SCC461 – Programming for Data Scientists

Leandro Marcolino

Lancaster University

Week 7

Outline

Peer Feedback Exercise

- WAIT FOR ALL INSTRUCTIONS BEFORE YOU START MOVING!
- You will work in pairs
- Discuss your CW 6 with your partner
- Ask your partner what he/she is struggling with, and teach him/her
- Similarly, tell your partner what you are struggling with, and he/she will teach you
- If you are new to programming, find an experienced programmer as a pair
- If you are an experienced programmer, find someone that is new to programming as a pair

Outline

What will be printed?

```
a = 2;

b = a;

b = 999;

print(a);
```

What will be printed?

```
a = [1, 2, 3, 4, 5];

b = a;

b[1] = 999;

print(a);
```

What will be printed?

```
a = [1, 2, 3, 4, 5];

b = a[:];

b[1] = 999;

print(a);
```

What will be printed?

```
def callMe(input):
    input = 1;
    return;

myInput = 0;
callMe(myInput);
print(myInput);
```

What will be printed?

```
def callMe(input):
    input[0] = 1;
    return;

myList = [0];
callMe(myList);
print(myList[0]);
```

Outline

Abstractions

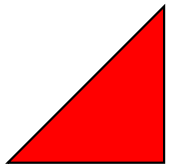
What is a triangle?

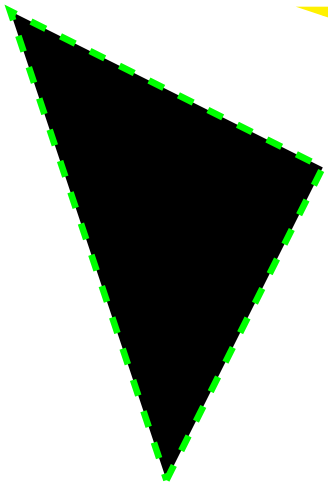
Abstractions

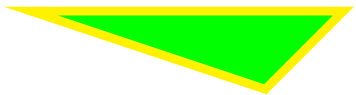
Triangle: The plane figure formed by connecting three points not in a straight line by straight line segments; a three-sided polygon. *American Heritage Dictionary of the English Language, Fifth Edition*

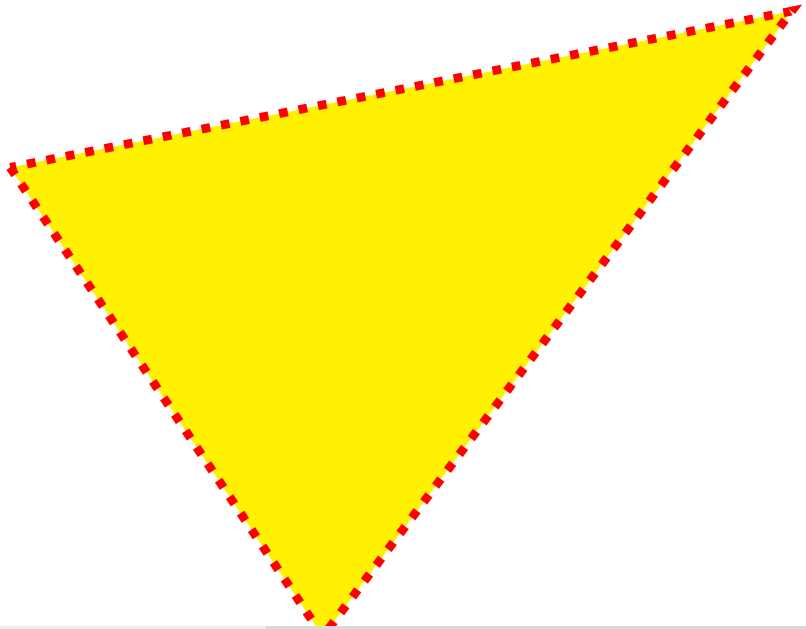
Abstractions

Triangle: The plane figure formed by connecting three points not in a straight line by straight line segments; a three-sided polygon. *American Heritage Dictionary of the English Language, Fifth Edition*





