AST21105 Object-Oriented Programming & Design

Lab 2 – C++ Basics

A. Submission Details

In this lab, you are required to submit **ONE** C++ program to solve the given problem shown in the section "Mark Six". To make the program implementation easier, you are suggested to write your program using Visual Studio .NET instead of doing it directly using paper & pencil. After you have completed the implementation, submit your program file (i.e. MarkSix.cpp for this lab) using the electronic "drop-box" in Canvas, **One week** after your lab section is conducted. For details, please refer to the following:

Tuesday sections	*by*	*19:00 on 26 January 2016 (Tuesday)*
Wednesday sections	*by*	*19:00 on 27 January 2016 (Wednesday)*

You are reminded to double-check your solution to verify everything is correct before submission. You are also required to put your name and student ID at the beginning of your source file.

Important: You are only able to submit your work once.

B. Objective

The objective of this lab is to help you get familiar with variable declaration, basic array and control statements that you have learned in Java last year. Also, you will be introduced the difference between Java and C++ so as to speed you up the learning process of C++. The first part of this lab is meant to be a simple review of variable declarations and basics control statements using Microsoft Visual Studio .NET. The second part of this lab is a practical task that allows you to use all the techniques introduced (i.e. variable declarations, array allocation and control statements). You will be asked to write a program to test simulate the Mark Six lottery game and display the result on screen.

C. Review

1. Declare variables for storing values

   The set of data types supported in C++ is extremely similar to those you have learned in Java. The following shows a list of these types.

Integer data types	short, int, long
Floating point data types	float, double
Character type	char
Boolean type	bool

   You should be able to notice that the only difference between Java and C++ in the list of supporting data types would be:
   (a) No byte in C++
   (b) boolean in Java becomes bool in C++

   Also, it is worth to mention that:
   (a) Characters in C++ are represented in ASCII, while those in Java are represented in Unicode
   (b) In C++, we can declare any of the numeric data types to be unsigned, e.g.,
      (i) *unsigned int val1;* // val1 can store a value in range of 0 to 232 – 1
      (ii) *signed int val2;* // val2 can store a value in range of -2 31 to 231 – 1
      (iii) *int val3;* // val3 can store a value in range of -2 31 to 231 – 1

   Syntax:
   // Declare a variable with name <name of variable> in type <type>
   <type> <name of variable>;
   // Declare a variable with name <name of variable> in type <type> with initial value <initial value>
   <type> <name of variable> = <initial value> ;
   // Declare a number of variables with name <variable name1>, <variable name2>, … all in type <type>
   <type> <variable name1>, <variable name2>, ...;
   /* Declare a number of variables with name <variable name1>, <variable name2>, … all in type <type> with initial value <initial value1>, <initial value2>, …, respectively */
   <type> <variable name1> = <initial value1>, <variable name2> = <initial value2>, ...;
   where <type> could be in one of the following types:
   short, int, long, float, double, char, bool

2. In C++, you are allowed to use variables that are not initialized, i.e. C++ compiler won't warn you even you use un-initialized variables
   Note: Although this is allowed, you just SHOULDN'T DO

3. Obtain inputs from keyboard and store them in variables:

Syntax:
// Should include iostream in order to use cin for getting data from keyboard
#include <iostream>
using namespace std;
…
// Use cin to get data from keyboard
cin >> <variable1> >> <variable2> >> … >> <variableN>;
where <variable1>, <variable2>, …, <variableN> are variables.

4. Perform calculation using arithmetic operations:

Syntax:
// Add the two values stored in variable 1 and variable 2
<name of variable for the result> = <name of variable 1> + <name of variable 2>;
// Subtract the value stored in variable 2 from variable 1
<name of variable for the result> = <name of variable 1> - <name of variable 2>;
// Multiply the value stored in variable 1 by the value stored in variable 2
*<name of variable for the result> = <name of variable 1> * <name of variable 2>;*
// Divide the value stored in variable 1 by the value stored in variable 2
<name of variable for the result> = <name of variable 1> / <name of variable 2>;
// Find the remainder of value stored in variable 1 divided by the value stored in variable 2
<name of variable for the result> = <name of variable 1> % <name of variable 2>;

5. Basic C++ array

Recall, an array is a collection of variables that are of the same type. Comparing to Java, C++ offers the flexibility to declare an array in two different ways. In this lab, you only need to manage the first type.

