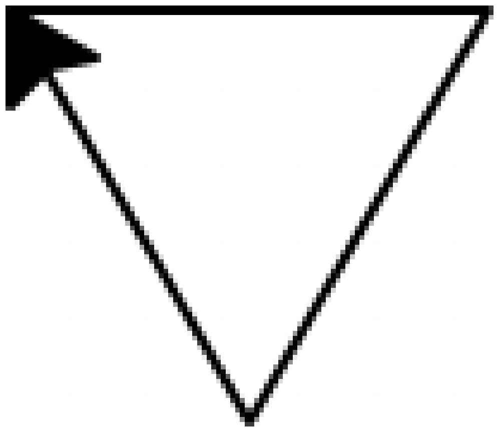
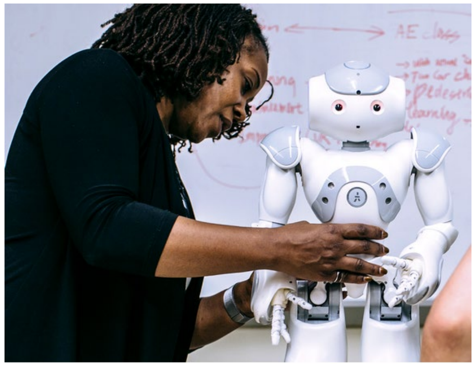
Compsci 101
Turtle, Bagels, Loop Tracing, Files
Live Lecture

Susan Rodger
Nicki Washington
February 23, 2021



Ayanna Howard

- Educator, Researcher and Innovator
- Professor & Chair of the School of Interactive Computing, Georgia Tech
- Now Dean of Engineering at The Ohio State University!
- Robotics – Robots and Bias, Robots changing lives of children with disabilities, Robots beyond part of the family
- Top 50 U.S. Women in Tech, Forbes, 2018



I believe that every engineer has a responsibility to make the world a better place. We are gifted with an amazing power to take people's wishes and make them a reality.

Announcements

- APT-2 due tonight!
  - Remember you get 24-hour grace period, can't turn in after that!
- APT-3 out today – due 3/2
- Assignment 2 Turtles out – due 3/4

- Lab 4 Friday – No Prelab

APT Quiz 1 coming…

- APT Quiz 1 is 3/5-3/8
- Open around 8am 3/5
- There are two parts
- Pick a start time for each part,
  - Once you start a part, You have 1.5 hours
  - If you get accommodations, you get those
- 4 APTs to solve (2 in each part)

- Will put up problems from an old APT Quiz so you can practice

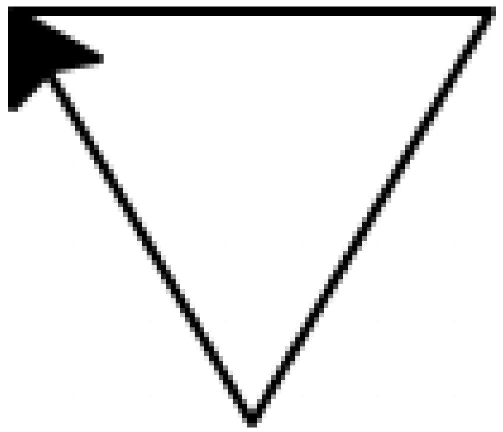
WOTO-1 – Turtles Simple
http://bit.ly/101s21-0223-1

WOTO-2: Let’s draw a triangle! http://bit.ly/101s21-0223-2

- Equilateral triangle
  - Corner degrees: 60
  - Side length: 100

WOTO-2: Let's draw a triangle! http://bit.ly/101s21-0223-2

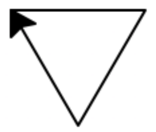
- Equilateral triangle
  - Corner degrees: 60
  - Side length: 100



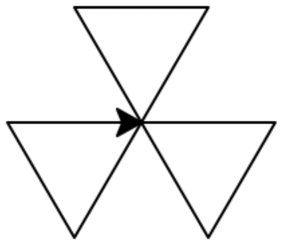
WOTO-2: Let's draw it 3 times!

Orientation and location matters!

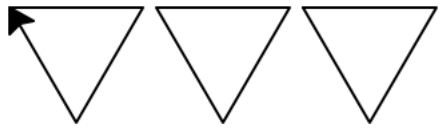
- **TPS: What will the turtle draw?**
  - Note: Think about where the turtle is and facing after each iteration



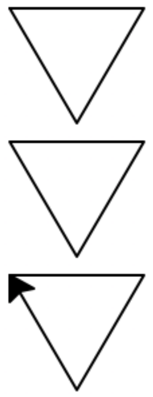
Option 1 (Draw 3 triangles on top of each other)



