物件導向程式設計

Case Study III: Random Number Generation

Joseph Chuang-Chieh Lin
Dept. CSIE, Tamkang University

Platform

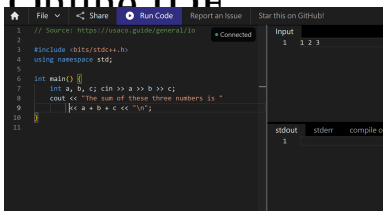
- Dev-C++

  Click here to download.

  Note: Please use this version otherwise you can't compile your programs/projects in Win10.

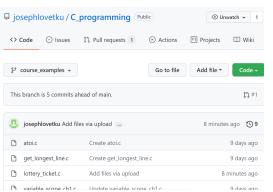
- OnlineGDB (https://www.onlinegdb.com/)

- Real-Time Collaborative Online IDE (https://ide.usaco.guide/)



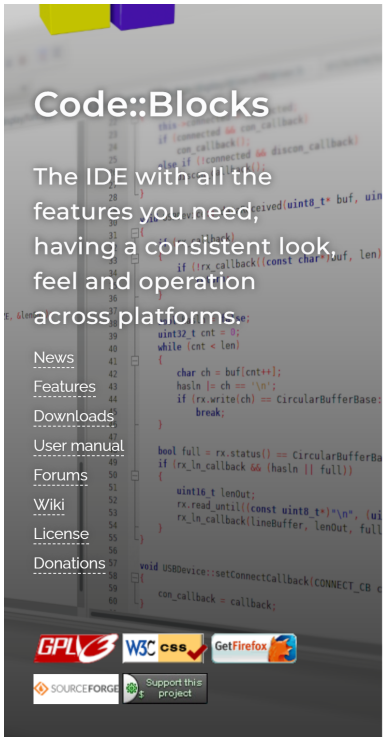
- Other resources:
  - MIT OpenCourseWare - Introduction to C++ [link].
  - Learning C++ Programming [Programiz].
  - GeeksforGeeks [link]

My GitHub page:
click the link here to visit.



Platform/IDE

- https://www.codeblocks.org/



Code::Blocks

Code::Blocks

The free C/C++ and Fortran IDE.

Code::Blocks is a free C/C++ and Fortran IDE built to meet the most demanding needs of its users. It is designed to be very extensible and fully configurable.

Built around a plugin framework, Code::Blocks can be extended with plugins. Any kind of functionality can be added by installing/coding a plugin. For instance, event compiling and debugging functionality is provided by plugins!

If you 're new here, you can read the **user manual** or visit the **Wiki** for documentation. And don't forget to visit and join our **forums** to find help or general discussion about Code::Blocks.

We hope you enjoy using Code::Blocks!

The Code::Blocks Team

Latest news

Migration successful

We are very happy to announce that the process of migrating to the new infrastructure has completed successfully!

Read more

rand()

- Required header: `cstdlib`
- It generates an unsigned integer between 0 and `RAND_MAX`.
  - `RAND_MAX` is defined in `cstdlib`.
  - Every number in the range is chosen with equal chance when `rand` is called each time.
- `rand()` actually generates pseudo-random numbers.

Rolling a Six-Sided Die

```
#include <iostream>
#include <iomanip> // for setw(); setting width of output
#include <cstdlib>

using namespace std;

int main() {
    for (unsigned int counter = 1; counter <= 20; ++counter) {
        // pick random number from 1 to 6 and output it
        cout << setw(10) << (1+rand()%6);
        // if counter is divisible by 5, start a new line
        if ( counter % 5 == 0 )
            cout << endl;
    } // end for
}
```

Try to play around the code by yourself.

Adding 'Seeds'

- `srand()`:
  - In `cstdlib` header.
  - A function takes an unsigned integer argument and seeds the rand function to produce **different** sequence of random numbers for each execution.

Rolling a Six-Sided Die **with Seeds**

```
#include <iostream>
#include <iomanip> // for setw(); setting width of output
#include <cstdlib>

using namespace std;

int main() {
    unsigned int seed = 0;
    cout << "Enter seed: ";
    cin >> seed;
    srand(seed); // seed random number generator
    for (unsigned int counter = 1; counter <= 20; ++counter) {
        // pick random number from 1 to 6 and output it
        cout << setw(10) << (1+rand()%6);
        // if counter is divisible by 5, start a new line
        if (counter%5 == 0)
            cout << endl;
    } // end for
}
```

Try to play around the code by yourself.

C++ 11 Random Numbers

- According CERT, `rand()` does not have good statistical properties and can be predictable.
- C++ 11 provides a more secure library of random-number capabilities that can't predicted.
  - Located in the **`random`** header.
- C++ 11 provides many classes that represent various random number generation *engines* and *distributions*.
  - An engine implements a random-number generation **algorithm** that produce pseudo-random numbers.
  - A distribution controls the **range** of values produced by an engine, the **types** of those values and the **statistical properties** of the values.

Example

- We consider the default engine and uniform distribution as the example.
  - `default_random_engine`
  - `uniform_int_distribution`

Rolling a Six-Sided Die (C++11 random)

```
#include <iostream>
#include <iomanip> // for setw(); setting width of output
#include <random> // C++11 random number generation features
#include <ctime> // for time() function

using namespace std;

int main() {
    default_random_engine engine(static_cast<unsigned int>(time(0)));
    uniform_int_distribution<unsigned int> randomInt(1,6);

    for (unsigned int counter = 1; counter <= 20; ++counter) {
        cout << setw(10) << randomInt(engine);
        if (counter%5 == 0)
            cout << endl;
    }
}
```

Try to play around the code by yourself.

Normal Distribution

```
#include <iostream>
#include <random> // C++11 random number generation features
#include <cmath> // for using lround; rounding a real number
#include <vector>

using namespace std;

int main() {
    default_random_engine e;
    normal_distribution<> normal(4, 1.5); // using default type: double
    vector<unsigned> vals(9); // a vector of 9 0's
    for (size_t i=0; i<100; i++) {
        unsigned v = lround(normal(e));
        if (v < vals.size())
            ++vals[v];
    }
    for (size_t j=0; j<vals.size(); j++)
        cout << j << ": " << string(vals[j], '*') << endl;

    return 0;
}
```

```
0: **
1: ***
2: *********
3: ****************
4: ********************************
5: ********************
6: ************
7: ***
8:
```