CSCI 132:
Basic Data Structures and Algorithms

Intro to Java (Loops, Arrays)

Reese Pearsall
Spring 2025

Announcements

Note taker is needed in our course
- Commit to regularly attend class
- Type all notes during or after class
- Be in good academic standing and have strong note taking skills

This is a **PAID** position

Contact Office of Disability Services in Romney Hall 137 or **notetaking@montana.edu**

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

```java
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

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

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

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

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

```java
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

```java
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");

        }
    }
}
```
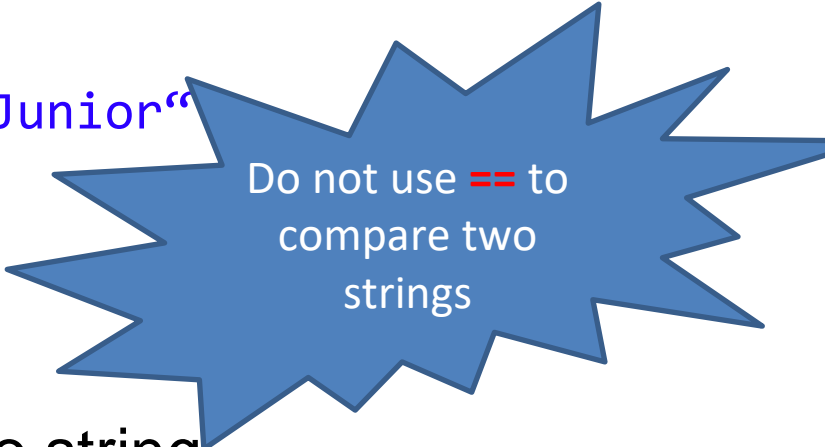
`Student.Java`

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 string

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

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

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

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

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

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

	0	1	2	3
test_scores	99	81	65	46

	0	1	2	3	4
names	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

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

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

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

}
```

2. Iterate by element

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

Both will give you the
exact same output…