

Objectives

- More Assignments and Arithmetic
- Input

Jan 18, 2023

Sprenkle - CSC111

1

1

From Last Friday: Looking Ahead

- Textbook Pre Lab 1 assignment due before lab on Tuesday
 - Covers some things we haven't yet covered in class; we'll review on Tuesday
- Extra Credit Opportunity:
 - Read an article that relates to CS
 - Summarize it on the discussions under "Extra Credit"
 - 5 pts extra credit added to lab grade

Jan 18, 2023

Sprenkle - CSC111

2

2

Review

Note: using slightly different terminology
Goal: comfort with terminology, synonyms

- What are the two modes for running Python?
- How do we tell our program to display output?
- How can we store information?
 - What is the syntax to do that?
- What are the rules and conventions for variable names?
 - What is another term for “variable names”?
 - Describe characteristics of *good* variable names
- What are the primitive types of information in Python?
- What are the arithmetic operators? Describe their syntax and semantics.
- What is our development process?
 - 1) Generally and 2) For our labs, on Linux/Ubuntu
- What are the expectations for a complete program for this class?

Jan 18, 2023

Sprenkle - CSC1111

3

3

Review: Printing Multiple Things

- **print** is a special command or a *function*
- To display multiple things on the same line, separate them with commas
 - `print("Hello,", "class")`
 - `print("x =", 5)`
 - `print(x*y, "is the magic number")`
 - `print(r, s, t)`

Syntax: ,
Semantics: display this too, separated by a space in the display

Jan 18, 2023

Sprenk

4

Review: Formalizing Process of Developing Computational Solutions

1. Create a sketch of how to solve the problem (the algorithm)
2. Fill in the details in Python
3. Execute the program
4. If output doesn't match your expectation
 - Debug the program (Where is the problem? How do I fix it?)

Not necessarily complete program at first

Jan 18, 2023

Our development process will evolve over time

5

5

Development Process for Lab

- Develop in **IDLE**
 1. Create a new file
 2. Develop the program (following previous slide)
 3. Close the shell
 4. Run the program again
 5. Save output from program

Jan 18, 2023

Sprenkle - CSCI111

6

6

Review: Lab Expectations

- Comments in programs
 - High-level comments, author
 - Notes for your algorithms, implementation
- Nice, readable, clearly labeled understandable output
 - User running your program needs to **understand** what the program is saying
- Honor System

Jan 18, 2023

Sprenkle - CSC1111

7

7

Other Lab Notes

- Trying to set you up for success now
 - Develop good development habits
 - You know the expectations and how you should develop as programs get larger, more complex
- I won't check your labs before every submission
- Learning how to solve problems
 - Every week: new problems, new techniques to solve problems
- I am explicit in directions/reminders early
 - Then stop reminding because you should know the process later
- Labs are due on Friday; review before the next lab


Jan 18, 2023

Sprenkle - CSC1111

8

8

Parts of an Algorithm

- Input, Output
- Primitive operations
 - What data you have, **what you can do to the data** 
- Naming
 - Identify things we're using
- Sequence of operations
- Conditionals
 - Handle special cases
- Repetition/Loops
- Subroutines
 - Call, reuse similar techniques

Jan 18, 2023

Sprenkle - CSC1111

9

9

More on Arithmetic Operations

Symbol	Meaning	Associativity
+	Addition	Left
-	Subtraction	Left
*	Multiplication	Left
/	Division	Left
%	Remainder ("mod")	Left
**	Exponentiation (power)	Right

Precedence rules: P E - MD% AS

negation

Jan 18, 2023

Sprenkle - CSC1111

10

10

More on Arithmetic Operations

Symbol	Meaning	Associativity
+	Addition	Left
-	Subtraction	Left
*	Multiplication	Left
/	Division	Left
%	Remainder ("mod")	Left
**	Exponentiation (power)	

Precedence rules: P E - MD% AS
negation

Associativity matters when you have the same operation multiple times. It tells you where you should start computing.

Jan 18, 2023

Sprengle - CSC1111

11

Two Division Operators

/ Float Division

- Result is a **float**
- Examples:
 - $6/3 \rightarrow 2.0$
 - $10/3 \rightarrow 3.3333333333333335$
 - $3.0/6.0 \rightarrow 0.5$
 - $19/10 \rightarrow 1.9$

// Integer Division

- Result is an **int**
- Examples:
 - $6//3 \rightarrow 2$
 - $10//3 \rightarrow 3$
 - $3.0//6.0 \rightarrow 0.0$
 - $19//10 \rightarrow 1$

Integer division is the *default* division used in many programming languages

Jan 18, 2023

Sprengle - CSC1111

12

12

Python Division Practice

1. `6.0//12 * 5.0`

2. `12 // 4 * 5.2`

3. `a = 12//5`

4. `b = 6/12`

5. `z = a / b`

Showing a mix of expressions
(just expression and within assignment statements;
integers and floats)