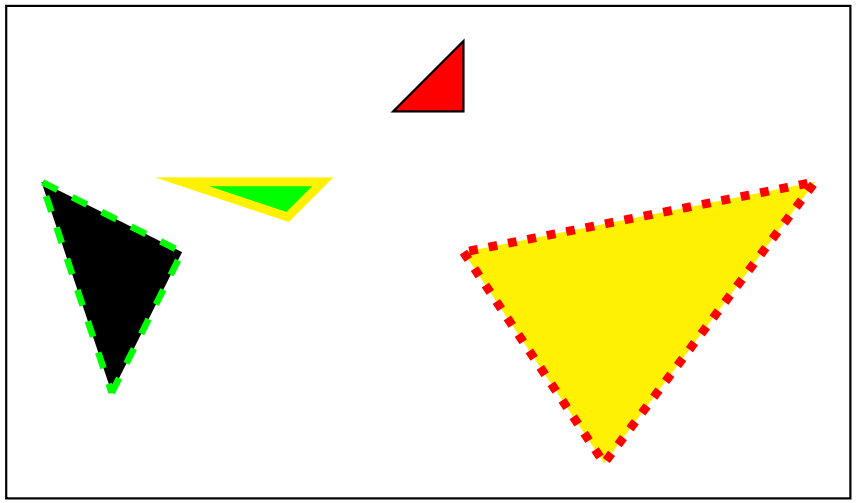




Abstractions

The plane figure formed
by connecting three
points not in
a straight line
by straight line segments;
a three-sided polygon.

Abstraction



Instantiation

Abstractions

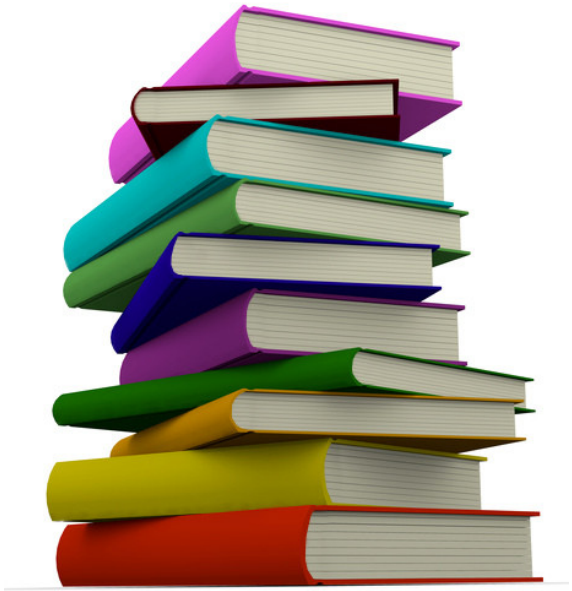
What is a stack?

Abstractions

Stack: An orderly pile, especially one arranged in layers. *American Heritage Dictionary of the English Language, Fifth Edition*

Abstractions

Stack: An orderly pile, especially one arranged in layers. *American Heritage Dictionary of the English Language, Fifth Edition*



Interface

- How to draw a triangle?
- How to remove an element from a stack?

Interface


AMOUNT DEPOSITED
EXACT CHANGE ONLY
CHECK SELECTION PRICE
MAKE ANOTHER SELECTION
COIN RETURN
INSERT COINS
M&M's
HERSHEY'S
Milk Chocolate
Choco Cremes
OREO
Combos
Grandma's
Teddy Grahams
Honey
BITES
Cheetos
CHEESE FIX
Lay's
Classic

Interface

Outline

Classes

```
class Point:
    """ Point class represents and manipulates
        x,y coords. """

    def __init__(self):
        """ Create a new point at the origin """
        self.x = 0;
        self.y = 0;
```

Objects

```
p = Point();              # Instantiate an object of
    type Point
q = Point();              # Make a second point

print(p.x, p.y, q.x, q.y);  # Each point object
    has its own x and y
```