Syntax:

// Delcare an array named <name of variable> of size <size> in type <type>
<type> <name of variable>[<size>];

where <type> could be in one of the following:
short, int, long, float, double, char, bool or user-defined data type

Important note:
- <size> has to be a constant, it cannot be a variable
- The initial values of the array are garbage
(Again, this is another feature that is different from Java. Recall, Java automatically initializes an array of int type to 0, array of double type to 0.0, array of boolean type to false, but this is not the case in C++)

6. Do checking using branching statements:

Syntax:

```
// If <boolean expression> is evaluated to true, <statement> is executed.
// Otherwise, just skip it.
if(<boolean expression>)
   <statement>;

// If <boolean expression> is evaluated to true, <statement 1>, <statement 2>, …, are executed.
// Otherwise, just skip them.
if(<boolean expression>) {
   <statement 1>;
   <statement 2>;
   // ...
}

// If <boolean expression> is evaluated to true, <statement A> is executed.
// Otherwise, <statement B> is executed.
if(<boolean expression>)
   <statement A>;
else
   <statement B>;

// If <boolean expression> is evaluated to true, <statement A1>, …, are executed.
// Otherwise, <statement B1>, …, are executed.
if(<boolean expression>) {
   <statement A1>;
   // ...
} else {
   <statement B1>;
   // ...
}

/* If <expression> matches a case, all the statement(s) of that case is/are executed until a break statement is
found. If none of the case matches, the statement(s) at the default case is/are executed. */
switch(<expression>) {
   case <constant a>:
      <statement a1>;
      // ...
      break;
   case <constant b>:
      <statement b1>;
      // ...
      break;
   default:
      <statement c1>;
      // ...
}
```

7. Executing statements repeatedly using looping statements:

Syntax:

```
/* If <boolean expression> is evaluated to true, <statement> is executed and this process is repeated until the <boolean expression> is evaluated to false. */
while(<boolean expression>)
   <statement>;

/* If <boolean expression> is evaluated to true, <statement 1>, …, are executed and this process is repeated until the <boolean expression> is evaluated to false. */
while(<boolean expression>) {
   <statement 1>;
   // ...
}

// Execute <statement> and evaluate the <boolean expression> to see if it is true.
/* If so, re-execute <statement> again. This process is repeated until the <boolean expression> is evaluated to false */
do
   <statement>;
while(<boolean expression>);

/* Execute <statement 1>, <statement 2>, …, and evaluate the <boolean expression> to see if it is true. If so, re-execute <statement 1>, …, again. This process is repeated until the <boolean expression> is evaluated to false. */
do {
   <statement 1>;
   // ...
} while(<boolean expression>);

/* Execute <initialize> followed by evaluating <boolean expression> at the first execution. If <boolean expression> is evaluated to true, <statement 1>, …, are executed. After this, <update> is executed at the second execution followed by <boolean expression>. If it is evaluated to true, <statement 1>, …, are executed and this process is repeated until the <boolean expression> is evaluated to false. */
for(<initialize>; <boolean expression>; <update>) {
   <statement 1>;
   // ...
}
```

8. Display information on screen:

Syntax:

```
// Should include iostream in order to use cout for displaying data on screen
#include <iostream>
using namespace std;

// Use cout to display information on screen
cout << <variable1 / value1 / string1 / expression1>
     << <variable2 / value2 / string2 / expression2>
     << ...
     << <variablen / valuen / stringn / expressionn>;
```

where <variablen / valuen / stringn> is a variable, a value, a string or an expression.

Example:

Write a program to get three numbers from users, for integer variables a, b and c. If a is not zero, find out whether a is common divisor of b and c. If a is the common divisor of b and c, display the string "a is a common divisor of b and c" on screen, otherwise, display "a is not a common divisor of b and c.). The program should allow users to use the program repeatedly until a sentinel value of a = -1 to terminate the program. (Hint: a number x is a divisor of y if the remainder of y/x is 0.)

```
#include <iostream>
using namespace std;
int main() {
   int a, b, c;
   do {
      cout << "Enter value of a: ";
      cin >> a;
      if( a == -1 )
         break;
      else if( a == 0 )
         cout << "a cannot be 0" << endl;
      else if( a != 0 ) {
         cout << "Enter value of b: ";
         cin >> b;
         cout << "Enter value of c: ";
         cin >> c;
         if( b % a == 0 && c % a == 0 )
            cout << "a is a common divisor of b and c" << endl;
         else
            cout << "a is not a common divisor of b and c" << endl;
      }
   }while( a != -1 );
   system("pause");
   return 0;
}
```

