Simple Java YEAH Hours

Brahm Capoor

What are YEAH hours?

Held soon after each assignment is released

Help you to get an early start on your assignments

Future dates TBA

Recorded and slides will be posted!

Roadmap

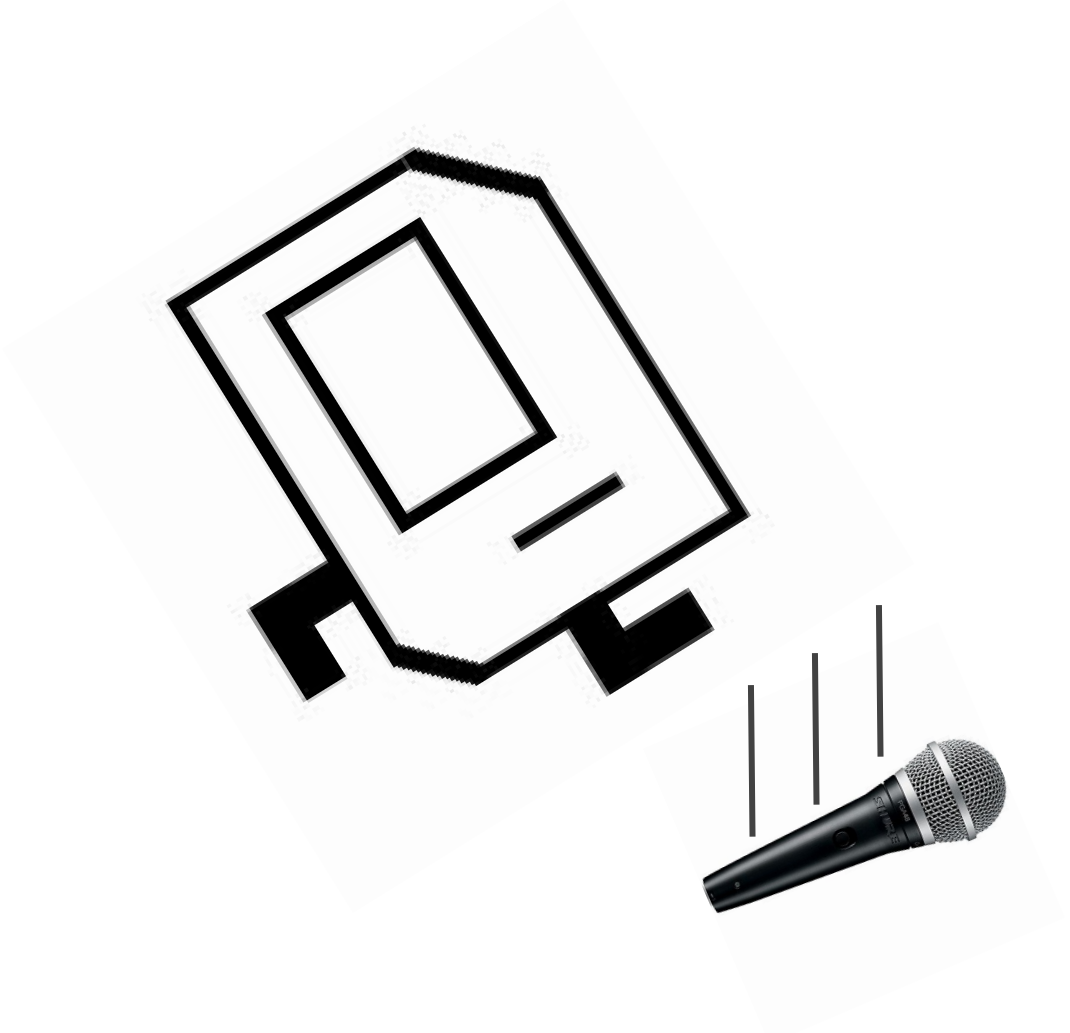
Review

Assignment overview and tips