Objects

```
p = Point();              # Instantiate an object of
    type Point
q = Point();              # Make a second point

p.x = 3;
p.y = 4;

print(p.x, p.y, q.x, q.y);  # Each point object
    has its own x and y
```

Constructor

```
class Point:
    """ Point class represents and manipulates
        x,y coords. """

    def __init__(self, x=0, y=0):
        """ Create a new point at x, y """
        self.x = x
        self.y = y

# Other statements outside the class continue
    below here.
```

Constructor

```
p = Point(4, 2);
q = Point(6, 3);
r = Point();          # r represents the origin (0,
    0)
print(p.x, q.y, r.x);
```

Methods

```
class Point:
    """ Create a new Point, at coordinates x, y
        """

    def __init__(self, x=0, y=0):
        """ Create a new point at x, y """
        self.x = x
        self.y = y

    def distance_from_origin(self):
        """ Compute my distance from the origin
            """
        return ((self.x ** 2) + (self.y ** 2))
            ** 0.5
```

Objects as Arguments

```
def print_point(pt):
    print("("+str(pt.x)+", " + str(pt.y) + ")")
```

Exercise

Program 1:

```
def x(pt):
    pt.x = -1;
    print(pt.x);

point = Point();

point.x = 5;

x(point);

print(point.x);
```

Program 2:

```
def x(pt):
    pt = -1;
    print(pt);

pt = 5;

x(pt);

print(pt);
```

What will be printed?

Exercise

Program 3:

```
def x(pt):
    pt.x = -1;
    print(pt.x);

pt1 = Point();
pt2 = Point();

pt1.x = 5;
pt1 = pt2;

x(pt2);
print(pt1.x);
```

What will be printed?

Exercise

```
Program 4:

def x(pt):
    pt.x = -1;
    print(pt.x);

pt = Point();
pt2 = Point();

pt2.x = 5;

x(pt2);

print(pt.x);
```

What will be printed?

