2nd lesson

Before Starting

- Record meeting
- Prerequisite courses - Add Coursera libraries:
  - Download courseraJars.zip and uncompress it
  - File → Project preferences → Modules → Dependencies → "+" sign → Jars or directories → Select all jars within the downloaded zip

```
import edu.duke.StorageResource;

StorageResource genes;
```

- https://www.w3schools.com/java/ is good place for checking/learning basic things
- More in depth: https://docs.oracle.com/javase/tutorial/java/javaOO/index.html

Concepts for today

- Loops
- Methods
- Classes???

Loops

A loop is used for executing a block of code several times depending on a condition.

For

https://www.w3schools.com/java/java_for_loop.asp

When you know exactly how many times you want to loop through a block of code, use the `for` loop:

```
for (<initialization>; <loop condition>; <loop update>) {
    // code block to be executed
}
```

e.g.

```
for (int i=0; i<10; i++) {
    System.out.println(i)
}
```

Note: Be aware that there are other kinds of for to iterate through collections. We will see them in the future when we see

collections. In case you want to take a look https://www.geeksforgeeks.org/for-each-loop-in-java/

While

https://www.w3schools.com/java/java_while_loop.asp

Mainly used when you do not know how many iterations you will do

```
while (condition) {
    // code block to be executed
}
```

e.g.

```
int i=0;
while(i<10){
   System.out.println(i);
   i++;
}
```

“We have a list of numbers. We do not know the length of the valuable data but we know it end is marked with a five, so we do not want to print it nor the numbers coming afterwards.”

```
int[] numbers = {1,2,-3,21,7,5,1,1,1,1,1,1};
int index = 0;
int currentValue = numbers[0];

while(currentValue != 5){
   System.out.println(currentValue);

   index++;
   currentValue = numbers[index];
}
```

Do-While

Almost the same as while, but the code within the “do” is executed the first time without checking the condition.

```
do {
    // code block to be executed
} while (condition);
```

e.g.

```
int i=0;
do {
```

```
    System.out.println(i);
    i++;
} while(i<0);
```

```
int[] numbers = {1,2,-3,21,7,5,1,1,1,1,1,1};
int index = 0;
int currentValue;

do{
    currentValue = numbers[index]
    System.out.println(currentValue);

    index++;
}while(currentValue != 5)
```

break

Exits the loop.

```
int i=0;
while(i<10){
    if(i == 5){
        break;
    }
    System.out.println(i);
    i++;
}
```

continue

Skips the current iteration of the loop.

```
int i=0;
while(i<10){
    if(i == 5){
        continue;
    }
    System.out.println(i);
    i++;
}
```

Methods

https://www.w3schools.com/java/java_methods.asp

Also known as functions. Used to encapsulate and reuse code.

```
[access_modifier] [static] <return_type> <name of the function>(<parameters>) {
                <body of the function>
}
```

e.g.

```
String greetPerson(String prefix, String name) {
    String greeting = "Hello" + prefix + " " + "name";
    return greeting;
}
```

```
greetPerson("Mr.", "Pablo")
```

```
String getMonthName(int month) {

    String monthString;
    switch (month) {
        case 1:  monthString = "January";
                 break;
        case 2:  monthString = "February";
                 break;
        case 3:  monthString = "March";
                 break;
        case 4:  monthString = "April";
                 break;
        case 5:  monthString = "May";
                 break;
        case 6:  monthString = "June";
                 break;
        case 7:  monthString = "July";
                 break;
        case 8:  monthString = "August";
                 break;
        case 9:  monthString = "September";
                 break;
        case 10: monthString = "October";
                 break;
        case 11: monthString = "November";
                 break;
        case 12: monthString = "December";
                 break;
        default: monthString = "Invalid month";
                 break;
```

```
    }

    return monthString;
}
```

If a method does not return anything, its return type should be void. e.g.

```
void helloWorld() {
    System.out.println(“Hello World!”);
}
```

Value vs Reference

```
void increase(int number) {
    number += 1;
    System.out.println("in: " + number);
}
```

```
int x = 0;
increase(x);
System.out.println("out: " + x);
```

The output will be
in: 1
out: 0

When changing contexts, simple types are passed by value. This means that the value is only modified in the current scope, no changes are applied to the original variable outside of it.

What is recommended: return the modified value:

```
int increase(int number) {
    number += 1;
    return number;
}
```

NOTE: This is different for classes. If you pass a class to a method and you modify the data of the class within the method, changes will still be present after exiting the method.

Classes

https://www.w3schools.com/java/java_oop.asp

Classes are used in Object Oriented Programming (OOP). They help structure problems in a way that is closer to reality.

Classes contain attributes (data) and methods (logic). They are named with their first letter in uppercase.

Classes vs Objects

Similar to idea/abstraction vs actual object. e.g. Cars in general vs my car parked in my garage.

```
public class Car {

    String plate;
    String color;
    String brand;
    String model;
    int cv;

    // Constructor
    public Car(String plate, String color, String brand, String model, int cv) {
        this.plate = plate;
        this.color = color;
        this.brand = brand;
        this.model = model;
        this.cv = cv;
    }

    public String getColor() {
        return color;
    }

    public void setColor(String color) {
        this.color = color;
    }
}
```

Creating an instance of a class

```
Car myCar = new Car("123456", "red", "Ford", "Fiesta 2001", 80);
```

Regular methods and attributes defined in a class belong to the instances created. To access them use “.”

```
myCar.plate
```

```
myCar.getColor()
```

They can be accessed or not outside of the class depending on the access level modified of that attribute or method.

Access Levels

Modifier	Class	Package	Subclass	World
public	Y	Y	Y	Y
protected	Y	Y	Y	N
no modifier	Y	Y	N	N
private	Y	N	N	N

Static methods or attributes belong to the class instead of the instance.

```
// static method example
public static String getAllBrands() {
    return new String[]{"Volvo", "Ford", "Subaru", ...}
}

// static attribute example
public static String allBrands = new String[]{"Volvo", "Ford", "Subaru", ...}
```

To use them, the class is leveraged instead of an instance:

```
Car.getAllBrands()
Car.allBrands
```

Exercises

Track assistance

http://www.beginwithjava.com/java/loops/questions.html

Tips

Get user input in console:

```
Scanner console = new Scanner(System.in);
int num;

System.out.print("Enter any positive integer: ");
num = console.nextInt();
```

```
System.out.println("Number entered " + num);
```

You can get other input types (e.g String) by changing `console.nextInt()` to a different method (e.g. `console.nextLine()`). Check the suggestions you get from IntelliJ as they will include the return type.