CORE JAVA

Pilot
Anna

Features of Java:

- Encapsulation
- Inheritance
- Polymorphism

IntelliJ

JVM: Java Virtual Machine
JRE: Java Runtime Environment

Main method

- entry point for executing a Java program

- **public** - JRE can access and execute this method.
- **static** - so that the JVM can load the class into memory and call the main method without creating an instance of the class first.
- **void** - it doesn't return anything. When the main method is finished executing, the Java program terminates
- **String[] args** - Each string in the array is a command line argument. And they can be used to pass information to the program, at runtime.

Variable vs Constant

- Constant is a data item whose value cannot change during the program's execution.

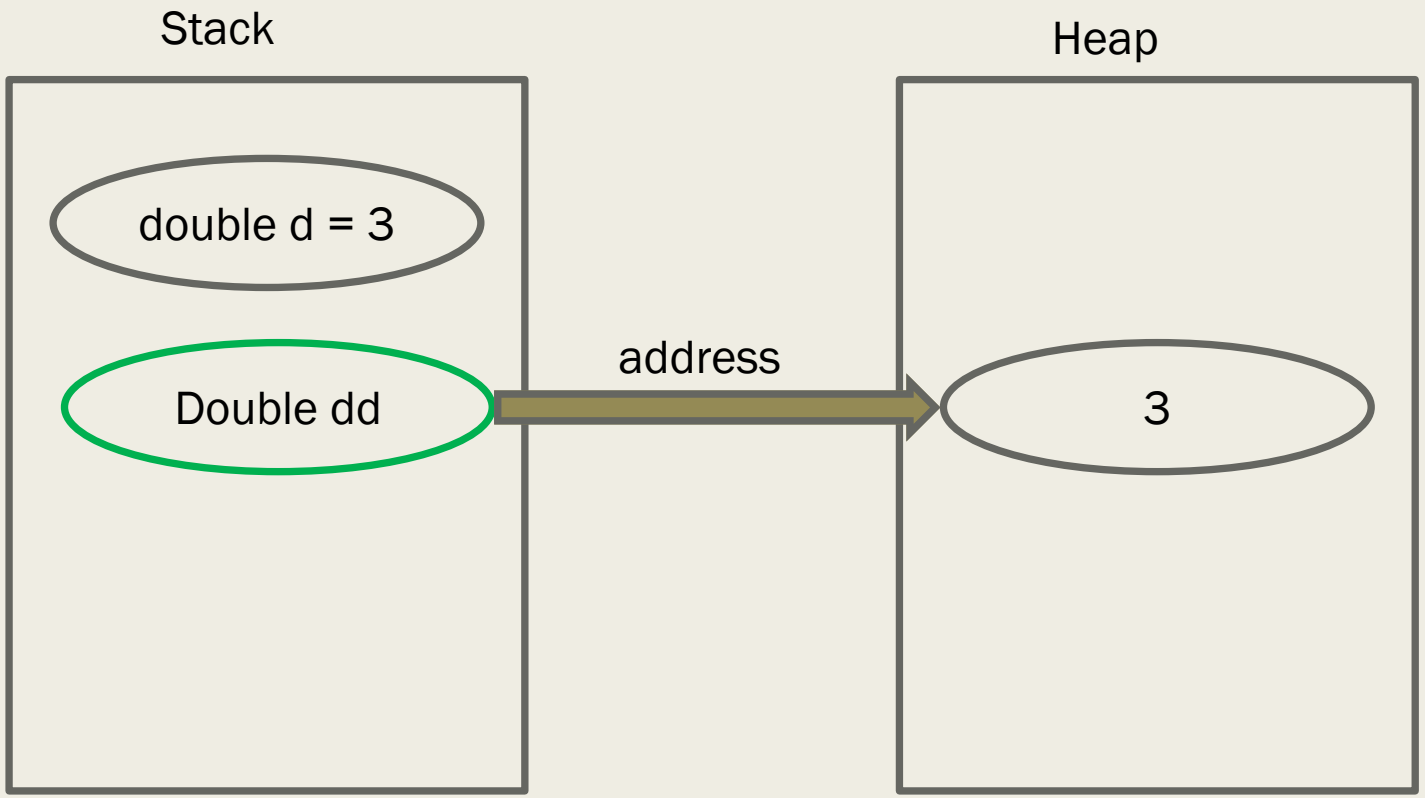
- Variable, on the other hand, changes its value dependent on the equation