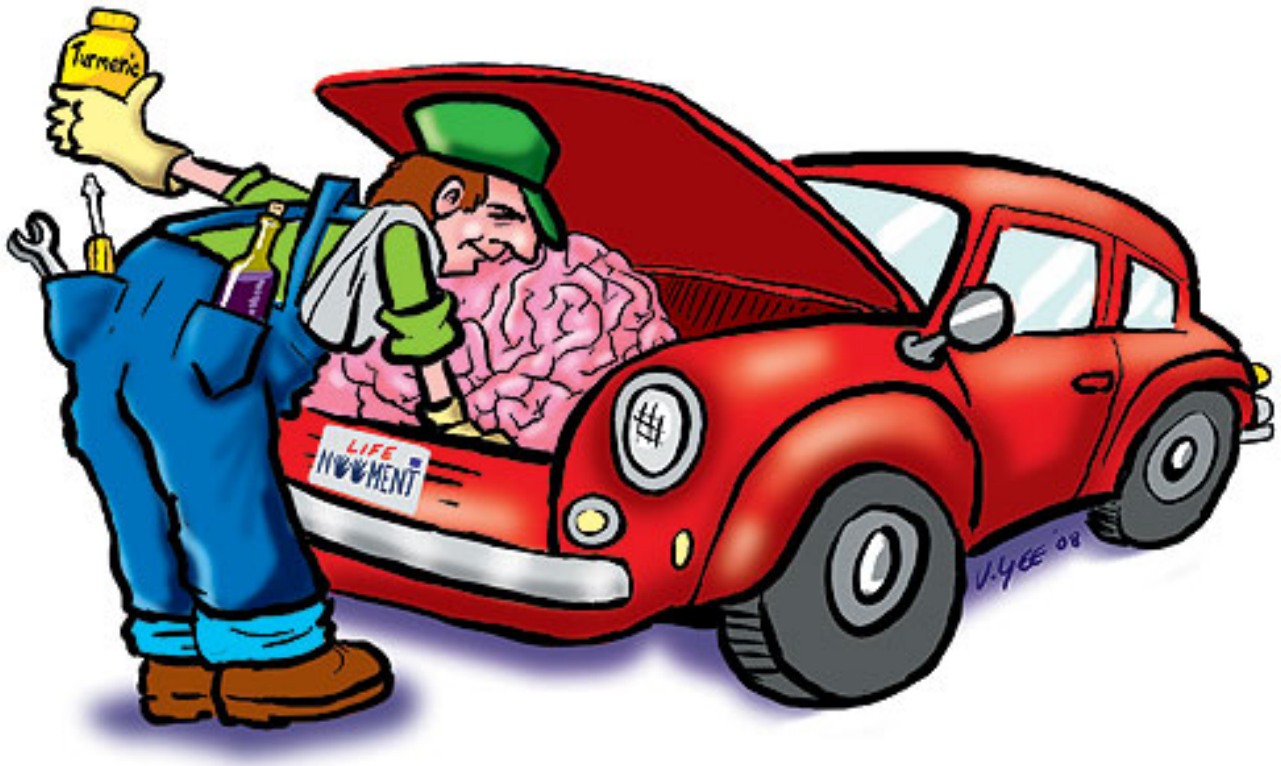
Exercise

Exercise

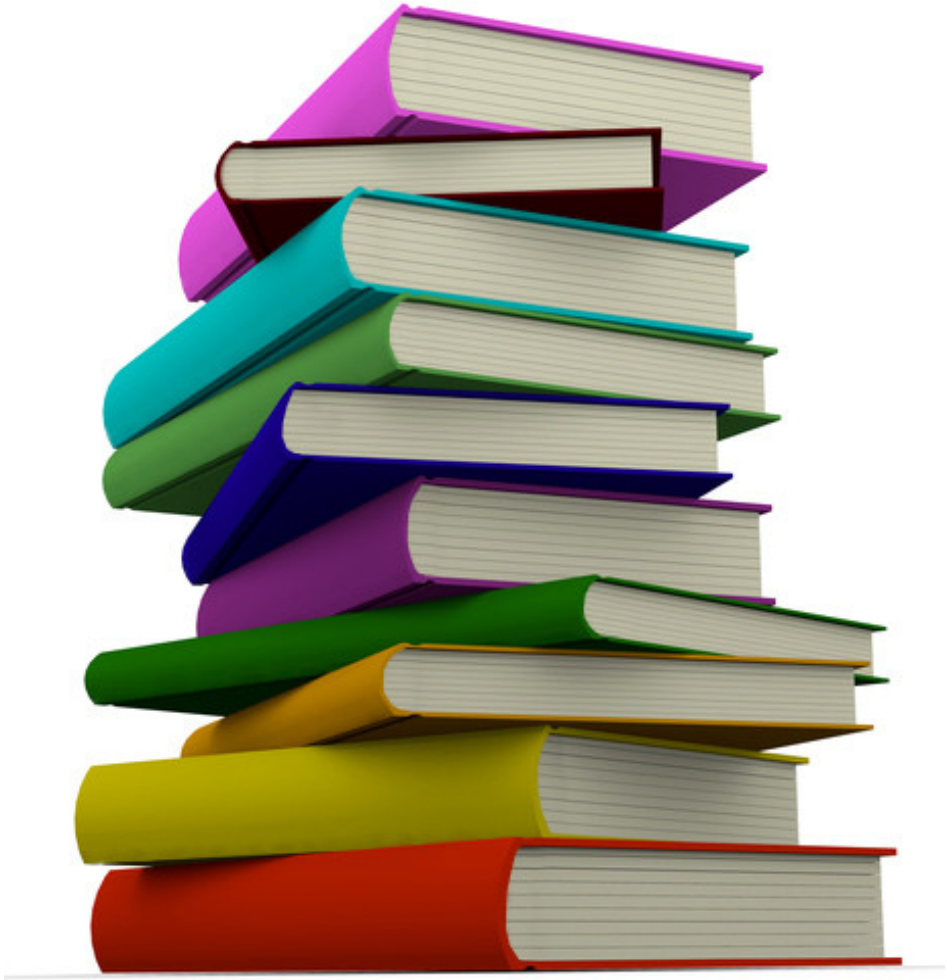
Write a method (inside Point class) that allows a Point to be copied.

