# Quiz 00 Review Session

## COMP 210 / 2024 Summer Session I

**UTAs:** Ajay Gandecha and Liana Ma

# Quiz 00 Format

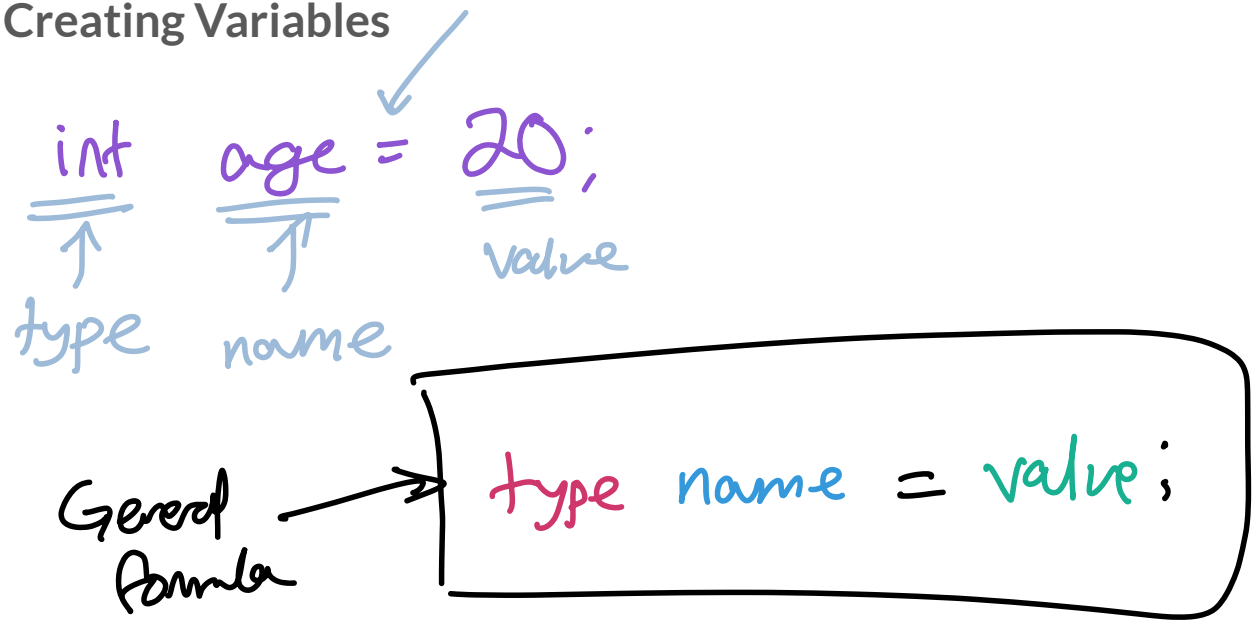
- 20 minutes at the start of class.
- *On paper* - bring a pencil!
- **Question Types:**
  - Multiple choice, T/F, select all that apply, fill in the blank.
  - *No code writing on this quiz - but be able to read given Java code!*

# On Quiz 00

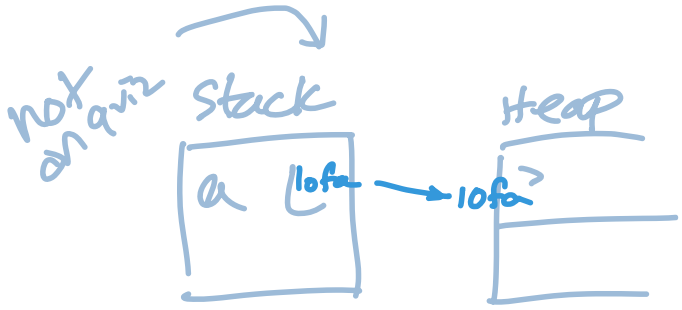
- Introductory Java programming
- Variables and data types
    - Value types vs Reference types (stack vs heap)
- Writing methods
    - Instance vs class methods
- Recursion

# Java Syntax

- Creating Variables



# Data Types




- **Value Types:** Value lives on the stack (defined entirely by their value)
  - Ex) `int`, `double`, `boolean`, `char`
- **Reference Types:** Value lives on the heap (identified by an address in memory)
  - Ex) `String`, arrays, `ArrayList<>`, `Triangle`, etc.

