

Objectives

- For Loops

1

Lab Review

- Follow examples
 - Find solutions to similar problems
 - Understand the solution
 - Adapt the solution to your problem

Task	Objective
Creating snowperson	Using an API to solve a new problem
Making a picture	Allow you to show your creativity!

2

Review

- How can we find out what we can do to an object?
- What is our *design pattern* for using the graphics library?
- What are the benefits of object-oriented programming (OOP)?
 - This is broader than just the graphics library, which is just one example of OOP

Jan 25, 2022

Sprenkle - CSCI111

3

3

Review: Our Design Pattern for Using the Graphics Library

- Import the Graphics Library
- Create the GraphWin
- Repeat
 - Construct the object
 - May need to construct the objects it needs first
 - Set up its color, width, ...
 - Draw the object
- Call `getMouse` to make the window stay open until the user clicks
- Then, call `close` on the window

Jan 25, 2022

Sprenkle - CSCI111

4

4

Review: Benefits of Object-Oriented Programming

- **Abstraction**
 - Hides details of underlying implementation
 - Easier to change implementation
- Collects related data/methods together
 - Easier to reason about data
- Less code in main program
 - Our program code is relatively simple

Jan 25, 2022

Sprenkle - CSCI111

5

5

Recommendations

- Review the slides, example programs, and/or textbook every day to review what we discussed
 - This problem made sense in class... Does it still make sense?
- Practice a problem every day
 - I rarely use problems from the text book so they're good practice
- Ask questions
- “sense of accomplishment after lab”

Jan 25, 2022

Sprenkle - CSCI111

6

6

FOR LOOPS


Jan 25, 2022

Sprenkle - CSCI111

7

7

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
- Super Power:
Superhuman Speed

Jan 25, 2022

Sprenkle - CSCI111

8

8

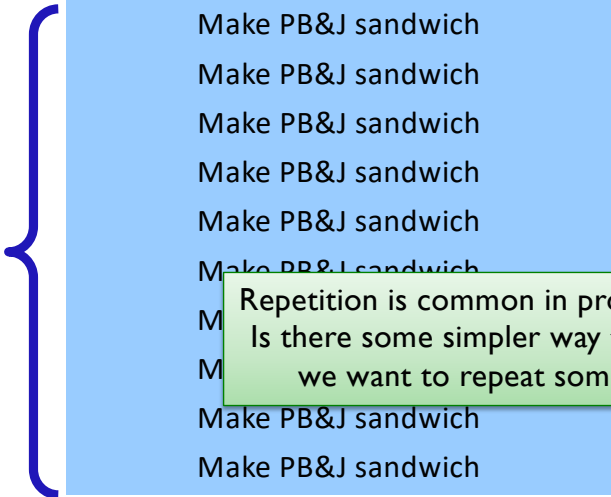
Looping/Repetition

We know how to make a PB&J

Make PB&J sandwich

Sandwich:

Make 10 PB&J sandwiches

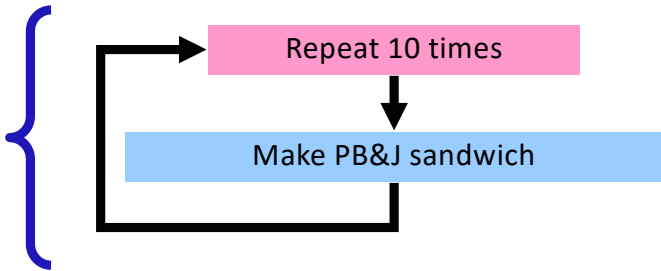


Repetition is common in programming. Is there some simpler way to say that we want to repeat something?

Looping/Repetition

Make PB&J sandwich

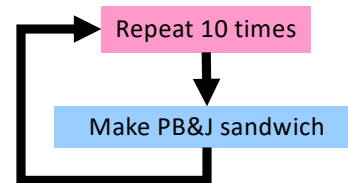
Make 10 PB&J sandwiches



What Goes in the Loop Body?

- Make PB&J Sandwich

1. Gather materials (bread, PB, J, knives, plate)
2. Open bread
3. Put 2 pieces of bread on plate
4. Spread PB on one side of one slice
5. Spread Jelly on one side of other slice
6. Place PB-side facedown on Jelly-side of bread
7. Close bread
8. Clean knife
9. Put away materials



Jan 25, 2022

Sprenkle - CSCI111

11

11

What Goes in the Loop Body?