Data Structures

- Stack
- Linked List
- Queue



Stack



Stack: Last in, first out!

Stack

Stack: A collection of items, with four methods:

- **Push:** Adds an item to the top of the Stack.
- **Pop:** Removes the top-most item of the Stack and returns it.
- **Clear:** Removes all items of the Stack.
- **Is Emtpy:** Returns whether the Stack is empty or not.

Exercise

Exercise

- Implement a Stack class.
- Test your implementation creating Stack objects, and pushing and popping items.

Fixed Stack

Exercise

- Implement a Stack class that can hold at most 10 items.
- You are not allowed to use list methods.
- You can: create a list, and change the value of specific elements

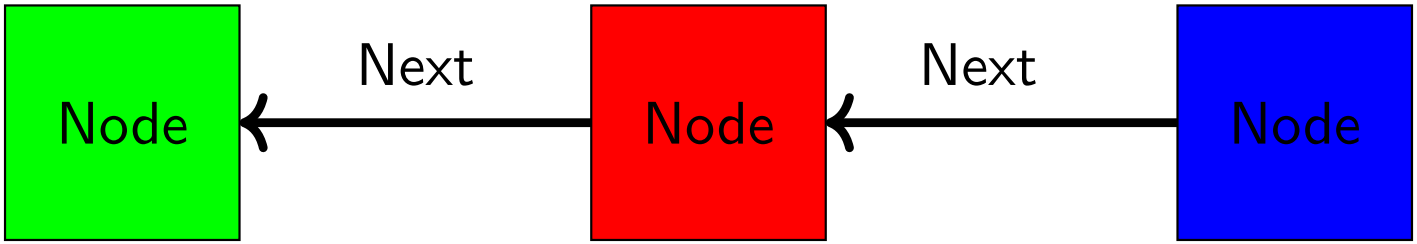
Growing Stack

Exercise

- Implement a Stack that automatically grows in size as items are pushed in.
- You are not allowed to use list methods.
- You can: create a list, and change the value of specific elements

Linked List

Linked List: A node points to the next node



Linked List

```
class Node:
    def __init__(self, content=None, next=None):
        self.content = content
        self.next    = next
```

Linked List

Example

```
node = Node("test")
print(node)
```

Linked List

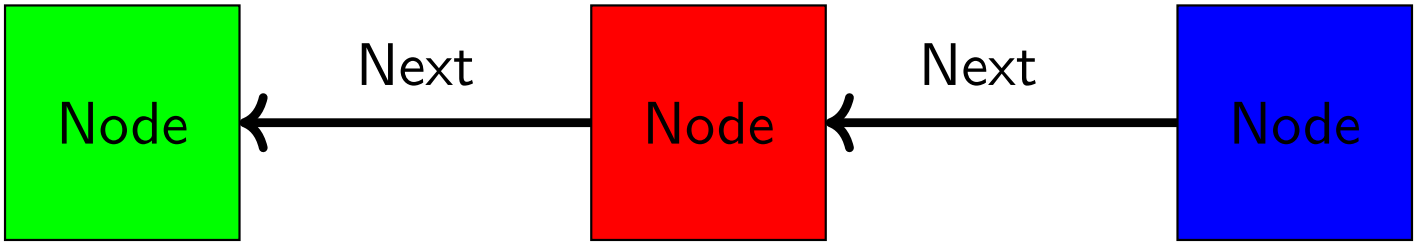
Example

```
node1 = Node(1)
node2 = Node(2)
node3 = Node(3)

node1.next = node2
node2.next = node3
```