Questions

Dropping the mic on Karel

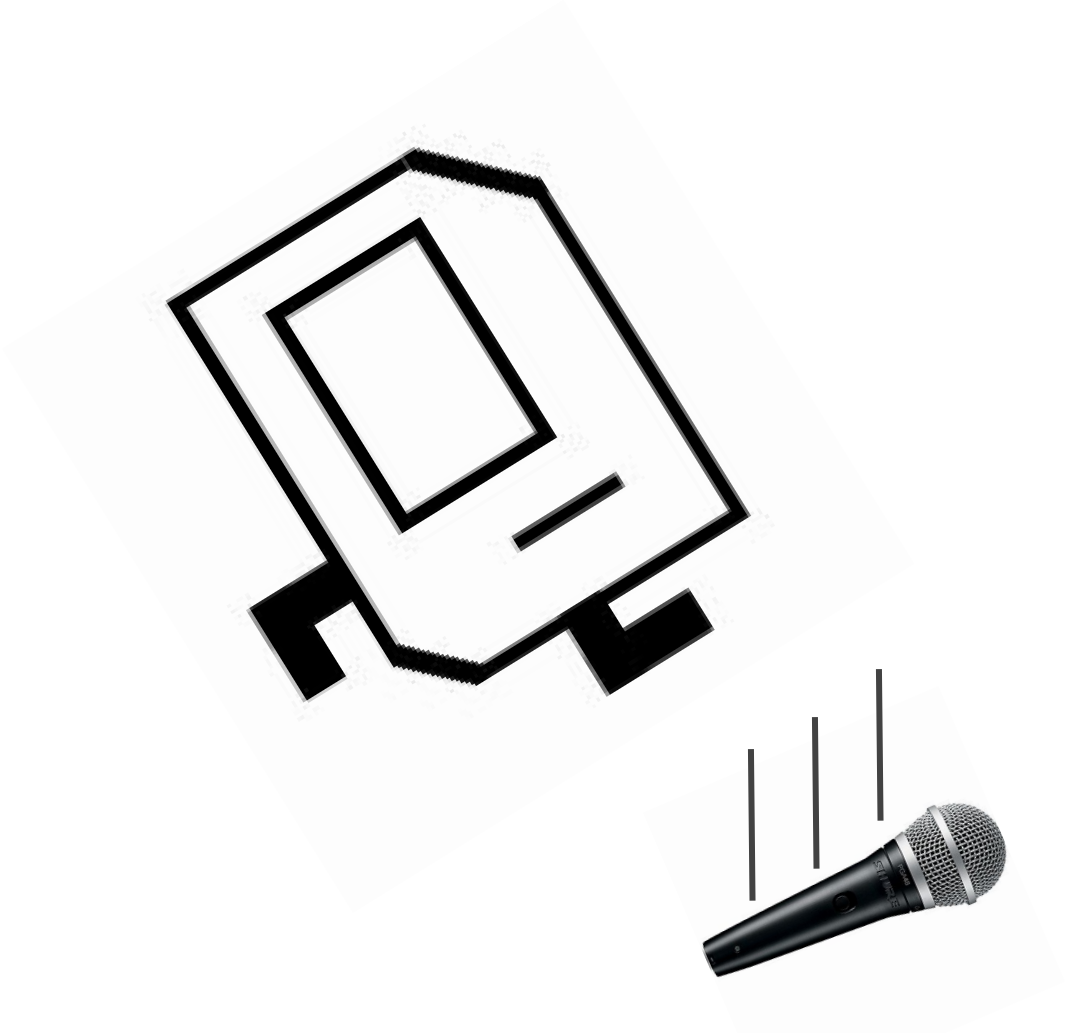
Karel taught us a lot of things!



Dropping the mic on Karel

Karel taught us a lot of things!

Control Flow

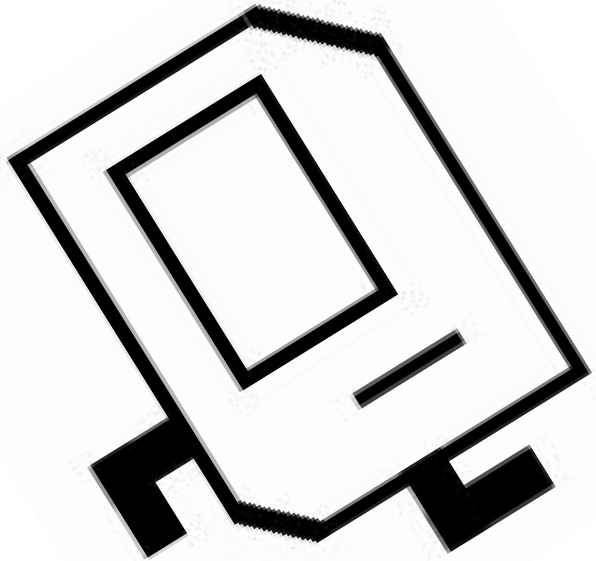


Dropping the mic on Karel

Karel taught us a lot of things!