8.0
#s w/a decimal
8 vs 8.0
int
double
single characters
whole #
true or false
Reference type
the 'new' keyword
Defined by a class ⇒ Reference type
capital letter ⇒ Reference type
int[] a = new int[5];
Reference type

declare String name;
initialize name = "kaki";

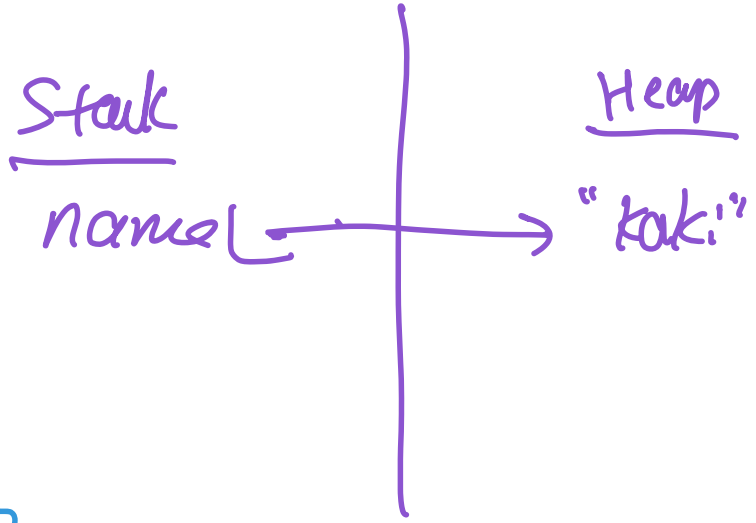
# Java Syntax

String a = new String("kaki");
String a = new "kaki";

- Strings

String name = "kaki";

type name value




- Arrays

String[] names = {"kaki", "Ajay"};

String[] name = new String[2];

size of array

Fixed length

# Java Syntax

- `if`-Statements

```
if ( condition ) {

} else {

}
```

Py: elif

Java: else if

```
if(a){ ...}
else if (b) {...}
else {..}
```

| Py | Java |
|---|---|
| or | \|\| |
| and | && |
| not | ! |

# Java Syntax

- `while-Loops`

```
int i = 0;
while ( i < ____ ) {
    ....
    i++;
}
```

arr. length

- `for-Loops`

```
for ( int i = 0;  i < arr.length; i++) {

}
```

```
for a in names:
    #...
```

```
for ( String a : names) {
    System.out.println( a );
}
```

array

# Java Syntax

- Methods

↳ Perform an action (like a py function)

accessModifier (optional) returntype name ( type name , type name... ) {

↳ public
↳ private

↳ static
↳ (nothing)

↳ valid type
↳ void
↓
when the method returns nothing

parameters

}

```
class Example {

    public static void printGreeting(String name) {
        System.out.println(name);
    }

}
```

static method == class method

vs

non-static method == "instance" method

Q) Can this method be used by objects outside of this class?

⇒ yes

Q) Do we need an Example object created to run printGreeting()?

```
----main(){
    Example.printGreeting();
}
```

# Recursion

- Calling a method inside of itself.

- Recursive case
  - ↳ Continue recursion
- Base case
  - ↳ No more recursion happens

```
Math.sqrt(___)   ✓

String.length()   ✗

String s = "hi";
s.length();   ✓
```

# Fibonacci Sequence - Run ~~fibonacci~~(5);

```
public static int fibonacci(int n) {
  if(n <= 1) {
    return n;
  }
  return fibonacci(n-1) + fibonacci(n-2);
}
```

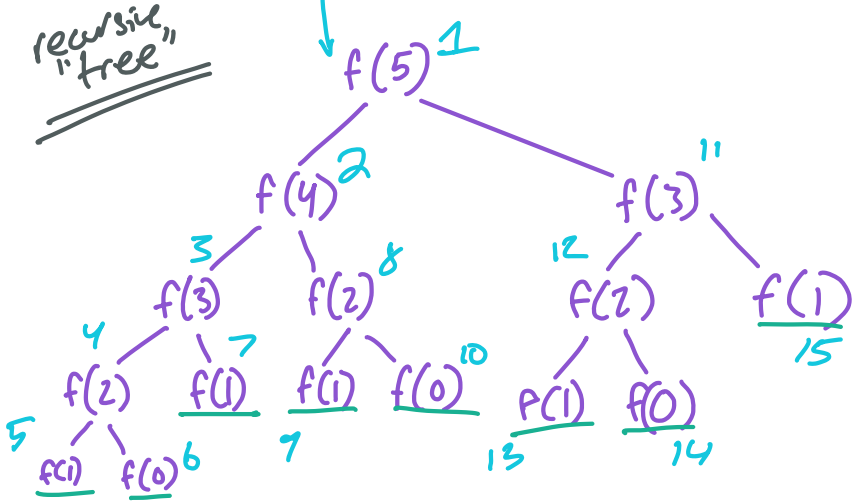
base case

recursive case

f(5) = f(4) + f(3) = 5
f(4) = f(3) + f(2) = 3
f(3) = f(2) + f(1) = 2
f(2) = f(1) + f(0) = 1
f(1) = 1
f(0) = 0

- How many stack frames were created? = 15
- What are the tradeoffs between using recursion vs. iteration?

Good for DS and for modeling subproblems




Downside: Takes more memory

class

Mystery.foo()

- class method
- instance method

Static!

Dog d = new Dog();
d.woof();
↑ object

- class method
- instance method

no static keyword