CSCI 127: Joy and Beauty of Data

Lecture 8: Modules

Reese Pearsall
Summer 2021
https://reesep.github.io/classes/summer2021/127/main.html

Announcements

- Lab 4 due **tomorrow** @ 11:59 PM

- Lab 5 due **Wednesday** @ 11:59 PM

You have two options:

1. Come to a live virtual review session Wednesday @ 2:00 PM in my Zoom room
   - If you show up, you will receive a 100% for Lab 5

2. Answer any two example test questions and submit your python solution to D2L
   - This will largely be effort based… if you attempted them you will receive 100% for Lab 5

- Program 2 due **Sunday** 5/30 @ 11:59 PM

Announcements

Midterm Exam on Thursday

Will be posted at 8 AM on **Thursday**. Due at 8:00 AM on Friday (~24 hrs)

I'm hoping that it shouldn't take more than 2 hours to complete

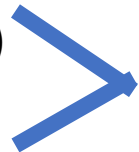**Exam Format (100 Points)**

10 multiple choice Qs / Short Answer **(20 pts)**   >   Two of these will be pulled from the practice exam

3 short programming problems **(80 pts)**

1. Basic Python Calculation/Function (20 pts)
2. Selection & Strings (30 pts)   >   One of these questions will be directly pulled from the example test questions on our course website
3. Nested List (30 pts)

Announcements

Midterm Exam on Thursday

Will be posted at 8 AM on **Thursday**. Due at 8:00 AM on Friday (~24 hrs)

I'm hoping that it shouldn't take more than 2 hours to complete

There will be two different submissions on D2L

Exam Format (100 Points)

10 multiple choice Qs / Short Answer **(20 pts)** ⟵ Filled out electronically/by hand and submitted as PDF

3 short programming problems **(80 pts)**

1. Basic Python Calculation/Function (20 pts)
2. Selection & Strings (30 pts)
3. Nested List (30 pts)

⟵ Solutions will be submitted as a .py file

Announcements

Midterm Exam on Thursday

Content covered on midterm

Data Types/Variables
Operators (+,-,*,%,...)
User input
.format
Functions
Selection
Iteration (for loops/while loops)
Lists
Strings
Modules
Recursion *(won't be a programming problem)*
Variable Scope

Good ways to study:

- Review Labs/Programs
- Review in class examples
- Do example test questions
- Read through textbook

Announcements

Extra Credit Opportunity

If you fill out a short, ***anonymous*** mid-semester survey (coming soon), I will add 1% to your midterm grade.

Submit a screenshot that you completed the survey to D2L

Announcements

Important:

You can now sign up for a 1 on 1 meeting time w/ Reese

Remember that **5%** of your final grade comes from meeting with me once during the semester

<put doodle link here>

- **EVERYONE** must sign up for a time via doodle. If none of the available times work, email me
- First come, first serve
- Please only sign up for 1 time slot. If you need more than 15 minutes, then feel free to book two consecutive time slots (times are in Mountain/Denver)

Modules

Modules are an external resource (python code) that provide a series of functionality for us to use.
We can import them to easily do things without needing to code it ourselves

For example, python can't generate random numbers on its own, so we imported the **random** module to give us the ability to generate random numbers

There are thousands of modules out there (some require a more tedious download process)

As programmers, we often rely on external modules and libraries

List of Python modules we can easily import

Global Module Index - https://docs.python.org/3/py-modindex.html

Math Module

The math module gives us access to plethora of mathematical functions

https://docs.python.org/3/library/math.html#module-math

Random Module

The random module gives us access to **(pseudo)** random number generation

https://docs.python.org/3/library/random.html#module-random

What is a pseudo random number generator?

A **deterministic** mathematical formula for generating sequences of numbers

Not *truly* random– How are we supposed to tell our computer to pick a random number?

A **seed** is given to the formula as a starting point (this seed changes each time you run a program)

42 → **PRNG**

Seed is passed into the formula to generate the first number (this is usually based your computer's clock)

PRNG

First number is generated

31

31

PRNG

Previous value gets passed into the formula to generate next number

31

PRNG

31 17 42

PRNG

31 17 42 56 74 11 12 75 66 32 54 61 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

PRNG

31 17 42 56 74 11 12 75 66 32 54 61 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

[...]

PRNG

31 17 42 56 74 11 12 75 66 32 54 61 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

[...]

... 93 38 86 6 40 20 5 88 4 68 70 95 84 35 9 83 77 19 17 62 48 15 33 43 46 63 31 17 42 56 74 11 12 75 66 32 54 61 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

PRNG

Notice anything?

31 17 42 56 74 11 12 75 66 32 54 61 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

[...]

... 93 38 86 6 40 20 5 88 4 68 70 95 84 35 9 83 77 19 17 62 48 15 33 43 46 63 31 17 42 56 74 11 12 75 66 32 54 61 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

PRNG

Our sequence will eventually start over and repeat!

31 17 42 56 74 11 12 75 66 32 54 61 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

[...]

... 93 38 86 6 40 20 5 88 4 68 70 95 84 35 9 83 77 19 17 62 48 15 33 43 46 63 **31 17 42 56 74 11 12 75 66 32 54 61** 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

PRNG

How long until the random number sequence starts to repeat?

31 17 42 56 74 11 12 75 66 32 54 61 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

[...]

????