Option 2



Option 3

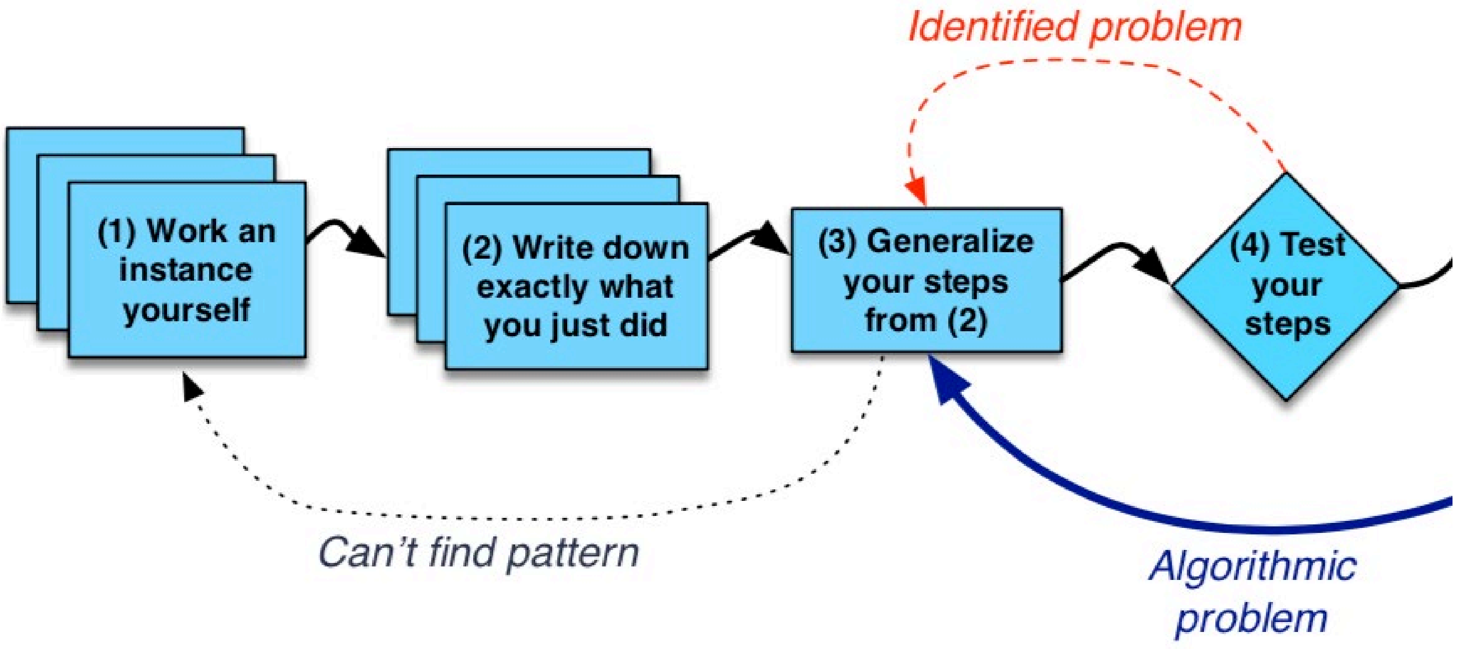


Option 4

Bagels (Accumulation)

APT Bagels

- How figure out how many bagels needed?
  - 7-steps!


Identified problem
(1) Work an instance yourself
(2) Write down exactly what you just did
(3) Generalize your steps from (2)
(4) Test your steps
Can't find pattern
Algorithmic problem