- Make PB&J Sandwich

1. Gather materials (bread, PB, J, knives, plate)
2. Open bread
3. Put 2 pieces of bread on plate
4. Spread PB on one side of one slice
5. Spread Jelly on one side of other slice
6. Place PB-side facedown on Jelly-side of bread
7. Close bread
8. Clean knife
9. Put away materials

Initialization

Loop Body

Finalization

Jan 25, 2022

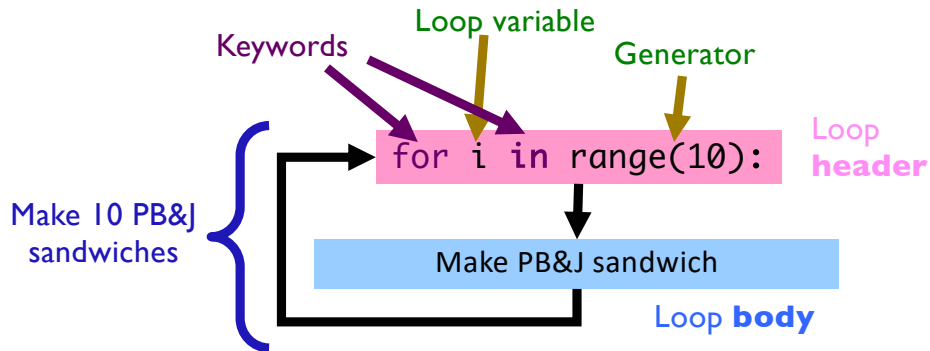
Sprenkle - CSCI111

12

12

The **for** Loop

- Use when know how many times loop will execute
 - Repeat N times



Jan 25, 2022

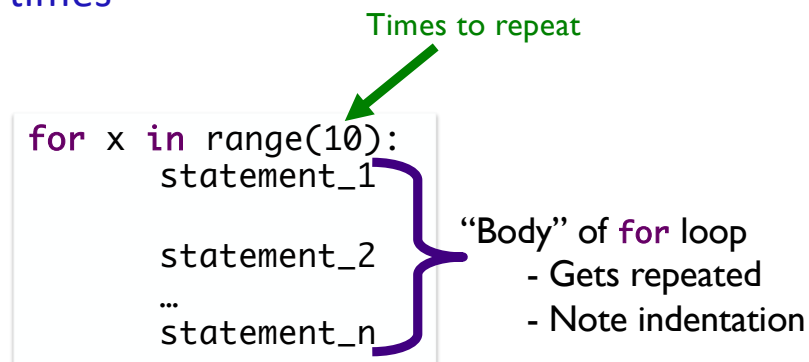
Sprenkle - CSCI111

13

13

for Loop Syntax and Semantics

- Use when know how many times loop will execute
 - Repeat N times



Jan 25, 2022

Sprenkle - CSCI111

14

14

Analyzing `range()`

- `range` is a *generator*
- What does `range` do, exactly, with respect to the loop variable `i`?

```
for i in range(5):  
    print(i)  
  
print("After the loop:", i)
```

`range_analysis.py`

Jan 25, 2022

Sprenkle - CSCI111

16

16

for loop analysis

```
for i in range(5):  
    # like assigning i values(0,1,2,3,4)  
    # consecutively, each time through loop  
  
    # rest of loop body ...
```

- When we have `range(5)`,
 - `i` is set to the values (0, 1, 2, 3, 4)
 - Which means that loop executes 5 times
- Optional: start and step parameters

Jan 25, 2022

Sprenkle - CSCI111

17

17

`range([start,] stop[, step])`

- `[xxx]` means that xxx is optional
- 1 argument: `range(stop)`
- 2 arguments: `range(start, stop)`
- 3 arguments: `range(start, stop, step)`

Jan 25, 2022

Sprenkle - CSCI111 [using_range.py](#)

18

18

`range([start,] stop[, step])`

- 1 argument: `range(stop)`
 - Defaults: `start = 0, step = 1`
 - Iterates from 0 to `stop-1` with step size=1
- 2 arguments: `range(start, stop)`
 - Default: `step = 1`
 - Iterates from `start` to `stop-1` with step size=1
- 3 arguments: `range(start, stop, step)`
 - Iterates from `start` to `stop-1` with step size=`step`

Jan 25, 2022

Sprenkle - CSCI111 [using_range.py](#)

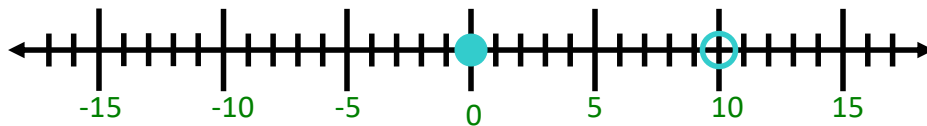
19

19

range

● range is a number generator

- 1 argument: `range(stop)`
- 2 arguments: `range(start, stop)`
- 3 arguments: `range(start, stop, step)`



[start, stop)