Encapsulation

Data Types	Where to store	Example	Copy
Primitive Types	in stack	double = 3;	value
Reference Types	in heap (address in stack)	Double d = new Double(3);	copy address

Note: Every primitive type corresponds to a reference type. i.e. boolean and Boolean

Stack vs Heap



Type Casting Examples:

```
// create double type variable

double num = 10.99;
System.out.println("The double value: " + num);

// convert into int type int

int data = (int)num;
System.out.println("The integer value: " + data);
```

Type Casting Examples:

```
// create int type variable

int num = 10;
System.out.println("The integer value is: " + num);

// converts int to string type

String data = String.valueOf(num);
System.out.println("The string value is: " + data);
```

Type Casting Examples:

```
// create string type variable

String data = "10";
System.out.println("The string value is: " + data);

// convert string variable to int

int num = Integer.parseInt(data);
System.out.println("The integer value is: " + num);
```

Wrapper Class

int - Integer (boxing and unboxing)

Question?

```
int intOne = 1;
int intTwo = 1;
Integer boxingOne = new Integer(1);
Integer boxingTwo =  Integer.valueOf(1);
int sum = boxingOne + intOne;


System.out.println(intOne == intTwo);
System.out.println(intOne == boxingTwo);
System.out.println(boxingOne == boxingTwo);
System.out.println(sum);
```

Why we need wrapper class?

Wrapper classes are fundamental in Java because **they help a Java program be completely object-oriented**.

The primitive data types in java are not objects, by default. They need to be converted into objects using wrapper classes.

Operators

- & - bitwise AND
- && - logic AND
- | - bitwise OR
- || - logic OR
- ! - not
- Bitwise move >>>, <<< (not required to know details)

Iterator

- for loop
- do while

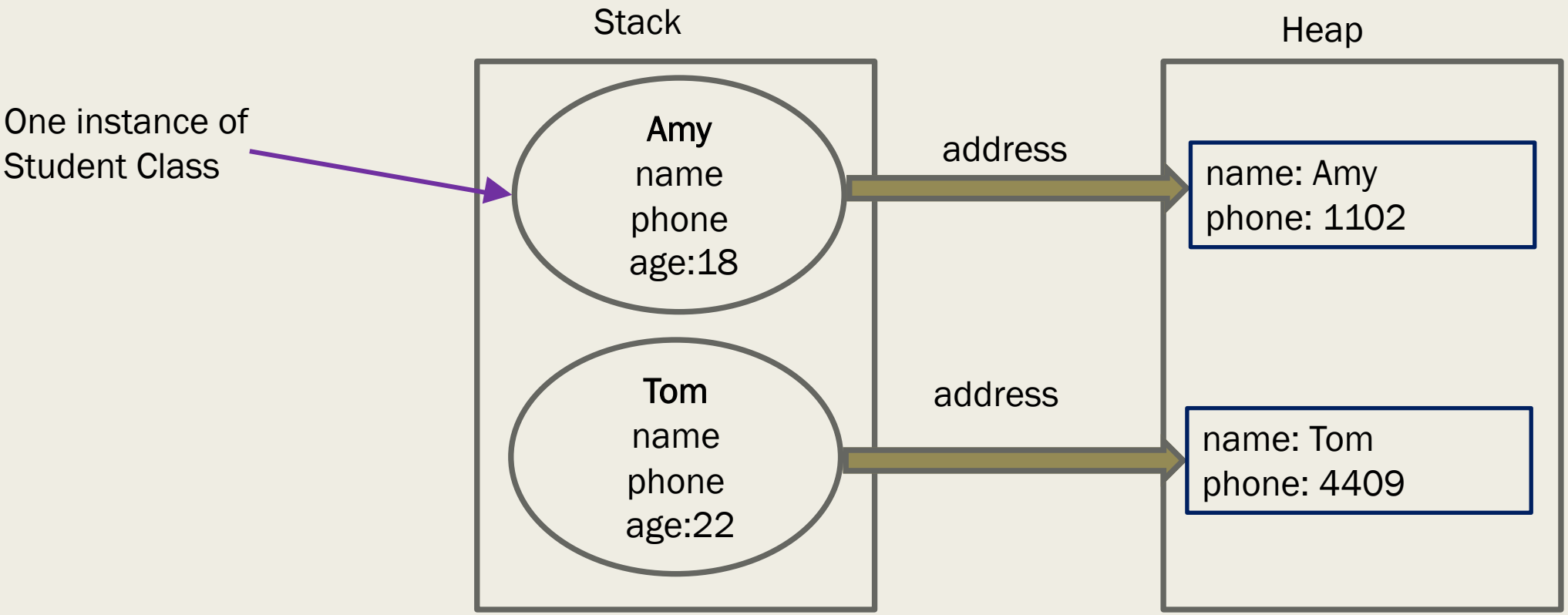
(difference do while loop and while loop - do the check for the first time)