APT: Bagel Counting

Problem Statement

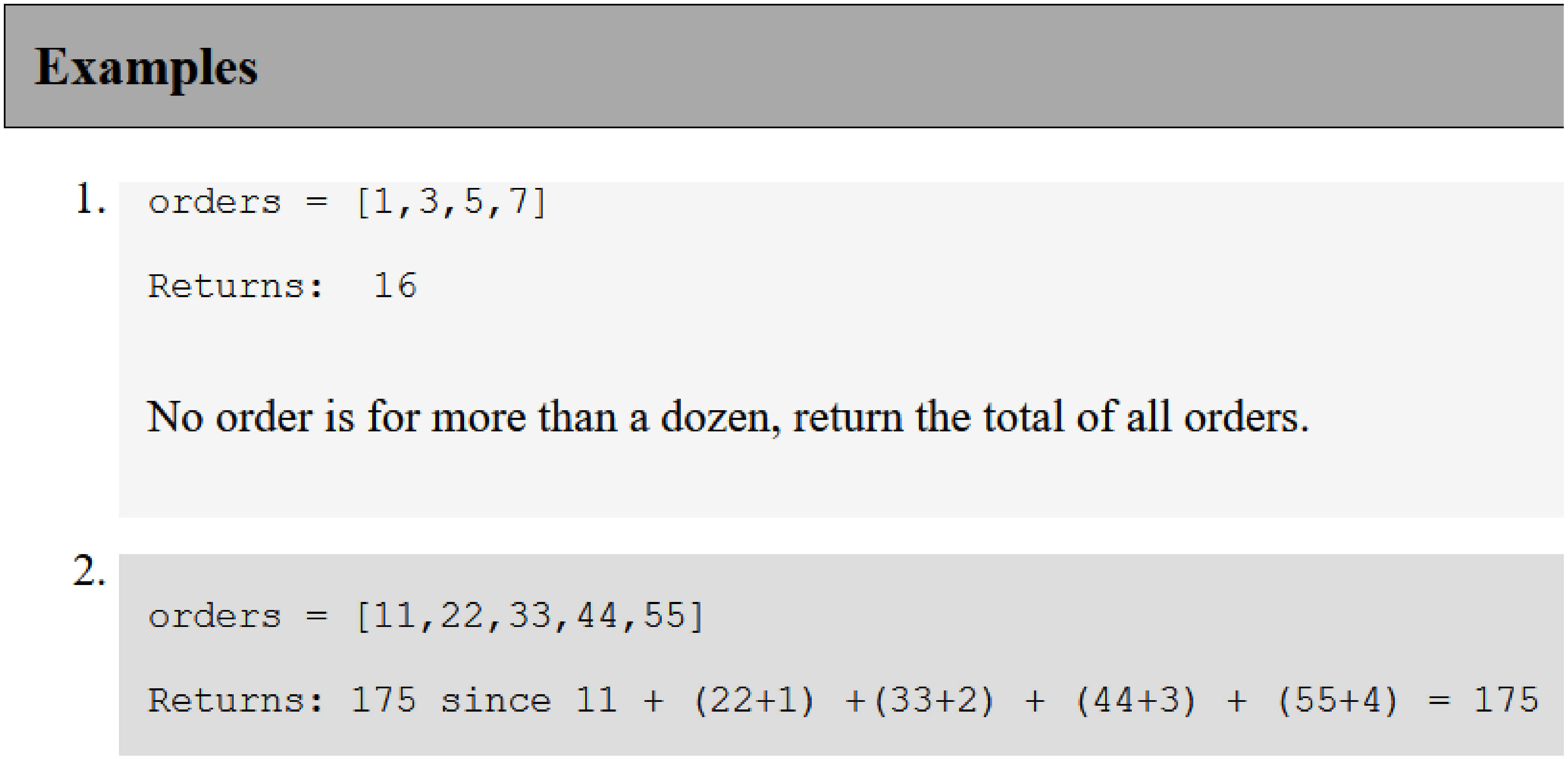
Class

```
filename:  Bagels.py

def bagelCount(orders) :
    """
    return number of bagels needed to fulfill
    the orders in integer list parameter orders
    """

    # you write code here
```

You are in charge of web-based orders for your neighborhood bagel store, *The Bagel Byte*. Each evening you must total the orders to be picked up the next day. Some orders are simply for *N* bagels, but each order of a dozen or more bagels is topped off with an extra bagel, the so-called "baker's dozen". This means, for example, that an order for 25 bagels actually requires 27 bagels to fulfill since there are two extra bagels needed for each dozen in the order. An order for 11 bagels doesn't require any extra since it's for less than a dozen.

Given a list of integers representing bagel orders determine the number of bagels needed to fulfill all the orders.

Examples



Examples

1.
```
orders = [1,3,5,7]

Returns:  16
```

No order is for more than a dozen, return the total of all orders.

2.
```
orders = [11,22,33,44,55]

Returns: 175 since 11 + (22+1) +(33+2) + (44+3) + (55+4) = 175
```

Step 1 and 2

- Step 1: Solve an instance (think)
  - orders = [11, 3, 24, 17]

Step 1 and 2

- **Step 1: Solve an instance (think)**
  - orders = [11, 3, 24, 17]
  - 11 + 3 + (24+2) + (17+1) = 58
  - Total: 58
- **Step 2: What did we do?**
  - Write down in words

WOTO-3 Step 3: Generalize http://bit.ly/101s21-0223-3

WOTO-3 Step 3: Generalize

- Go through list
    - If less than 12
        - Do nothing
    - If greater than or equal to 12
        - Add however many times 12 goes into the order
- Sum everything

Step 4: Test steps

- Go through list
  - If less than 12
    - Do nothing
  - If greater than or equal to 12
    - Add however many times 12 goes into the order
- Sum everything

- [11, 22, 33, 44, 55]
- 11
  - Nothing (less than 12)
- 22
  - +1
- 33
  - +2
- 44
  - +3
- 55
  - +4
- Sum: 175

Step 5: Code

- Go through list
  - If less than 12
    - Do nothing
  - If greater than or equal to 12
    - Add however many times 12 goes into the order
- Sum everything

- for loop!
- if statement

- Think: if's or if…else statement?
  - floor div: //

Could we use the accumulator pattern?

Yes!

Step 5: Code

```
def bagelCount(orders):
    """
    return number of bagels needed to fulfill
    the orders in integer list parameter orders
    """
    total = 0
    for order in orders:
        if order < 12:
            total += order
        else:
            extra = order // 12
            total += order + extra

    return total
```

Initialize before loop

Update inside loop

Do something with value after loop

Code-Tracing a Loop

1. Find the changing variables/expressions
2. Create table, columns are variables/expressions
    1. First column is loop variable
    2. Add columns to help track everything else
3. Each row is an iteration of the loop
    1. *Before* execute code block, copy down each variable’s value
    2. Execute code block, update a value in the row as it changes

WOTO-4 Loop Tracing http://bit.ly/101s21-0223-4

- Remember the steps
- (1) Find the changing variable/expressions,
- (2) Create the table with these as the column
- (3) Each row is an iteration of the loop