... 93 38 86 6 40 20 5 88 4 68 70 95 84 35 9 83 77 19 17 62 48 15 33 43 46 63 **31 17 42 56 74 11 12 75 66 32 54 61** 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

PRNG

This is typically referred to as the period**. The period for Python PRNG's is 2^19937-1**

31 17 42 56 74 11 12 75 66 32 54 61 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

[...]

????

... 93 38 86 6 40 20 5 88 4 68 70 95 84 35 9 83 77 19 17 62 48 15 33 43 46 63 **31 17 42 56 74 11 12 75 66 32 54 61** 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

PRNG

This is typically referred to as the period**. The period for Python PRNG's is 2^19937-1**

31 17 42 56 74 11 12 75 66 32 54 61 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

[...]

????

… 93 38 86 6 40 20 5 88 4 68 70 95 84 35 9 83 77 19 17 62 48 15 33 43 46 63 **31 17 42 56 74 11 12 75 66 32 54 61** 22 16 39 64 98 50 49 41 42 36 28 45 7 47 76 97 69 37 2 73 34 60 92 59 18 82

So, what does this mean???

If you know how the underlying PRNG works for some application, and you know what you **seed** you are giving it:

**You can determine what random numbers will be generated**

This is something to think about--- We rely on random number from computers all the time

FYI -- It's really difficult to successfully crack PRNGs like this. There are many protections in place to prevent humans from cheating the system

But let's look at times when PRNGs have been exploited……

Video Games

Video games **heavily** rely on random numbers. The gameplay and what players experience is often determined by RNG
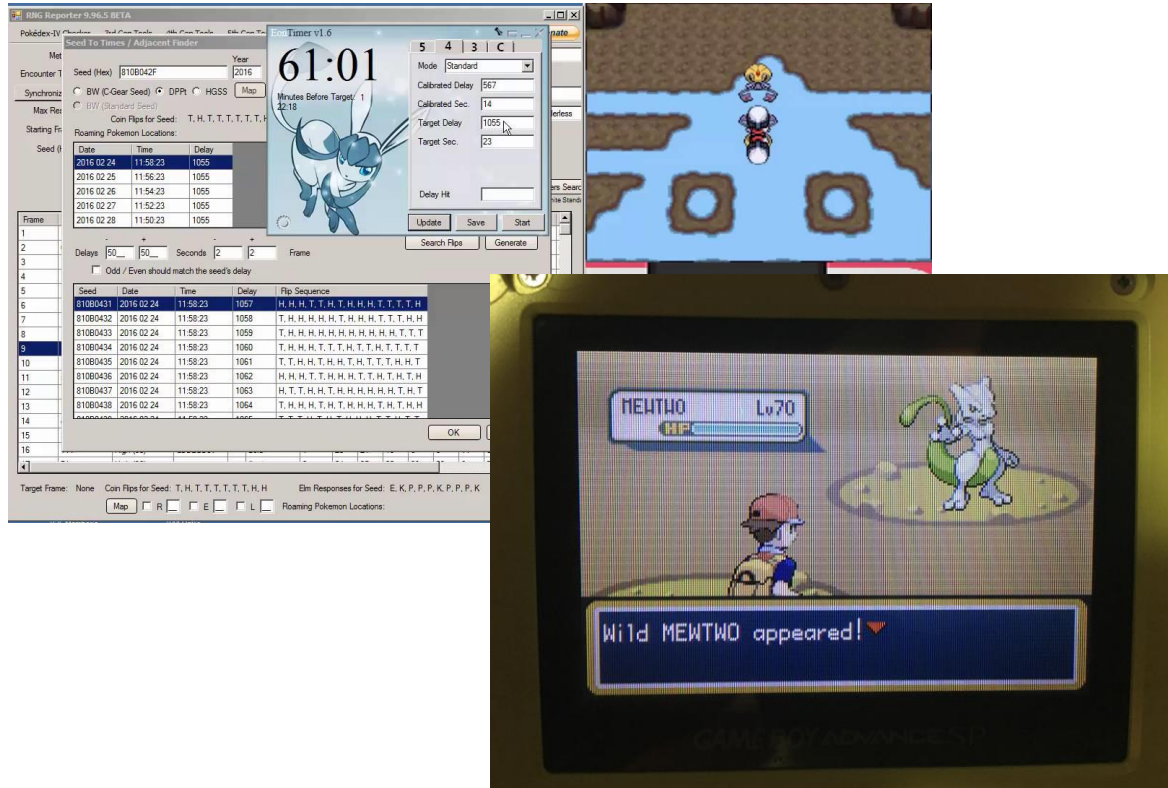
Some players have figured out how certain game's PRNGs work, and they determined how the seed is determined

Which means if they have the correct setup, they can **control** the outcome of the game

Gambling Machines

Hacking Slot Machines by Reverse-Engineering the Random Number Generators

Interesting story:

> The venture is built on Alex's talent for reverse engineering the algorithms — known as pseudorandom number generators, or PRNGs — that govern how slot machine games behave. Armed with this knowledge, he can predict when certain games are likeliest to spit out moneyinsight that he shares with a legion of field agents who do the organization's grunt work.
>
> These agents roam casinos from Poland to Macau to Peru in search of slots whose PRNGs have been deciphered by Alex. They use phones to record video of a vulnerable machine in action, then transmit the footage to an office in St. Petersburg. There, Alex and his assistants analyze the video to determine when the games' odds will briefly tilt against the house. They then send timing data to a custom app on an agent's phone; this data causes the phones to vibrate a split second before the agent should press the "Spin" button. By using these cues to beat slots in multiple casinos, a four-person team can earn more than $250,000 a week.