`range(10)`
`range(0, 10)`
`range(0, 10, 1)`

Jan 25, 2022

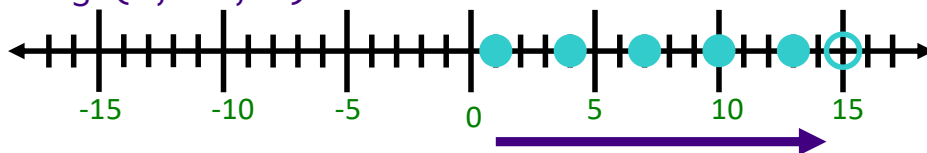
Sprenkle - CSCI111

20

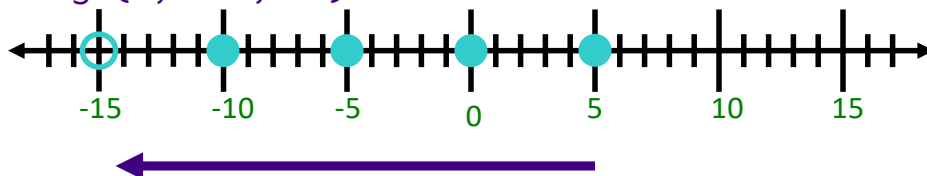
20

Sequence generated by range

`range(1, 15, 3):`



`range(5, -15, -5):`



`more_range_examples.py`

Jan 25, 2022

Sprenkle - CSCI111

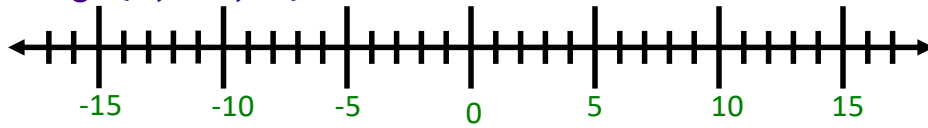
21

21

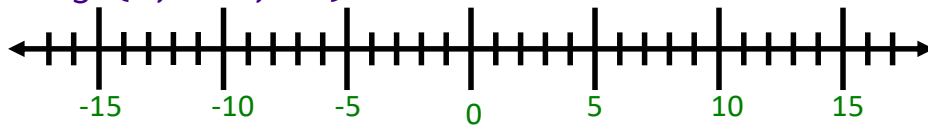
Practice

Place these: ● ○
Which direction?

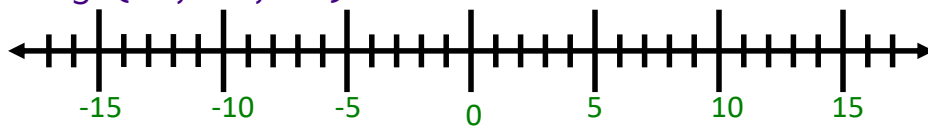
`range(2, 14, 2):`



`range(8, -10, -3):`



`range(-5, 15, -3):`



Jan 25, 2022

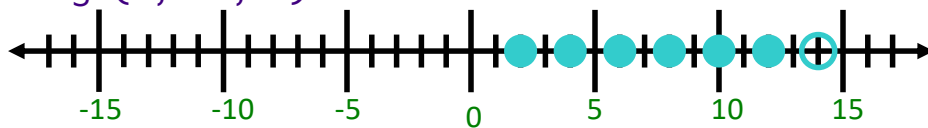
Sprenkle - CSCI111

22

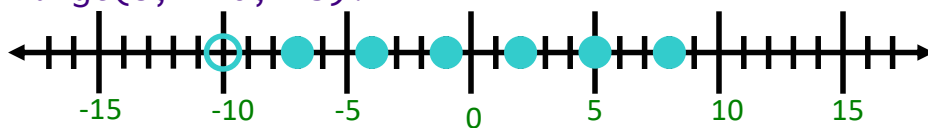
22

Practice Solution

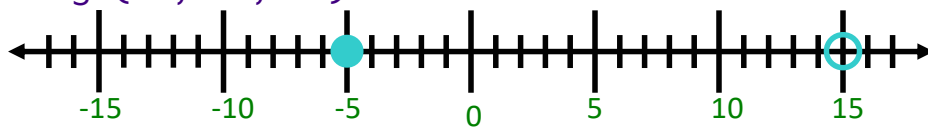
`range(2, 14, 2):`



`range(8, -10, -3):`



`range(-5, 15, -3):`



Jan 25, 2022

Sprenkle - CSCI111

23

23

Practicing **for** Loops

- Write the Python code to display the following:

➤ A) 1
2
3
4
5

➤ B) 2
5
8
11

➤ C) ****

Questions to ask:

- What is getting repeated?
- How many times?

How do the answers to those questions inform your solution?

Jan 25, 2022

Sprengle - CSCI111

24

24

Process of Solving Loop Problems

- What is getting repeated?
 - Informs what goes in the *loop body*
- How many times?
 - Informs what the arguments to `range` should be

Jan 25, 2022

Sprengle - CSCI111

25

25

Programming Practice

- Add 5 numbers, inputted by the user
- After implementing, simulate running on computer
 - You can pretend to be the computer

Jan 25, 2022

Sprenkle - CSCI111

`sum5.py`

26

26

This Week

- Lab 2 – Friday
- Broader Issue due Thursday night

Jan 25, 2022

Sprenkle - CSCI111

27

27