D. Mark Six

In this part, you are required to:

i) Create a New Project and give your project the name Lab2b.

ii) Add a source file to your project, called MarkSix.cpp

iii) Write a C++ program to simulate Mark Six lottery game. The program should first randomly generate 6 regular numbers and 1 extra number in range of 1 to 49. Then, it should prompt the player to enter his / her guess and finally output the
(1) randomly generated numbers,
(2) number of input values matched and
(3) what prize is awarded.
The prize awarded of Mark Six lottery game is based on the number of drawn numbers matched described as follows:

1st Prize	Matched all six drawn numbers
2nd Prize	Matched Five drawn numbers and the extra number
3rd Prize	Matched Five drawn numbers
4th Prize	Matched Four drawn numbers and the extra number
5th Prize	Matched Four drawn numbers
6th Prize	Matched Three drawn numbers and the extra number
7th Prize	Matched Three drawn numbers
No Prize	Matched less than three drawn numbers

Note: You should make sure all the randomly generated numbers are distinct and in valid range. If not, numbers should be re-generated. In addition, you should also check the user guess and ensure all the entered numbers are distinct and in range. If not, display error message and ask player to re-enter his / her guess.

A sample screen display of the program sessions is given below:

Session 1:

```
[ Mark Six Simulation ]
Six regular numbers drawn are 32 29 33 40 7 35
Extra number drawn is 11
----------------------------------------------------------------
Enter guess numbers: 2 3 5 12 64 19
Out of range number found, please re-enter
Enter guess numbers: 2 3 5 12 49 3
Repeated number found, please re-enter
Enter guess numbers: 2 3 5 12 49 1
----------------------------------------------------------------
Number of input values matched: 0
No Prize
```

Session 2:

```
[ Mark Six Simulation ]
Six regular numbers drawn are 32 28 8 35 20 9
Extra number drawn is 36
----------------------------------------------------------------
Enter guess numbers: 32 28 8 9 20 36
----------------------------------------------------------------
Number of input values matched: 5.5
2nd Prize
```

The input which is underlined corresponds to player's input.

Hints: You can use the following sample program as a reference for randomly generate numbers in C++ and the way to compare double values.

```
#include <iostream>
#include <ctime>
#include <limits>
using namespace std;
int main()
{
  // Declare a variable number to store an integer
  int number;
  // Initialize random seed (normally put only once in your program)
  srand( (unsigned int)time(NULL) );
  // Generate a random number between 1 to 9 number = rand() % 9 + 1;
  // Display the random number
  cout << "The random number generated is " << number << endl;
  // Prompt user to enter a value for comparison
  double input;
  cout << "Enter a value for comparison ";
  cin >> input;
  // Obtain a very small double value from C++
  double epsilon = numeric_limits<double>::epsilon();
  // Check if the random generated no. is the same as the input
  if(fabs(input - number) < epsilon)
    cout << "Correct" << endl;
  else
    cout << "Wrong" << endl;
  // Hold the command window
  system("pause");
  return 0;
}
```

iv) Compile your program and test it by executing your program.

✧ Your program should follow the input and output format as specified in part 3. Marks will be deducted if you failed to do so.

Marking Scheme:

Graded items	Weighting
1. Correctness of program (i.e. whether your code is implemented in a way according to the requirements as specified.)	60%
2. Indentation	30%
3. Documentation (with reasonable amount of comments embedded in the code to enhance the readability.)	10%
	100%

Program Submission Checklist

Before submitting your work, please check the following items to see you have done a decent job.

Items to be checked	☑ / ☒
1. Did I put my name and student ID at the beginning of all the source files?	☐
2. Did I put reasonable amount of comments to describe my program?	☐
3. Are they all in .cpp extension and named according to the specification?	☐
4. Have I checked that all the submitted code are compliable and run without any errors?	☐
5. Did I zip my source files using Winzip / zip provided by Microsoft Windows? Also, did I check the zip file and see if it could be opened? ***(Only applicable if the work has to be submitted in zip format.)***	☐
6. Did I submit my lab assignment to Canvas?	☐

-End-