**FSDS MAY BATCH 2022(Python Assignment -4)**

**Submitted by: Shubham Tiwari**

Q1: Which two operator overloading methods can you use in your classes to support iteration?

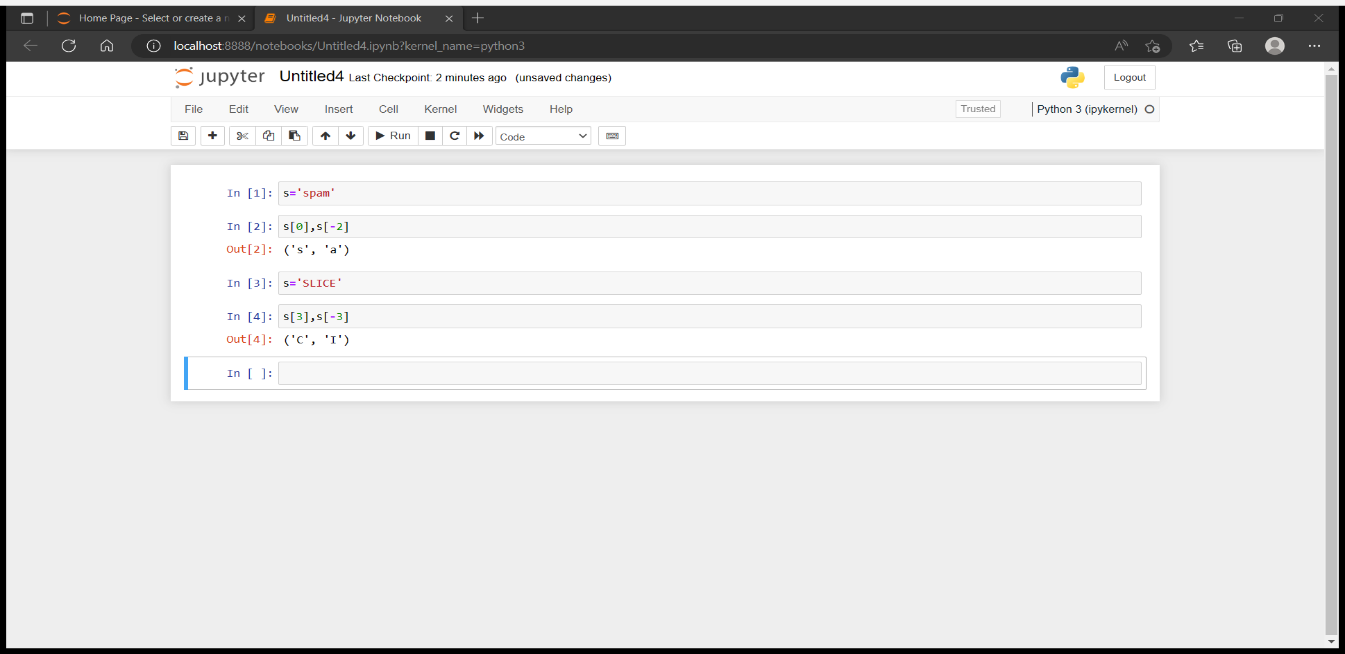
Ans: The operator overloading methods which we can use in our classes to support iteration can be **\_\_getitem or \_\_iter\_\_,**but in all iteration python tries to use \_\_iter\_\_ which returns an object thar supports the iteration with a next method.First if no \_\_iter\_\_ is found by inheritance search ,python goes on the \_\_get item indexing method.

Q2: In what contexts do the two operator overloading methods manage printing?

Ans: **Operator Overloading** means giving extended meaning beyond their predefined operational meaning. For example operator + is used to add two integers as well as join two strings and merge two lists. It is achievable because ‘+’ operator is overloaded by int class and str class.Whenever we change the behavior of the **existing operator** through operator overloading, we have to redefine the special function that is invoked automatically when the operator is used with the objects so this way it manages printing.

Q3: In a class, how do you intercept slice operations?

Ans: **Slicing operations**: In python language ,the indexing basically starts from zero index and ends at one less than the length of string so it basically prints according to our needs .There are basically two types of slicing operations i.e positive slicing and negative slicing , in positive slicing the indexing starts from left end whereas in negative slicing it starts from right end. This way the slicing operations take place .Below is the image in which slicing operation have been performed.



Q4: In a class, how do you capture in-place addition?

Ans:Python provides the operator x += y to add two objects in-place by calculating the sum x + y and assigning the result to the first operands variable name x. we can set up the in-place addition behavior for our own class by overriding the magic “dunder” method \_\_iadd\_\_(self, other) in our class definition.

Q5: When is it appropriate to use operator overloading?

Ans: Operator overloading is mostly useful when we are making a new class that falls into an existing "Abstract Base Class" (ABC) infact, many of the ABCs in standard library module [collections](http://docs.python.org/library/collections.html#module-collections) rely on the presence of certain special methods (and special methods, one with names starting and ending with double underscores AKA "dunders", are exactly the way we perform operator overloading in Python). This provides us the good starting generalization.