Mr. Giansante

C++ Programming

Random Numbers

August 2018

Random Numbers

It is often useful to generate random numbers to produce simulations or games.

One way to generate these numbers in C++ is to use the function `rand()`.

To use `rand()`, you must include the cstdlib library

```
#include <cstdlib>
```

The function produces pseudo-random numbers such as:

```
1804289383
846930886
1681692777
```

The numbers are called pseudo-random because they are generated by a complex formula and thus are not truly random. But they appear random.

The "random" numbers that are produced can be manipulated using math to fit into a certain range.

For example, if you are simulating a dice rolling game, we might want to generate random numbers between 1 and 6.

When we divide any number by 6, the only possible remainder are 0, 1, 2, 3, 4 or 5.

So, when we divide the number produced by rand() by 6, the result will be 0, 1, 2, 3, 4 or 5.

Recall that in C++, the % operator returns the remainder from division.

Since we actually want a number between 1 and 6, we simply need to add 1 to the remainder generated.

The general formula for generating a number between min and max is ...

```
= rand() % ( max - min + 1 ) + min
```

Example. Generate a random number between 1 and 6.

```
min = 1, max = 6

= rand() % ( 6 - 1 + 1) + 1
```

Which simplifies to ...

```
= rand() % 6 + 1
```

Example. Generate a random number that represents the age of a teenager.

Teenagers can have an age between 13 and 19, so we need to generate random numbers with ...

```
min = 13, max = 19

= rand() % ( 19 - 13 + 1) + 13
```

Which simplifies to ...

```
= rand() % 7 + 13
```

"Seeding" the Random Sequence

Each time we call `rand()`, we get the next number in the sequence.

If we want to get a different sequence of numbers for each execution, we need to go through a process of randomizing.

Randomizing is “seeding” the random number sequence, so we start in a different place. The function that does this is `srand()` which takes an integer as the seed.

It is important to only invoke the srand call ONCE at the beginning of the program. There is no need for repeat calls to seed the random number generator (in fact, it will make your number less evenly distributed).

A commonly used technique is to seed the random number generator using the clock. The `time()` function will return the computer’s time.

The function `time(NULL)` will return the number of seconds elapsed in computer time.

To use the `time()` function, we need to include the `<ctime>` library.

The combination of statements below will produce different values each time the program is run.

```
#include <cstdlib>
#include <ctime>

srand(time(NULL));

cout << (rand() %10) + 1;
```