Control Flow

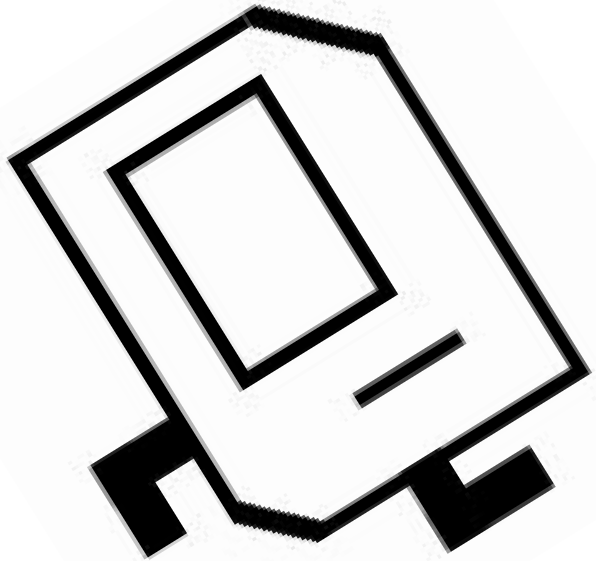
Decomposition & Top Down Design



Dropping the mic on Karel

Karel taught us a lot of things!

- Control Flow
- Decomposition & Top Down Design
- Algorithmic Strategy



Control Flow in Karel

```
for (int i = 0; i < 5; i++) {      // do whatever is in the loop 5 times
     if (beepersPresent()) {       // what to do if a particular condition is true
          move();
     } else {                      // what to do if that condition is false
          putBeeper();
     }
}

while (frontIsClear()) {          // do this until a particular condition is false
     move();
     putBeeper();
}
```

Control Flow outside Karel

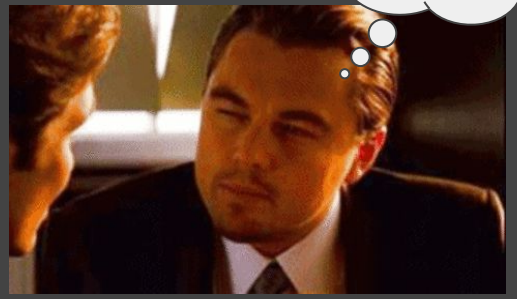
```
for (int i = 0; i < 100; i++) {    // do whatever is in the loop 100 times
     if (i % 2 == 0) {             // what to do if a particular condition is true
          println("Even: " + i);
     } else {                      // what to do if that condition is false
          println("Odd: " + i);
     }
}

while (true) {                     // loop indefinitely
     if (agentOfChaos()) {
          break;                   // ~~savagely~~ immediately end while loop
     }
     print("Good prevails!");
}
```

Control Flow-ception

```
for (int i = 0; i < 10; i++) {
     for (int j = 0; j < 10; j++) {
          if (i == j) {
               println(“i and j are equal!”);
          } else {
               int difference = i - j;
               if (difference > 0) {
                    println(“i is bigger than j by ” + difference + “!”);
               } else {
                    println(“j is bigger than i by ” + difference + “!”);
               }
          }
     }
}
```

Control Flow-ception




```
for (int i = 0; i < 10; i++) {
     for (int j = 0; j < 10; j++) {
          if (i == j) {
               println(“i and j are equal!”);
          } else {
               int difference = i - j;
               if (difference > 0) {
                    println(“i is bigger than j by ” + difference + “!”);
               } else {
                    println(“j is bigger than i by ” + difference + “!”);
               }
          }
     }
}
```

Control Flow-ception

```
for (int i = 0; i < 10; i++) {
     for (int j = 0; j < 10; j++) {
          if (i == j) {
               println(“i and j are equal!”);
          } else {
               int difference = i - j;
               if (difference > 0) {
                    println(“i is bigger than j by ” + difference + “!”);
               } else {
                    println(“j is bigger than i by ” + difference + “!”);
               }
          }
     }
}
```

