for loops and methods
Announcements

• Quiz 1 on Friday
  – Review today. 5:00. CENTR 212
Assume there’s a method “drawRandomFruit()”
How would you create this:
While loops

```java
final int DIMENSION = 9;
int row = 0;
int col = 0;
while (row < DIMENSION) {
    while (col < DIMENSION) {
        drawRandomFruit();
        col++;
    }
    nextLine();
    col=0;
    row++;
}
```
While loops

```
final int DIMENSION = 9;
int row = 0;
int col = 0;
while (row < DIMENSION) {
    while (col < DIMENSION) {
        drawRandomFruit();
        col++;
    }
    nextLine();
    col=0;
    row++;
}
```
for loops

```java
final int DIMENSION = 9;
for (int row = 0; row < DIMENSION; row++) {
    for (int col = 0; col < DIMENSION; col++) {
        drawRandomFruit();
    }
    nextLine();
}
```

for (start; condition; update) {
    code;
}

- Start: int row = 0;
- Condition: row < DIMENSION;
- Update: row++
while vs. for loops

• When to use one vs. other?
Choose the best loop for each situation:

1) Loop through all pixels in a picture and set each pixel’s red value equal to its green value
2) Loop until a user enters a stop condition, e.g. “quit”
3) Loop until two variables match each other, e.g. $x == y$, where both $x$ and $y$ are changed in the loop body
Summary

• for
  – Use when you know number of iterations to loop

• while
  – Use when unsure how many times to loop
Methods
Methods
GTA 5 Menu

switch (selection) {
    case SINGLE PLAYER:
        // 3 million lines of code;
        break;
    case MULTI PLAYER:
        // 5 million lines of code;
        // much of it duplicated from above;
        break;
    case YOU GET THE POINT:
}
switch (selection) {
    case SINGLE_PLAYER:
        singlePlayer();
        break;
    case MULTIPLAYER:
        multiPlayer();
        break;
    case YOU_GET_THE_POINT:
}
methods

• Already have experience calling methods
  - println(), next(), drawC()

• Already have experience defining a method
  - public static void main(String[] args) { }
  - private static void drawS(int x, int y) { }
Methods in math

\[ f(x) = x + 3; \]

Function name: \( f \)
Input: \( x \)
Output: \( x+3 \)
Method definition

```java
public static void main(String[] args) { }
```

- **public static** — we’ll cover later in the class, for now assume it’s required
- **void** — what does the method return. Can be a primitive type, String, or nothing (void)
- **main** — the name of the method. “main” is special
- **(String[] args)** — the parameters. What gets passed into the method
public class Test {
    public static void newMethod(int i) {
        System.out.println("Value: "+i);
        return;
    }

    public static void main(String[] args) {
        int x = 10;
        newMethod(5);
        newMethod(x-1);
    }
}
Calling methods & Scope

- Variables can only be “seen” inside the method that defines them
  - Global variables exist
- When you call method, parameter types must match

```java
public class Test {
    public static int everyone = 10;
    public static void newMethod(int i, String s, boolean b) {
        boolean test2 = !b;
        int alpha = 1;
        return;
    }
    public static void main(String[] args) {
        boolean test1 = true;
        char c = 'A';
        newMethod(3, "adam", test);
    }
}
```
Calling methods & Scope

• Variables can only be “seen” inside the method that defines them
  – Global variables exist
• When you call method, parameter types must match

```java
public class Test {
    public static int everyone = 10;

    public static void newMethod(int i, String s, boolean b) {
        boolean test2 = !b;
        int alpha = 1;
        return;
    }

    public static void main(String[] args) {
        boolean test1 = true;
        char c = 'A';
        newMethod(3, "adam", test);
    }
}
```
• Can only return nothing or 1 thing.
  – And type must match the method definition
• Can have multiple return statements, but only 1 will be executed

```java
public static float calculateArea(float radius) {
    float result = 3.14 * radius * radius;
    return result;
}

public static void main(String[] args) {
    float area = calculateArea(2);
}
```
parameters

• Like a variable definition
  – Memory is allocated and argument’s value assigned
  – Parameter can only be used within the scope of the method
  – Even if no parameters needed, must have ()
  – Only pass-by-value
public class Test {
    public static void newMethod(int i) {
        i++;
        return;
    }

    public static void main(String[] args) {
        int i = 10;
        newMethod(i);
        System.out.println(i);
    }
}

A) 10
B) 11
C) Compiler Error
D) Runtime error
E) I’m lost
public class Test {
    public static int confuse = 0;
    public static void newMethod(int i) {
        i++;
        confuse++;
        return;
    }

    public static void main(String[] args) {
        confuse++;
        newMethod(confuse);
        System.out.println(confuse);  
    }
}