OOP - Object Oriented programming

- What is object?
- What is class? - Give an example like Student class
- What is constructor?

- How to decide objects and classes
  - object is an instance of class, and class is a template of object.
  - class is composed by fields and methods
  - Initialization - when values are put into the memory that was allocated

Stack vs Heap



Object.java internal methods

Go to inheritance Graph

- equals()
- toString()
- hashCode()
- getClass()

- **equals()**

What’s the difference between equals() and == ?

== checks if both objects point to the same memory location whereas . equals() evaluates to the comparison of values in the objects

Question1 ?

```
Human h1 = new Human();
Human h2 = new Human();
System.out.println(h1 == h2);
System.out.println(h1.equals(h2));
```

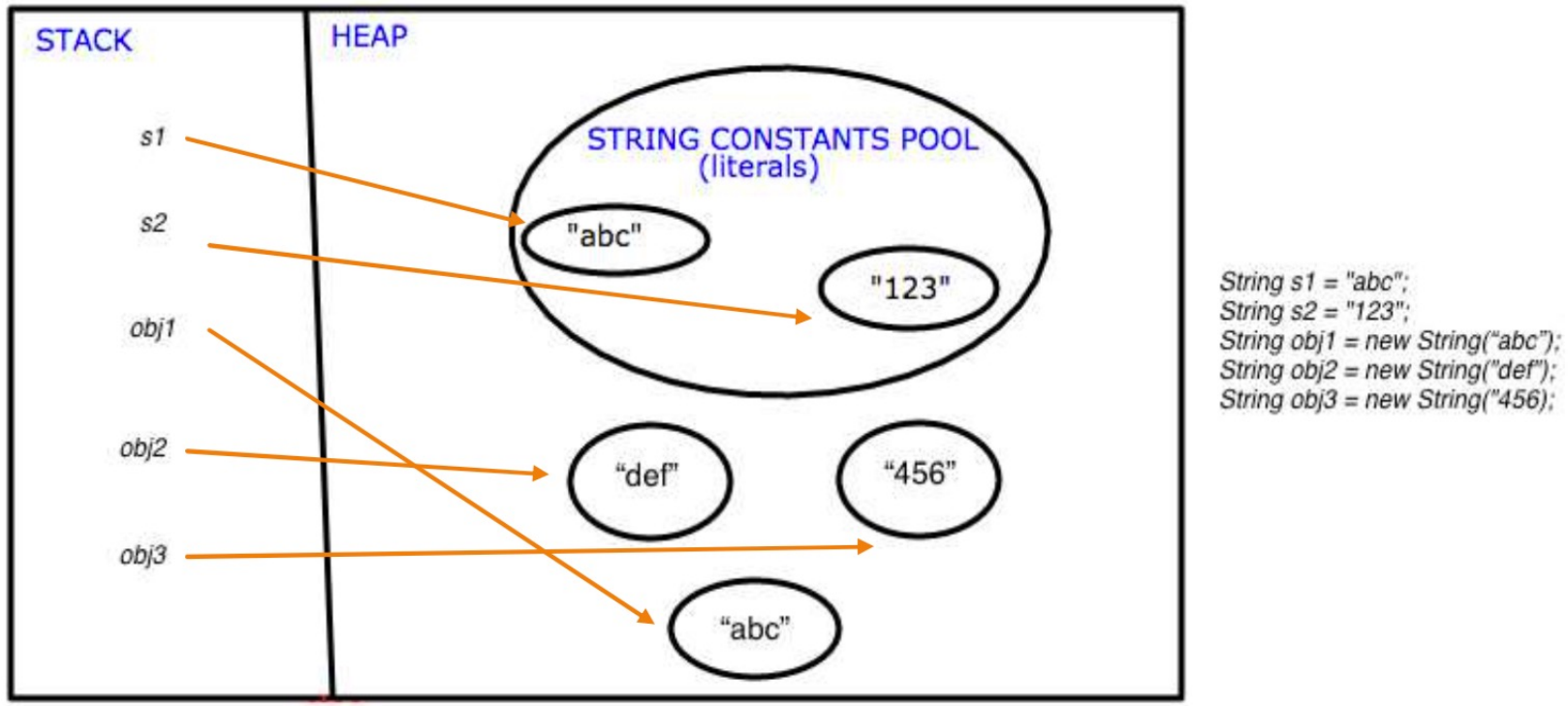
Question2 ?

```
String s1 = new String(“123”);
String s2 = new String(“123”);
System.out.println(s1 == s2);
System.out.println(s1.equals(s2));
```

Question3 ?

```
String s1 = “123”;
String s2 = “123”;
System.out.println(s1 == s2);
System.out.println(s1.equals(s2));
```

String Pool




```
String s1 = "abc";
String s2 = "123";
String obj1 = new String("abc");
String obj2 = new String("def");
String obj3 = new String("456);
```

Why we need String Pool?

It is created to decrease the number of string objects created in the memory. Whenever a new string is created, JVM first checks the string pool.

toString()

will convert object into String type.

Example:

```
Human human = new Human();
System.out.println(human.toString());

String name = “111”;
System.out.println(name);
```

Inheritance

What is inherit?

inherit attributes and methods from one class to another.

- **subclass** (child) - the class that inherits from another class
- **superclass** (parent) - the class being inherited from

Using **extends** as key word

Modifiers

- public can visit from different packages and classes
- private can visit only from same class (getter and setter)
- protected can visit from same packages or subclassed (child)

- static belongs to class and only stores in class's memory in heap
- final once give value, cannot be changed
- abstract cannot be initialized (to access it, it must be inherited from another class)

mutable vs immutable

A **mutable** object can be changed after it's created, and an **immutable** object can’t.

In Java, everything (except for strings) is mutable by default. Strings are immutable in Java. (If you want mutable strings in Java, you can use a StringBuilder/StringBuffer object)

Inside a class, make objects immutable by making all fields final and private.

Polymorphism

- Override

A same method as the parent class, but different implementation.

- Overload

Allows a class to have more than one method with the same name, but with different parameters. (different number params or different data types)

Features of Java:

- Encapsulation
- Inheritance
- Polymorphism

Arrays

Data types

String[]

Int[]

Two-dimension Arrays

https://leetcode.com/problemset/all/

Homework 1