Graphics

```
GRect rect = new GRect(50, 50, 200, 200);
rect.setFilled(true);
rect.setColor(Color.BLUE);

GOval oval = new GOval(0, 0, getWidth(), getHeight());
oval.setFilled(false);
oval.setColor(Color.GREEN);

GLabel text = new GLabel("banter", 200, 10);

add(text);
add(rect);
add(oval);
```

Graphics

```
GRect rect = new GRect(50, 50, 200, 200);
rect.setFilled(true);
rect.setColor(Color.BLUE);

GOval oval = new GOval(0, 0, getWidth(), getHeight());
oval.setFilled(false);
oval.setColor(Color.GREEN);

GLabel text = new GLabel("banter", 200, 10);

add(text);
add(rect);
add(oval);
```

Things to remember

- Coordinates are `doubles`
- Coordinates are measured from the top left of the screen
- Coordinates of a shape are coordinates of its top left corner
- Coordinates of a label are coordinates of its bottom left corner
- Remember to `add` objects to the screen!
- Use the online documentation!
- These are class variables!

Primitive variables

```
int x = 7;   // declare and initialize a variable
x = 9;       // change the value of x
x = x + 1;   // increment (add 1 to) x.  A.K.A. x++
x = x + 2;   // add 2 to x.             A.K.A. x += 2
x /= 2       // divide x by 2, and truncate result

double d = 3.5;

boolean isThisTrue = true;
isThisTrue = !isThisTrue;  // flip isThisTrue
```

Primitive variables

```
int x = 7;    // declare and initialize a variable
x = 9;        // change the value of x
x = x + 1;    // increment (add 1 to) x.  A.K.A. x++
x = x + 2;    // add 2 to x.              A.K.A. x += 2
x /= 2        // divide x by 2, and truncate result

double d = 3.5;

boolean isThisTrue = true;
isThisTrue = !isThisTrue;  // flip isThisTrue
```

Things to remember

- The expressive hierarchy:
  `boolean < char < int < double`
- Compare variables using ==
  ```
  if (x == 7) {
  ```
- Conditional operators: && and ||
  ```
  if (x == 7 && y == 6.3)
  if (x == 7 || x == 6)
  ```
  Avoid this:
  ```
  if (x == 7 || 6)
  ```
- Use constants!

```
private static final int MY_NUM = 10;
```

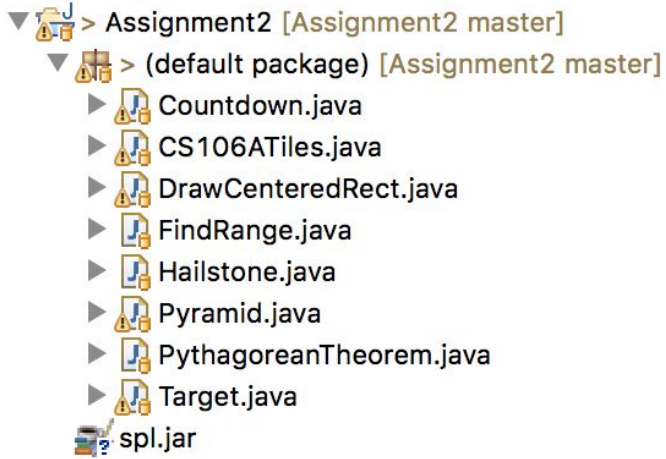
Assignment 2!

You can do all of it right now

You'll learn tools to improve your style for some problems after Wednesday's lecture

High level overview

- Due Monday 29/1/2017
- 8 Problems
- 2 warmups
- 3 Graphics Programs
- 3 Console Programs



Problem 1 Draw a blue, filled rectangle in the center of the screen with dimensions 350 x 270

– – –

Questions to ask yourself:

1. How do I find the center of the screen?
2. Given the location of the center of the screen, where should I put the rectangle?

Useful ideas from lecture

- Coordinates are measured from the top left of shapes and the window

Useful methods:

- `getWidth()` tells you the width of the canvas
- `getHeight()` tells you the height of the canvas
- `rect.getWidth()` tells you the width of `rect`
- `rect.getHeight()` tells you the height of `rect`
- See lecture/video and `GRect` documentation for more!