Linked List

Linked List: A node points to the next node



Printing a Linked List

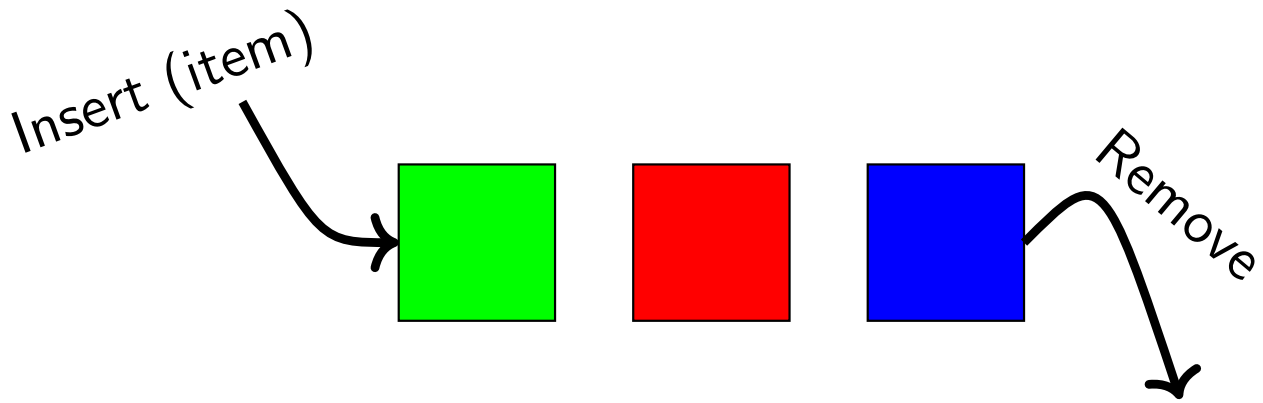
```
def print_list(node):
    while node is not None:
        print(node, end=" ")
        node = node.next
    print()

print_list(node1)
```

Output: 1 2 3

Queue

Queue: First in, first out!



Queue

Queue: A collection of items, with four methods:

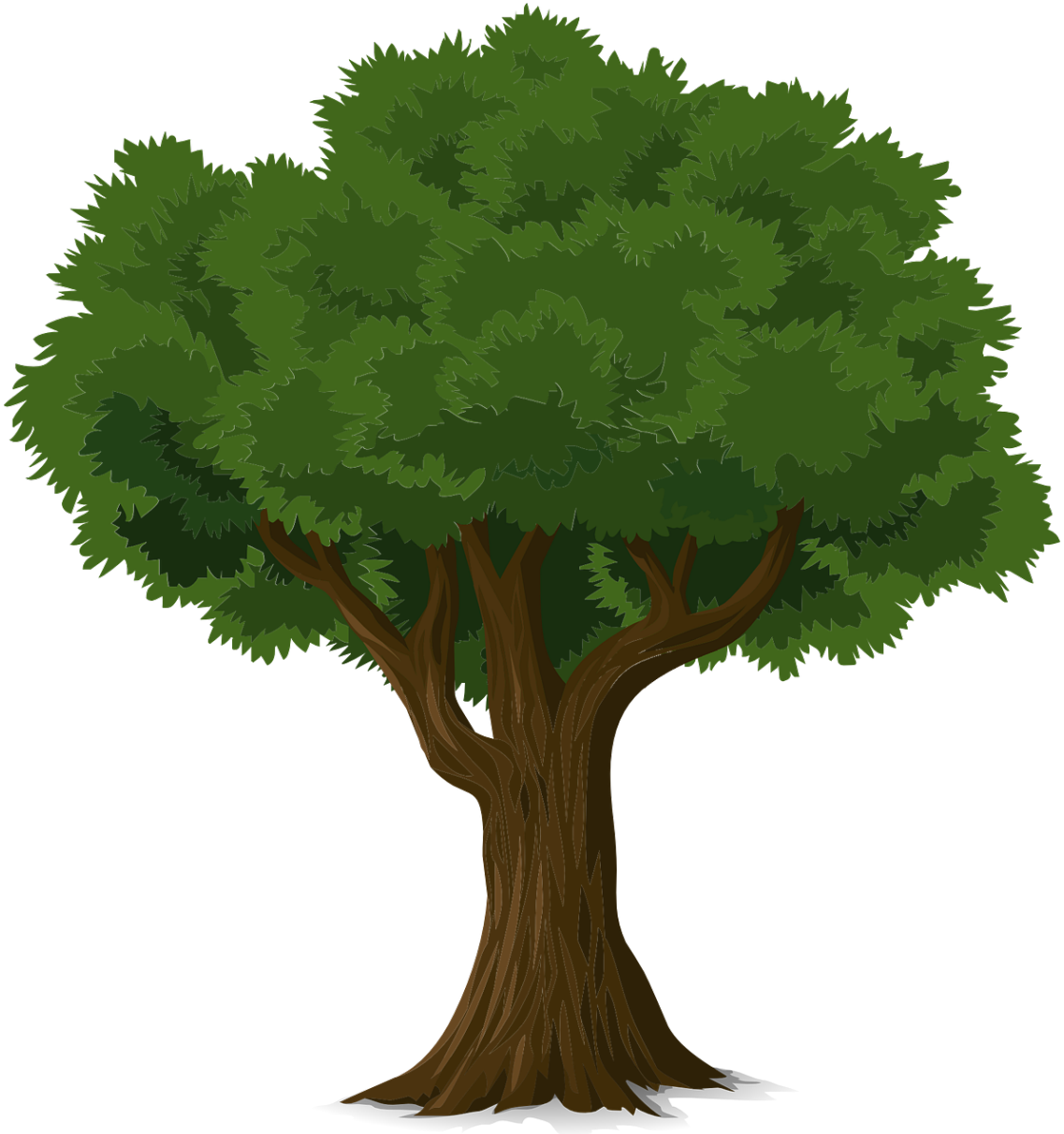
- **Insert:** Adds an item to the end of the Queue.
- **Remove:** Return the item that is in the Queue for the longest time, and remove the item.
- **Clear:** Removes all items of the Queue.
- **Is Empty:** Returns whether the Queue is empty or not.

