

CSCI 132:

Basic Data Structures and Algorithms

Lecture 4: Intro to Java (Loops, Arrays)

Reese Pearsall
Fall 2023

Announcements

- Lab 1 due **tomorrow** at 11:59 PM
- After today, you will be able to complete it
- Submit .java files (don't rename them)

Student Success Center - Fall 2023

Tutoring Schedule - Barnard Hall 259: Monday, August 28th - Friday, December 8th

Schedule	Monday	Tuesday	Wednesday	Thursday	Friday
8:00 a.m.					
9:00 a.m.	Adiesha Liyana Muzhou Chen	Kaden Bach	Sage James	Braeden Sopp	
10:00 a.m.	Adiesha Liyana Muzhou Chen	Gerard Shu Fuhnwi		Ruby Martin Sultan Ya	Justin Mau
11:00 a.m.		Gerard Shu Fuhnwi Connor Meyn	Henry Jacobson Alex Ellingsen	Racquel Bowen Sultan Ya	Justin Mau
Noon	Fatima Odobo Karishma Rahman	Maksym Makarchuk	Shama Maganur Fatima Odobo		Braeden Sopp
1:10 p.m.	Angelo Porcello	Brady Ash	Shama Maganur	Katie Harmon	Riley Slater
2:10 p.m.	Angelo Porcello		Gideon Popoola Wei You	Katie Harmon Karishma Rahman	Riley Slater
3:10 p.m.	Asibul Islam Shahnaj Mou	Muhammad Arju	Gideon Popoola Jasmine Vang	Katie Harmon	Gage Nesbit Boone Smail
4:10 p.m.		Muhammad Arju	Jasmine Vang		Joshua Bowen
5:10 p.m.	Asibul Islam Shahnaj Mou	Molly Claussen			
6:00 p.m.					
7:00 p.m.					
8:00 p.m.					
9:00 p.m.					

- Lab 2 and Program 1 will be posted soon TM

Example: A student is allowed to register for CSCI 476 if they have a GPA greater than 2.0, **and** if they are a Junior **or** Senior

```
public void allowToRegister() {  
    if (this.gpa > 2.0) { // check the first condition (Alternatively, we could use an && here)  
        if (this.year.equals("Junior") || this.year.equals("Senior")){  
            System.out.println("Student is allowed to register for CSCI 476");  
        }  
    }  
}
```

We can check one of two conditions is true using the or operator (||)

Student.Java

(we do not have the **or** keyword in Java)

```
student1.determineYear();
```

StudentDemo.Java

Example: A student is allowed to register for CSCI 476 if they have a GPA greater than 2.0, **and** if they are a Junior **or** Senior

```
public void allowToRegister() {  
    if (this.gpa > 2.0) { // check the first condition (Alternatively, we could use an && here)  
        if (this.year.equals("Junior") || this.year.equals("Senior")){  
            System.out.println("Student is allowed to register for CSCI 476");  
        }  
    }  
}
```

Student.Java

Why do `this.year.equals("Junior")` and not `this.year == "Junior"`

Checking for string equality in Java is a little bit funky...

Using `==` does **not** check for equivalence of values between two strings...

Example: A student is allowed to register for CSCI 476 if they have a GPA greater than 2.0, **and** if they are a Junior **or** Senior

```
public void allowToRegister() {  
  
    if (this.gpa > 2.0) { // check the first condition (Alternatively, we could use an && here)  
  
        if (this.year.equals("Junior") || this.year.equals("Senior")){  
  
            System.out.println("Student is allowed to register for CSCI 476");  
  
        }  
  
    }  
  
}
```

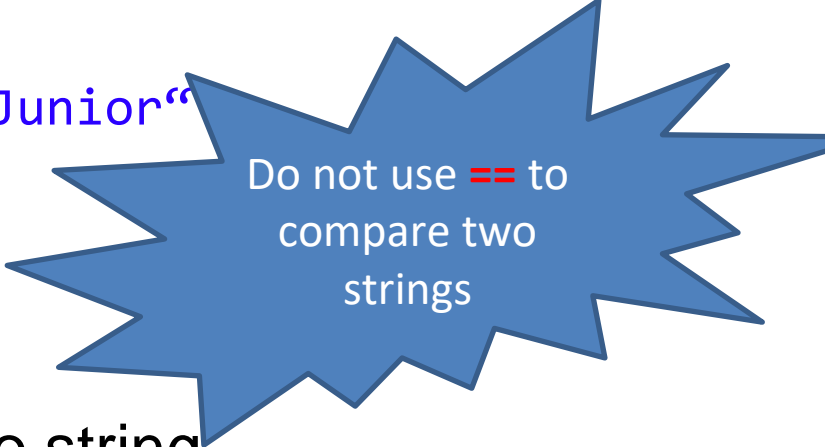
Student.Java

Why do `this.year.equals("Junior")` and not `this.year == "Junior"`?

Checking for string equality in Java is a little bit funky...

Using `==` does **not** check for equivalence of values between two strings...

Instead, we need to use the `.equals()` method between two strings



Do not use `==` to
compare two
strings

Arrays are a *collection* of data
→ Once initialized, are **fixed** in size
→ Can only hold one data type



Declaring an array and giving it a value

```
int[] test_scores = {99, 81, 65, 46};
```

Declaring an array allocating 5 empty spots (we need to fill them later)

```
String[] names = new String[5];
```

```
System.out.println(test_scores[2]);  
>> 65  
  
System.out.println(test_scores[4]);  
>> ERROR
```

test_scores	0	1	2	3	
	99	81	65	46	
names	0	1	2	3	4
	null	null	null	null	null

For loops can be used to iterate across an array.

Two ways:

1. Iterate by index

```
String[] animals = {"Zebra", "Elephant", "Lion", "Penguin"};

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

    System.out.println(animals[i]);

}
```

2. Iterate by element

For loops can be used to iterate across an array.

Two ways:

1. Iterate by index

```
String[] animals = {"Zebra", "Elephant", "Lion", "Penguin"};
```

```
for (int i = 0; i < animals.length; i++) {  
    System.out.println(animals[i]);  
}
```

Start at index 0 stop at index 4 (length of array) Increase the index by 1 each time

Each for loop has:

1. A start
2. A stop
3. A step

2. Iterate by element

For loops can be used to iterate across an array.

Two ways:

1. Iterate by index

```
String[] animals = {"Zebra", "Elephant", "Lion", "Penguin"};

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

    System.out.println(animals[i]);

}
```

2. Iterate by element

```
for (String i : animals) {
    System.out.println(i);
}
```

Both will give you the
exact same output...