Object-Oriented Programming Language

09/22/2016

Lab Assignment 1

Due 2030

Lab Grading Policy: Attendance 20%, Score 80%

In case you have difficulty in finishing the exercises on time, you should upload them before **noon on Saturday** and a penalty of 20% discount will be applied on your score. No late submission is permitted. We will in general post the reference solutions **by Sunday**.

(**Hint**: if you are not familiar with random number generation in C++, take a quick look of the note below. You can skip it and jump directly to Exercise 1 if you are familiar with the topic)

Random number generation

Computers are often used to generate random numbers. To do so in C++, you need to include cstdlib header file and call the random number generation function rand. Each time rand is called, it returns a random integer in the range of 0 through RAND_MAX (RAND_MAX is a compiler-dependent constant defined in cstdlib). The following code generates ten random numbers.

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

int main()
{
   int i;
   for (i=0; i<10; i++)
      cout << rand() << ' ';
   cout << endl;
   return 0;
}
```

Q: If we want to simulate throwing a dice ten times, will the above code work? If not, how to modify the code for this scenario?

A:

```
   for (i=0; i<10; i++)
      cout << rand()% 6 + 1 << ' ';
```

All random numbers are called pseudo-random. This means that **they are part of a predefined set of numbers in the language.** In order for you to mix up the set each time you run a program, you need to **seed** the generator by making use of the srand() function. A standard method of doing this is to use the computer internal clock.

The library file ctime contains a function time that uses computer internal clock to return an integer. For our purpose it is sufficed to provide an argument of 0 to the function.

```
#inlcude <ctime>

...
srand(time(0));
...
```

This will seed the generator to the current time object. Essentially, this will make the generator change it's set every time you run the program.

The modern way to generate random number in C++ is to use the random number device. Random-number-generating since C++11 is wrapped in terms of device-engine frame. The procedure begins with an object of `random_device`. then you have to make an engine for setting this device. A random number is generated with your favored distribution, in which the random engine is installed. Below is a sample of random integer generating:

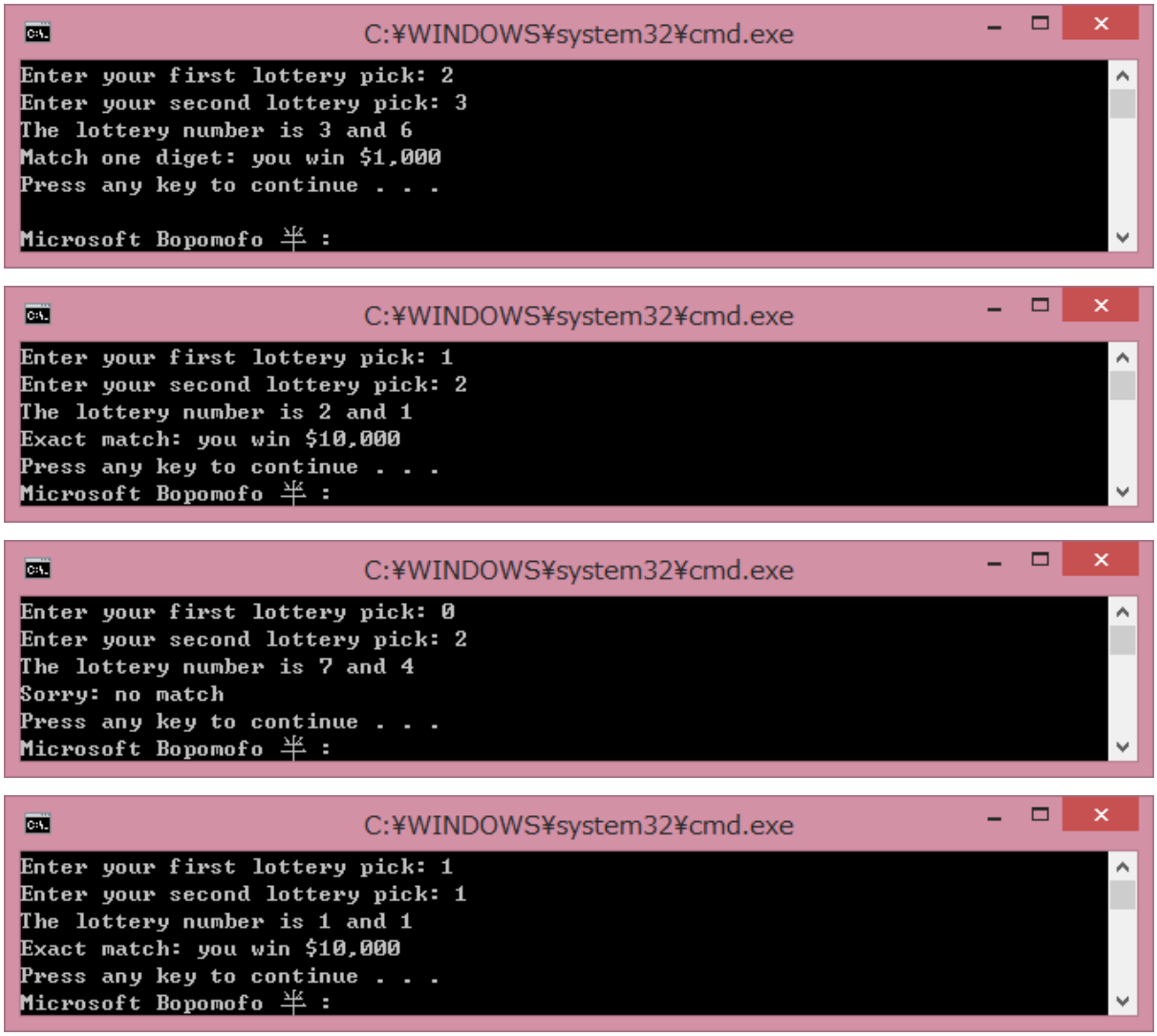
```
std::random_device rd;
std::default_random_engine r_engine(rd());
std::uniform_int_distribution<int> dist(0, 9);

int the_random_number = dist(r_engine);
```

Exercise 1 (40%): Suppose you want to develop a program to play lottery. The program randomly generates a lottery of a two-digit number, prompts the user to enter two numbers, and determines whether the user wins according to the following rules:

1. If the user input matches the lottery number in the exact order, the award is $10,000.
2. If all digits in the user input match all digits in the lottery number, the award is $3,000.
3. If one digit in the user input matches a digit in the lottery number, the award is $1,000.

Write a program to play the game and below are some sample outputs:



Exercise 2 (40%): Write a C++ program to output Celsius and Fahrenheit temperature equivalents. Ask the users to enter the range of Celsius temperatures and process the operation in a one-degree interval. Below is a sample run:

```
Please input the range of Celsius temperature values: 1 20
Celsius Fahrenheit
1       33.8
2       35.6
3       37.4
4       39.2
5       41
6       42.8
7       44.6
8       46.4
9       48.2
10      50
11      51.8
12      53.6
13      55.4
14      57.2
15      59
16      60.8
17      62.6
18      64.4
19      66.2
20      68
Press any key to continue . . .
```