Queue

Exercise

- Implement a Queue class, using Linked Lists

Why are we learning Stack, Queue and Linked List?



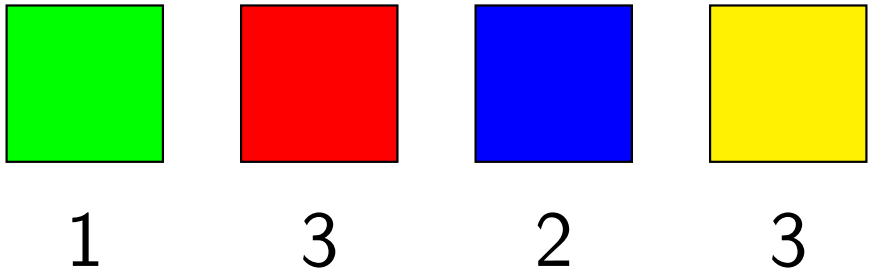
Outline

Assignment – Part 1

1. **FibonacciQueue** (2%):
   - Create and test a FibonacciQueue class, for storing integers
     - A FibonacciQueue works just like a Queue, except that:
     - In the FibonacciQueue, the xth item removed will be multiplied by the xth number in the Fibonacci sequence
     - You must use the Linked List technique.

Priority Queue

- Each item has an associated priority
- Items are removed from highest to lowest priority (if two items have the same priority, the first one to be inserted is removed first)



Assignment – Part 2

2. Create and test a PriorityQueue class. (2%)
   - Numbers are input in pairs: the first one is the actual item, the second is the item's priority
   - You must use the Linked List technique.
3. Which previous systems inspired Alan Kay in the creation of OOP programming, and how did they inspire him? (1%)

Python

The language reference is your friend!

Python

References

- https://docs.python.org/3/
- http://openbookproject.net/thinkcs/python/english3e/

Turtles!

http://openbookproject.net/thinkcs/python/english3e/hello_little_turtles.html

Thank you!

l.marcolino@lancaster.ac.uk
http://www.lancaster.ac.uk/staff/sorianom/