CSCI 13500 Midterm S22 v3 EMPLID

NAME: FIRST LAST

SEAT ROW NUMBER

1. (30 points) Answer the following questions.
   (1) Declare an array of chars called grades with size 5. Initialize the elements, from the first one to the last one, to be ‘a’, ‘b’, ‘c’, ‘d’, ‘f’.

```
char grades[] = {'a', 'b', 'c', 'd', 'f'};
Or
char grades[5] = {'a', 'b', 'c', 'd', 'f'};
Or
char grades[5];
grades[0] = 'a';
grades[1] = 'b';
grades[2] = 'c';
grades[3] = 'd';
grades[4] = 'f';
```

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

```
string foo(int n);
```

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

(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 -1 and 2 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 [-1, 2] is to shift the result in (2) by -1, that is, -1 + 3.0 * rand() / RAND_MAX.
(4) Complete code is double num = -1 + 3.0 * rand() / RAND_MAX;

   (4) Given array of strings as follows
   string greetings[] = {"Hi", "Hello", "Great!"};
   What is the value for greetings[1][2]?

greetings[1] is string "Hello", greetings[1][2] is the letter indexed at 2, which is the third letter (the first letter is indexed at 0), so the answer is letter ‘l’.

   (5) Write a command to compile hello.cpp in command line.

g++ hello.cpp (you also can write g++ hello.cpp -o myprog, where myprog is a runnable file).

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

```
int value = 0;
for (int i = 3; i <= 6; i += 2)
{
    value += i;
    cout << value << endl;
}
```

In the very beginning value is 0.

variable i	condition i <= 6	value += i	cout << value << endl	i += 2
3	Yes	value becomes 3	print 3 in a line	i becomes 5
5	Yes	value becomes 8	Print 8 in a line	i becomes 7
7	No			

Output
3
8

(7) Write code to calculate the square root of $b^2 - 4ac$, where a, b, c are properly declared and initialized. Hints: you may use sqrt function.

```
sqrt(b*b – 4 * a * c)
```

(8) If m is 5 and n is 1, what is the value of 1.0 / 2 * m * n?

The value is 0.5 * 5 * 1, which returns 2.5

(9) Suppose n is an int >= 100, write code to extract its last three digits. For example, suppose n is 121, after your code, you get 121; if n is 1265, then your code gets 265.

```
int lastThreeDigits = n % 1000;
```

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

Here is an analysis when size is 4.

num	num >= 1	for (int i = 0; i < size - num; i++) cout << "*"; print "*" (size-num) times	for (int i = 0; i < num; i++) cout << "#"; print "#" num times	cout << endl;	num--
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 looks like #### *###	2

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 36 random integers in [10, 100], tally how many of grades are larger than or equal to 90, and how many grades are between 60 (included) and 90 (not included), and how many grades are less than 60.

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

//Write code in main that generate 36 random integers
//in [10, 100], tally how many of grades are larger than //or
equal to 90, and how many grades are between 60
//(included) and 90 (not included),
//and how many grades are less than 60.

//keys:
//(1) generate 36 random integers
//for (int i = 0; i < 36; 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 numGE90 = 0; //score >= 90
    int num60To90 = 0; //score >= 60 && score < 90
    int numLT60 = 0; //score < 60
    for (int i = 0; i < 36; i++)
    {
        //Generate a random int in [10, 100].
```

```
        score = rand() % (100 - 10 + 1) + 10;

        if (score >= 90)
           numGE90++;
        else if (score >= 60)
                num60To90++;
             else numLT60++;
    }

    //print out tallied result (optional)
    cout << "score >= 90: " << numGE90 << endl;
    cout << "score >= 60 and score < 90: "
         << num60To90 << endl;
    cout << "score < 60: " << numLT60 << endl;
    return 0;
}
```

(2.2) Define a function, sort two given strings according to their lengths in **non-ascending** order (that is, the first string has no fewer letter than the second one), 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 and b are not changed. Suppose a is “good” and b is “better”, then after the function, a is “better” and b is “good”.

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

//Define a function, sort two given strings
//according to their lengths in non-ascending order
//(that is, the first string has no fewer letter than
//the second one), 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 and b are not changed.
//Suppose a is "good" and b is "better",
//then after the function, a is "better" and
//b is "good".

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 == "better" && b == "good"
           && 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 average.

```
double value;
double sum = 0;
int numScores = 0; //numScores is initialized to be 0
while (fin >> value)
{
   numScores++;
   sum += value;
}

double avg = sum / numScores; //sum is double type
cout << "average = " << avg << endl;
```

Call close method of fin.

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

A complete code is as follows.

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

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

//Sample output:
//average = 57.1
int main()
{
    ifstream fin("data.txt");
        //fin opens file "data.txt" to read.
        //ifstream means Input File Stream.

    //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;
    double sum = 0;
    int numScores = 0; //numScores is initialized to be 0
    while (fin >> value)
    {
        numScores++;
        sum += value;
    }

    double avg = sum / numScores; //sum is double type
    cout << "average = " << avg << 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 ‘0’, ‘1’, ‘2’, 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 '0', '1', '2', otherwise, return false.

bool only012(string str);

int main() //optional
{
    assert(only012("01202") == true);
        //only012("01202") is expected to return true.
    assert(only012("") == false);
    assert(only012("0123") == false);
    return 0;
}

bool only012(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 '0', '1', or '2' letter
       if (str[i] != '0' && str[i] != '1' &&
           str[i] != '2')
          return false;

   //test every letter in str,
   //and it is one of '0', '1', or '2',
   //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 3. For example, if the given integer is 21, then the number of 3's is 0, if the given integer is -300, then the number of 3's is 1.

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

//Define a function, for a given int,
//return how many digits are 3.
//For example, if the given integer is 21,
//then the number of 3's is 0,
//if the given integer is -300,
//then the number of 3's is 1.

int num3s(int n);

int main()
{
    assert(num3s(21) == 0);
        //expected return of num3s(21) is 0.
    assert(num3s(-300) == 1);
        //expected return of num3s(-300) is 1.
    assert(num3s(-123) == 1);

    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 3 or not.
//Each integer has at least a digit to be throw away,
//so we use do-while statement.
//Note that -123 % 10 returns -3,
//so need to consider when n is negative.
int num3s(int n)
{
   if (n < 0) //-123 % 10 returns -3.
      ////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 num3s = 0;
   int lastDigit;
   do {
```

```
        lastDigit = n % 10;
            //save last digit of n to lastDigit

        if (lastDigit == 3)
           num3s++;

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

    } while (n != 0);

    return num3s;
}
```