Jan 18, 2023

Sprenkle - CSC1111

14

14

Python Math Practice

1. `5 + 3 * 2`

2. `2 * 3 ** 2`

3. `-3 ** 2`

4. `2 ** 3 ** 3`

Jan 18, 2023

Sprenkle - CSC1111

15

15

Modulo Operator: %

- Modular Arithmetic: Remainder from division
 - $x \% y$ means the remainder of $x//y$
 - Read as “x mod y”
- Example: $6 \% 4$
 - Read as “six mod four”
 - $6//4$ is 1 with a remainder of 2, so $6\%4$ evaluates to 2
- Typical use: only with positive integers
- Precedence rules: P E - MD% AS

Jan 18, 2023

Sprenkle - CSC1111

16

16

Modulo Practice

1. $7 \% 2$

2. $3 \% 6$

3. $6 \% 2$

4. $7 \% 14$

5. $14 \% 7$

6. $6 \% 0$

Jan 18, 2023

Sprenkle - CSC1111

17

17

Brainstorm

- What useful thing does % 10 do?
 - $3 \% 10 =$
 - $51 \% 10 =$
 - $40 \% 10 =$
 - $678 \% 10 =$
 - $12543 \% 10 =$
- What useful thing does // 10 do (integer division)?
 - $3 // 10 =$
 - $51 // 10 =$
 - $40 // 10 =$
 - $678 // 10 =$
 - $12543 // 10 =$
- What useful thing does % 2 do?

Jan 18, 2023

Sprenkle - CSC1111

18

18

Trick: Type Conversion

- You can convert a variable's type
 - Use the type's **constructor**

Conversion Function/Constructor	Example	Value Returned
int(<number or string>)	int(3.77)	3
	int("33")	33
float(<number or string>)	float(22)	22.0
str(<any value>)	str(99)	"99"

Jan 18, 2023

Sprenkle - CSC1111

19

19

Trick: Arithmetic Shorthands

- Called **extended assignment operators**
- Increment Operator
 - $x = x + 1$ can be written as $x += 1$
- Decrement Operator
 - $x = x - 1$ can be written as $x -= 1$
- Shorthands are similar for $*$, $/$, $//$:
 - `amount *= 1.055`
 - `x //= 2`


Jan 18, 2023

Sprenkle - CSC1111

20

20

Parts of an Algorithm

- **Input, Output** 
- Primitive operations
 - What data you have, what you can do to the data
- Naming
 - Identify things we're using
- Sequence of operations
- Conditionals
 - Handle special cases
- Repetition/Loops
- Subroutines
 - Call, reuse similar techniques

Jan 18, 2023

Sprenkle - CSC1111

21

21

Interactive Programs

2.8 in Text Book

- Meaningful programs often need input from users
- Demo: `input_demo.py`

Jan 18, 2023

Spenkle - CSC1111

22

22

Getting Input From User

- `input` is a *function*
 - **Function:** A command to do something
 - A “subroutine”
- Syntax:
 - `input(<string_prompt>)`
- Semantics:
 - Display the prompt `<string_prompt>` in the terminal
 - Read in the user’s input and *return* it as a string/text

Jan 18, 2023

Spenkle - CSC1111

23

23

Getting Input From User

- Typically used in assignments
- Examples:
 - `name=input("What is your name? ")`
 - `name` is assigned the string the user enters
 - `width=eval(input("Enter the width:"))`
 - What the user enters is *evaluated* (as a number) and assigned to `width`
 - Use `eval` function because expect a number from user
 - Alternatively, could use `int` or `float` (conversion functions) instead of `eval`

Jan 18, 2023

Sprenkle - CSC1111

24

24

Getting Input from User

```
color = input("What is your favorite color? ")
```

Semantics: Sets the variable `color` to the user's input

Terminal:

Grabs every character up to the user presses "enter"

```
> python3 input_demo.py
What is your favorite color? blue
Cool! My favorite color is _light_ blue !
```

Jan 18, 2023

Sprenkle - CSC1111

input_demo.py

25

25

Reverse Engineering

Terminal:

```
> python3 input_demo.py
What is your favorite color? blue
Cool! My favorite color is _light_ blue !
```

- Think about what was displayed
- What code was written to make that happen?
 - Typically, we hear “display”, we think “print statement”
 - But, that’s not what was used here because we were displaying a *prompt*

Jan 18, 2023

Sprenkle - CSC1111

input_demo.py

26

26

Identify the Parts of a Program

```
# Demonstrate numeric and string input
# by Sara Sprenkle for CS111
#

color = input("What is your favorite color? ")
print("Cool! My favorite color is _light_", color, "!")

rating = eval( input("On a scale of 1 to 10, how much do
you like Zendaya? ") )
print("Cool! I like her", rating*1.8, "much!")
```

Identify the comments, variables, functions,
expressions, assignments, literals

Jan 18, 2023

Sprenkle - CSC1111

input_demo.py

27

27

Looking Ahead

- Lab 1 due on Friday
- Broader issue write up due Thursday at 11:59 p.m.
- Can practice programming on your own
 - Python visualizer
 - See Resources to download Python and IDLE
 - Login remotely to the lab machines and create a practice directory