**Polymorphism> operator overloading**

What is an operator? –

Diagram

Description automatically generated

Polymorphism : one thing which has multiple forms.

We can add two integer values, we can add a int and a float value, can we add two strings? – so + works with

Q: will the below code work? If yes, what is the output, if not observe the error message

Graphical user interface

Description automatically generated

In the front end the things are simplified which can be called as ‘synthetic sugar’. Let’s observe how it works behind the scenes.

First check the classes for a and b, make a note of them. Let’s check the int class as shown below

Graphical user interface, text, application

Description automatically generated

Run the below code and notice that both are printing the same output. We are familiar with the first print statement. However, in programming we do most of the things with methods as highlighted in the second print statement. **Click on the method + ctrl button we can see more info(class/method)**

Graphical user interface, application

Description automatically generated

So when we usually perform the regular print function, behind the scenes the second is being called hence the same output.

🡪Let’s take a look on how to add two strings, run the below code snippet and observe the output. Both got concatenated. Check the add method by using **ctrl+click on the add method which is part of str class.**

Graphical user interface, application

Description automatically generated

* So what happens is when you use + it will call \_\_add\_\_() method, when you use – it will call \_\_sub\_\_() method, when you use \* it will call \_\_mul\_\_() method which are called as **magic methods** as they are working behind the scenes.

A picture containing background pattern

Description automatically generated

* Let’s take a look at operator overloading. To understand it let’s create a class and create 2 student objects as shown below

Graphical user interface, text, application

Description automatically generated

By looking at the error message we can see that + operand type is not supported for student and student. Behind the scenes the + symbol will call the add method. However, in our student class we don’t have the add method, as when you say s1 + s2 how your compiler will know what to do. That’s where we need to mention in our class a separate method \_\_add\_\_ and it will be called when a user mentions + symbol. **By doing so we are overloading the operator and you can change the definition for it and define anything you want.**

Let’s define a add method as shown below. Highlighted in red shows what’s happening behind the scenes.

A picture containing graphical user interface

Description automatically generated

* Create a new method \_\_add\_\_ as shown below and run the code.

A picture containing timeline

Description automatically generated

* If you want to add 2 students you need to overload the operator of +, because integer knows what is +, str knows what is +, however our student class doesn’t know what is plus. Hence we had mentioned that when the user calls + it will call the \_\_add\_\_ method. Look at some predefined functions like \_\_mul\_\_ etc..
* Let’s look at another example: compare the student marks which means whoever have the maximum marks they will WIN. We have to mention the > operator which we know, however our class doesn’t know that.
* Graphical user interface, application

  Description automatically generated

If you run the code it will give error.

* Let’s define a method inside the student class (gt = greater than) and (ge = greater than equal), we need gt as we had need greater than only.

A picture containing graphical user interface

Description automatically generated

Now if we run the code we should get **s1 wins.** S1 values are higher, try changing the s2 values.

If you want to perform any operation on the objects which are user defined you have to define all these methods.

* Now let’s look at something else. What if you have a variable a = 9, the moment you print a it will print the value of a NOT the address of a. Now question is what happens when we print the s1 it will NOT print the value of s1, instead it will print the address of s1

Graphical user interface, application

Description automatically generated

We don’t want address, we want values. When we try to print the object it doesn’t matter it is int/class, behind the scene it is calling the method called as \_\_str\_\_ as shown below.

Graphical user interface

Description automatically generated with low confidence which means when we try to print a it is calling print(a.\_\_str\_\_()) as highlighted above and in the same way for print(s1) it is also calling \_\_str\_\_()

Graphical user interface, text, application

Description automatically generated

Let’s take a look at \_\_str\_\_() by clicking ctrl + str we will have the builtin’s , it is printing the module name which is \_\_main\_\_(), class student, object address.

Graphical user interface, text, application

Description automatically generated

We want to print the value of s1, hence we need to override this method. Let’s create a method as shown below.

We want to define the value of m1 and m2: self.m1 and self.m2. Observe the output, this time we got values 58 and 69

Graphical user interface, text, application

Description automatically generated

* Now if we want to print the object by removing str method in the print statement, run the code we will get error
* Graphical user interface, application

  Description automatically generated

The method we created in the class for str it returning a non str value, however, we need a str value. Let’s modify the method. We know that by default print will take the str and we are passing non str value to it, hence we will modify the return statement inside the method.

Graphical user interface, text, application

Description automatically generated

Print(s2) as well and check the output.

Conclusion: Whenever you perform +,-,/ there are methods behind the scenes which we are calling.

**Method overloading and method overriding**

**Diagram

Description automatically generated**

In Python we ***do not*** have method overloading. However, in java and some other programming languages they allow multiple methods with same name and multiple parameters/arguments.

Graphical user interface, text, application

Description automatically generated

Let’s take a look at method overloading:

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Method Overriding:

Which means we have two methods with same name and same parameters/arguments but ***NOT*** in the same class

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

class Student:

    def \_\_init\_\_(self, m1,m2):

        self.m1 = m1

        self.m2 = m2

    def sum(self, a=None,b=None,c=None):

        s = 0

        if a!=None and b!=None and c!=None:

            s = a + b + c

        elif a!=None and b!=None:

            s = a + b

        else:

            s = a

        return s

s1 = Student(58,59)

print(s1.sum(5,4))