Problem 2 Print out a countdown down from 10 to 1 and then print “Liftoff!”

#rocketscience

– – –

Questions to ask yourself:

1. What sort of control flow structure best suits this problem?
2. What’s a nice way to represent what the current number is?

Useful ideas from lecture

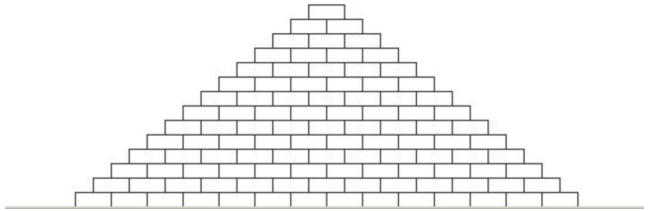
- You can use the variables inside `for` loops!

Problem 3 Draw a pyramid!

– – –

Questions to ask yourself:

1. What sort of control flow structure best suits this problem?
2. How do I decompose this problem?
3. What information do I need to draw a row and the bricks inside a row?



Useful ideas from lecture

- You can use the variables inside `for` loops!
- You can nest `for` loops!
- This checkerboard example from lecture

Useful methods

- `getWidth()` tells you the width of the canvas
- `getHeight()` tells you the height of the canvas
- `rect.getWidth()` tells you the width of `rect`
- `rect.getHeight()` tells you the height of `rect`
- See lecture/video and `GRect` documentation for more!

** remember that coordinates should be `double`s

Problem 4 Bullseye!

Questions to ask yourself:

1. Can this problem be decomposed?
2. What information is needed to draw each circle?

Useful ideas from lecture

- Stay tuned for Wednesday's lecture to better decompose the problem!

Useful methods

- See lecture/video and `GOval` documentation for more!

Problem 5 CS 106A Tiles

Questions to ask yourself:

1. Can this problem be decomposed?
2. What information is needed to draw each rectangle?

Useful ideas from lecture

- Stay tuned for Wednesday's lecture to better decompose the problem!
- Remember that a label's coordinate is its bottom left corner

Useful methods

- `label.getAscent()` tells you the distance between the baseline of the label and the top of the label. This is useful for centering!
- See lecture/video and `GRect` documentation and `GLabel` documentation for more!

Problem 6 Pythagorean Theorem

Questions to ask yourself:

1. What data type should I store numbers as?
2. How many variables do I need?

```
Enter values to compute the Pythagorean theorem.
a: 3.5
b: 4.2
c = 5.4671747731346585
```

Useful ideas from lecture

- Primitive data types
- The expressive hierarchy

Useful methods

- `math.sqrt(n)` tells you the square root of `n`
- Look at the lecture for more!

Problem 7 Keeping track of the largest and smallest

Questions to ask yourself:

1. What sorts of things do you need to store?
2. How do you initialize variables?

```
This program finds the largest and smallest numbers.
? 11
? 17
? 42
? 9
? -3
? 35
? 0
smallest: -3
largest: 42
```

Useful ideas from lecture

- Loop structures
- Variable scope
- Edge cases
- Sentinel values

Problem 8 Hailstone sequence

Questions to ask yourself:

1. What sorts of things do you need to store?
2. How do you initialize variables?

```
Enter a number: 17
17 is odd, so I make 3n + 1: 52
52 is even so I take half: 26
26 is even so I take half: 13
13 is odd, so I make 3n + 1: 40
40 is even so I take half: 20
20 is even so I take half: 10
10 is even so I take half: 5
5 is odd, so I make 3n + 1: 16
16 is even so I take half: 8
8 is even so I take half: 4
4 is even so I take half: 2
2 is even so I take half: 1
The process took 12 to reach 1
```

Useful ideas from lecture

- Loop structures
- Variable scope
- Edge cases
- Sentinel values

A last few tips and tricks

- “Write a `GraphicsProgram` SubClass”: Don’t worry about what this means! (You’ll learn a lot about this in a few weeks)
- Draw things on paper for Graphics Programs
- Use Top Down Decomposition wherever you can
- Go to the LaIR!
- Incorporate your IG feedback!
- Use the debugger!

Questions?