

Girls Who Code Week 9

Object Oriented Programming Review

Katie Tooher and Carmen Seda

WIT Shout-Out of the Week:

Katie Moussouris



- Katie Moussouris is an American computer security researcher, entrepreneur, and pioneer in vulnerability disclosure, and is best known for her ongoing work advocating responsible security research.
- She began programming when she was in the 3rd grade on a computer that her mom bought for her and their family and was the first girl at her high school to take AP computer science
- She pioneered Symantec Vulnerability Research in 2004 for the antivirus company Symantec, which became the first program to allow Symantec researchers to publish vulnerability research.
- Senior Security Strategist Lead at Microsoft, where she ran the Security Community Outreach and Strategy team for Microsoft as part of the Microsoft Security Response Center (MSRC) team
- She now owns her own company called Luta Security, a consultancy to help organizations and governments work collaboratively with hackers through bug bounty programs.

WIT Shout-Out of the Week: Katie Moussouris



<https://www.linkedin.com/in/kmoussouris/>

Videos

<https://thehill.com/policy/cybersecurity/356983-ga-with-katie-moussouris-cybersecurity-professional>



Warm Up Activity



Warm Up Activity

1. Get a user inputted string and print it out in reverse order
2. Write a c# program that takes in a string as input and then replaces all e's with 3 and prints it back out
 - a. Use a foreach loop



Object-Oriented Programming REVIEW





Students



Classes



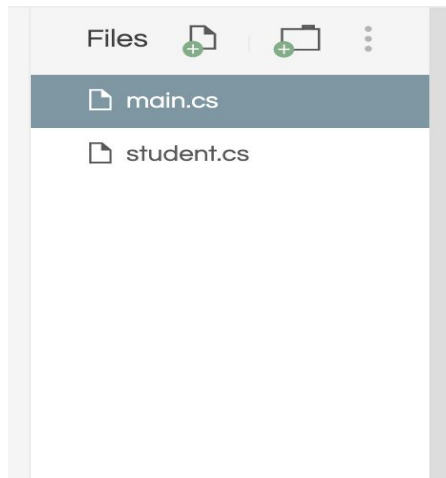


What is a Class?

- **Class** - a grouping of code used to represent an **object**
 - Contains methods and attributes for **object**
 - Way to model objects in code
 - Both reusable and distinct
- MainClass
 - In C#, all code must be contained in a **class**
 - MainClass is the **driver program**
 - Contains **main method**

How to Make a Class in Repl

- Create a **new file**
 - Name file same as class
 - Each class gets its own file
- Use `class` keyword
- Name class



Click the file tab to make a new file and name it student.cs to create your student class

```
class StudentClass{
```

```
}
```

Objects





Objects Definition

- **Object** - *instance* of a class
 - An entity/item
 - Only one piece of a whole class of items
 - Contains all attributes defined in class
 - Student Class - a single student is an object
- **Instance** - a single piece of a larger picture
- Objects are able to access all code from corresponding class



Object Examples

```
Student katie = new Student("Katie", 15, 20, 006108088);  
Student carmen = new Student("Carmen", 15, 20, 09022222);  
Student Jack = new Student();
```

How to create an object:

NameOfClass object-variable = new NameOfClass(attributes);



Attributes

- Attribute - a property that applies to every object of a class
- Attributes are **global variables** within a class
 - Global variables - defined inside class, outside methods
 - Can create as many attributes as you need
 - Student Class - we decided the attributes

```
class Student{  
    string name, grade, subject, allergy;  
    int age;
```

Constructors





Constructors

- **Constructors** - special methods called to create an object
 - Define attributes of object
 - Can have 0 parameters (default constructor)
 - Can have multiple parameters



Multiple (Overloaded) Constructors

You can have multiple constructors as long as the parameters are different

```
public Student(){  
    name = "Student";  
    grade = "freshman";  
    subject = "math";  
    allergy = "none";  
    age = 15;  
}
```

```
public Student(string name,  
    string grade, string subject,  
    string allergy, int age){  
    this.name = name;  
    this.grade = grade;  
    this.subject = subject;  
    this.allergy = allergy;  
    this.age = age;  
}
```

What the heck is **this**?

- **this** is a **keyword** used to clarify between global variables of a class and parameters of a method
- ONLY NEEDED if the parameter and global variable have the same exact name
- **this** refers to the **global variables**, NOT the **parameters**
 - Aka the attributes of the object

```
public Student(string name, string grade, string subject, string allergy, int age){  
    this.name = name;  
    this.grade = grade;  
    this.subject = subject;  
    this.allergy = allergy;  
    this.age = age;  
}
```

What the heck is **this**?

- Can avoid using **this** by naming parameters something else
- YOU WILL GET THE EXACT SAME RESULTS

```
public Student(string n, string g, string s, string al, int a){  
    name = n;  
    grade = g;  
    subject = s;  
    allergy = al;  
    age = a;  
}
```

Creating a Constructor



- Need: access modifier, name of class, parameters
- Parameters are the attributes of the object

```
public Student(string name, string grade, string subject, string allergy, int age){  
    this.name = name;  
    this.grade = grade;  
    this.subject = subject;  
    this.allergy = allergy;  
    this.age = age;  
}
```

```
public Student(string n, string g, string s, string al, int a){  
    name = n;  
    grade = g;  
    subject = s;  
    allergy = al;  
    age = a;  
}
```

Accessing Attributes





Set Methods (Setters)

- If you create an object and want to change an attribute later, you do so using “setter” methods
 - Setters are methods made within the object class
 - They take a parameter and assign that parameter to the corresponding attribute

```
public void setName(string name)
{
    this.name = name;
}
```



How to set an attribute with the setter method

```
class MainClass {  
    public static void Main (string[] args) {  
        Student jack = new Student();  
        jack.setName("Jack");  
    }  
}
```




Get Methods (Getters)

- If you create an object and want to display an attribute, you do so using “Getter” methods
 - Getters are methods made within the object class and return an attribute

```
public string getName() {  
    return name;  
}
```



How to call the getter method to return an attribute

```
class MainClass {  
    public static void Main (string[] args) {  
        Student jack = new Student();  
        Console.WriteLine("Student Grade: " + jack.getGrade());  
    }  
}
```

ToString()

You can also create a method for your object that prints all the attributes. Typically, this is done by **overriding** an existing method known as ToString();

```
public override string ToString() {  
    return name + " is a " + age + " year old " + grade + " whose  
    favorite subject is " + subject + " and is allergic to " +  
    allergy + ".";  
}
```

```
public override string ToString() {  
    return $"{name} is a {age} year old {grade} whose favorite  
    subject is {subject} and is allergic to {allergy}.";  
}
```

Student Class





Practice





Create a Teacher class

- Create a teacher class with attributes for the last name of the teacher, the subject they teach, and their age.
- Make sure you add getters and setters
- Make sure you add a ToString override method so that you can print out the object



Create a Parent class

- Create a parent class with attributes for their first name, last name, name of child, name of spouse, and age.
- Make sure you add getters and setters
- Make sure you add a ToString override method so that you can print out the object