CSE 1310 - Introduction to Computers & Programming

Pseudorandom Numbers

Alex Dillhoff

University of Texas at Arlington

Random Numbers

Generating random values is a useful function for many applications from gaming to security.

Random Numbers

For gaming applications, random numbers might be used to change the variation of difficulty of each playthrough.

Security applications use random number generation to generate complex passphrases to ensure the secure transmission of data.

Random Numbers in C

Utilities to generate random numbers are present in almost every programming language including C.

C implements these functions in `stdlib.h`.

Random Numbers in C

There are 3 functions to consider when generating random numbers in C:

1. `rand()` - Generate pseudo-random number
2. `rand_r()` - Re-entrant Version
3. `srand()` - Sets the seed of the pseudo-random number generator

Random Numbers in C

Example: Generate a random integer

Random Numbers in C

The function `rand()` returns an `int`.

If a `float` or `double` is desired, the generated integer should be divided by the maximum random value.

Random Numbers in C

Example: Generate a float

Random Numbers in C

It is also useful to generate numbers within a specific range.

For integers, this is useful for generating from a uniform distribution $[a, b]$. This can also be applied to floating point values as well.

Random Numbers in C

Example: Generate a value within a specific range

Random Numbers

Random numbers generated through most programs aren't **truly** random.

They are called **pseudorandom**.

Why **pseudo**random?

The methods employed to generate these numbers do so through a repeatable, **deterministic** process.

If the process is known along with the **seed**, the same sequence can be generated.

Setting a Static Seed

Example: Setting a static seed

Setting a Static Seed

Having the ability to generate the same sequence across different machines can be beneficial for testing and reproducibility.

Setting a Random Seed

The seed can be sourced from an external process such that the numbers generated are different with each run.

One common approach is to use the current time.

Setting a Random Seed

Example: Using time to set a random seed

Final Example

Example: Shuffle a string

Random Number Generation

One final note on random number generation: **You should generally not try to come up with your own algorithm for number generation.**

Cryptography is a complex field that studies algorithms related to security and number generation. Unless you are well-versed in this field, stick to existing implementations for random number generation and encryption.