//according to their lengths, return type is void.
//For example, suppose string a is "seller"
//(six letters) and string b is "buyer" (five letters),
//after calling this function, a is "buyer" and
//b is "seller".
//Suppose a is "good" and b is "better",
//then after the function, a and b are not changed.

void sort(string& a, string& b);

int main()
{
    string a("seller");
       //initialize a string a by "seller",
       //you can write as
       //string a = "seller"; as well.
       //string is few (if not only) type in C++
       //that can be initialized as a class constructor
       //like type objectName(parameter_in_constructor)
       //or like a primitive type use
       //type variableName = value_of_that_type;

    string b("buyer");

    cout << "Before sorting" << endl;
    cout << "a = " << a << endl;
    cout << "b = " << b << endl;

    sort(a, b);

    cout << "After sorting" << endl;
    cout << "a = " << a << endl;
    cout << "b = " << b << endl;

    a = "good";
    b = "better";
    cout << "\nBefore sorting" << endl;
        //\n means new line
    cout << "a = " << a << endl;
    cout << "b = " << b << endl;

    sort(a, b);

    cout << "After sorting" << endl;
    cout << "a = " << a << endl;
    cout << "b = " << b << endl;

    //Let assert function do an auto check.
    assert(a == "good" && b == "better"
           && a.length() <= b.length());

    return 0;
}

void sort(string& a, string& b)
{
    if (a.length() > b.length())
    {
       string temp = a;
       a = b;
       b = temp;
    }
}
```

(2.3) Write code.
Create an ifstream object fin that reads text file named data.txt.

```
ifstream fin("data.txt");
```

Write code to ignore the first line of data.txt.
Hints: you might want to use ignore method of file stream.

```
istream& ignore (streamsize n = 1, int delim = EOF);
```

Extract and discard characters
Extracts characters from the input sequence and discards them, until either *n* characters have been extracted, or one compares equal to *delim*.

```
fin.ignore(INT_MAX, '\n');
```

Suppose data.txt has only a column of decimal numbers. Write code to find out and print the minimum value.

```
double value;
fin >> value; //read the first data

if (fin.fail()) //no data
  return 0;

double min = value;
   //use the first data to initialize min
while (fin >> value)
{
   if (value < min)
      min = value;
}

cout << "min = " << min << endl;
```

Call close method of fin.

```
fin.close();
```

A complete code is as follows.

```
#include <iostream>
#include <fstream> //ifstream, ofstream
using namespace std;

//Here is an example of contents of data.txt
//scores
//92.6
//25.1
//53.6

//Sample output:
//min = 25.1
int main()
{
    ifstream fin("data.txt");
        //fin opens file "data.txt" to read.

    //skip the first line, which is just column names.
    //Skip INT_MAX letters or till meeting '\n',
    //whichever is met first.
    fin.ignore(INT_MAX, '\n');

    double value;
    fin >> value; //read the first data

    if (fin.fail()) //no data
       return 0;

    //Initialize max with one of its own data
    //Warning: do not set double max to be an arbitrary
    //value, for example, 100.
    //Then if all data are larger than 100,
    //their minimum cannot be 100.
    double min = value;
        //use the first data to initialize min
    while (fin >> value)
    {
        if (value < min)
           min = value;
    }

    cout << "min = " << min << endl;

    fin.close(); //finish reading the file opened by fin
    return 0;
}
```

3. (10 points) **Define a function**, for a given string str, return true if all its characters are only letters 'a', 'b', or 'c', otherwise, return false.

```
#include <iostream>
#include <cassert> //assert
using namespace std;

//Define a function, for a given string str,
//return true if all its characters are only
//letters 'a', 'b', 'c', otherwise, return false.

bool onlyabc(string str);

int main() //optional
{
    assert(onlyabc("abcbc") == true);
        //onlyabc("abcbc") is expected to return true.
    assert(onlyabc("") == false);
    assert(onlyabc("dbac") == false);
    return 0;
}

bool onlyabc(string str)
{
   int len = str.length();

   if (len == 0) //handle special case when str is "".
      return false;

   for (int i = 0; i < len; i++)
       //return false for first non 'a', 'b',
       //or 'c' letter
       if (str[i] != 'a' && str[i] != 'b' &&
           str[i] != 'c')
          return false;

   //test every letter in str,
   //and it is one of 'a', 'b', or 'c',
   //otherwise, we would have returned false
   //in one of the above for-loop.
   return true;
}
```

4. (15 points) Define a function, for a given int, return how many digits are 2. For example, if the given integer is 20, then the number of 2’s is 1, if the given integer is 101, then the number of 2’s is 0.

```
#include <iostream>
#include <cassert> //assert
using namespace std;

//Define a function, for a given int,
//return how many digits are 2.
//For example, if the given integer is -21,
//then the number of 2’s is 1,
//if the given integer is 101,
//then the number of 2's is 0.

int num2s(int n);

int main()
{
    assert(num2s(-21) == 1);
        //expected return of num2s(21) is 0.
    assert(num2s(-300) == 0);
        //expected return of num2s(-300) is 1.
    assert(num2s(-123) == 1);
    assert(num2s(212) == 2);

    return 0;
}

//key: throw the last digit away a time,
//until no more digit to throw away.
//Before throw away the last digit,
//check whether it is 2 or not.
//Each integer has at least a digit to be throw away,
//so we use do-while statement.
//Note that -132 % 10 returns -2,
//so need to consider when n is negative.
int num2s(int n)
{
   if (n < 0) //-132 % 10 returns -2,
      //we would not like to handle negative number
      //in the following code, so change n to be positive
      //if n is negative.
      n *= -1;

   int num2s = 0;
   int lastDigit;
   do {
       lastDigit = n % 10;
           //save last digit of n to lastDigit

       if (lastDigit == 2)
          num2s++;

       n /= 10; //throw away the last digit from n

   } while (n != 0);

   return num2s;
}
```