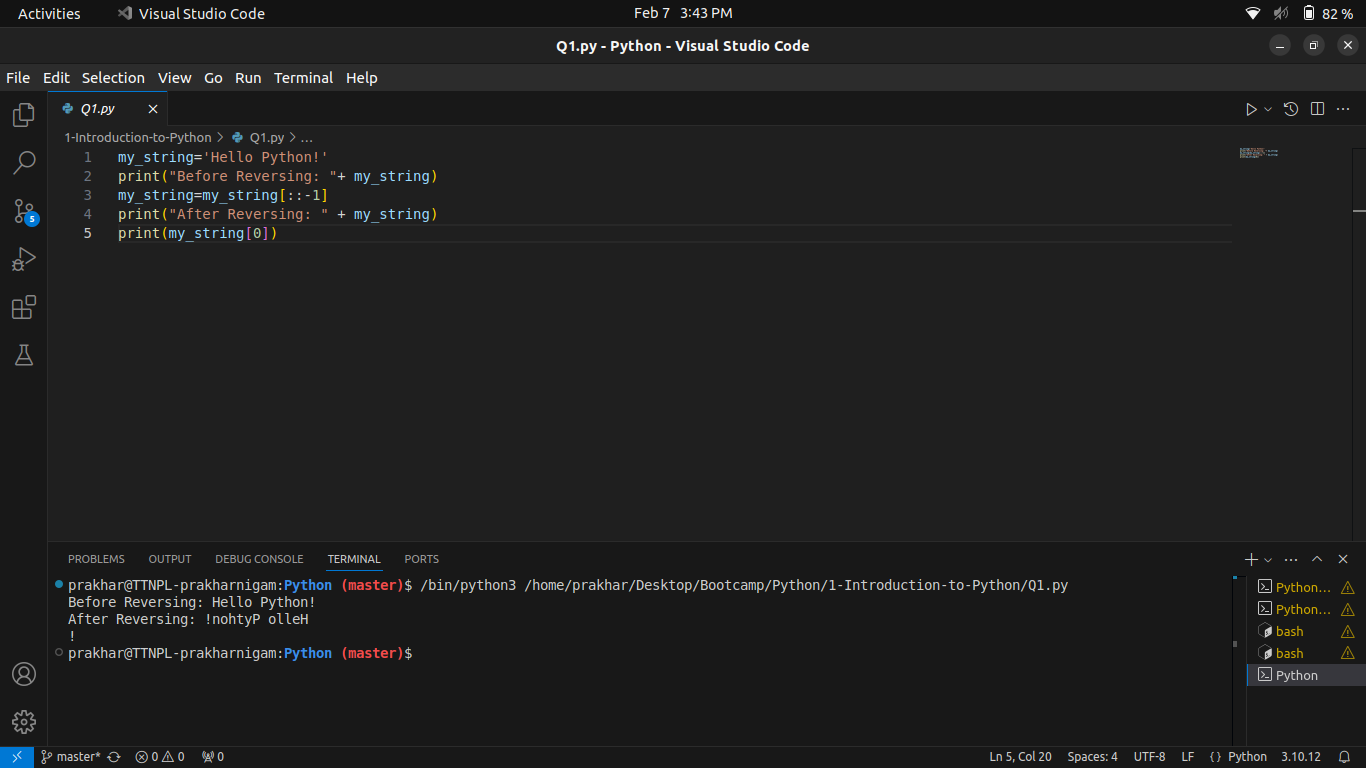
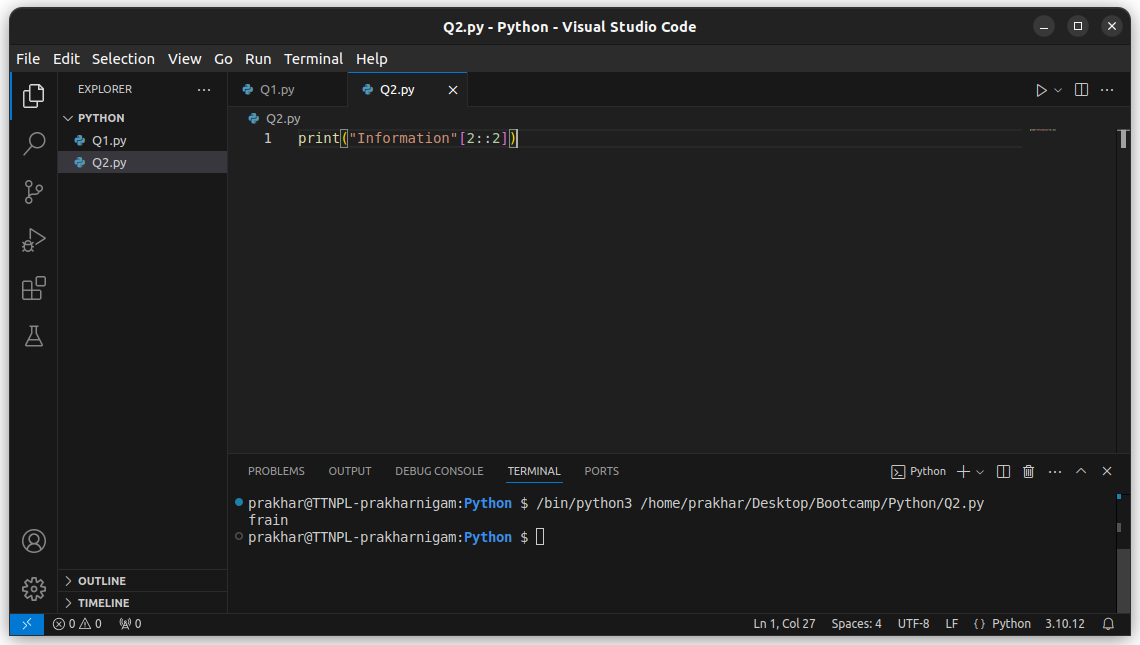
Assignment (Introduction to Python)

**Q1. Given string my\_string = ‘Hello Python!’, Reverse the string using slicing, print ’!’ using indexing**



**Q2. Use slicing to get word “frain” from “information”.**



**Q3. Using examples explain string.format and f-strings**

There are 2 methods of formatting strings with values of variables in python. The two methods are string.format and f-strings

**String.format:**

The string.format method is an improvement compared to the % operator (like in c we used to write %d or %f for decimal or float values) because it fixes a couple of issues and supports the string formatting mini-language. With .format, curly braces delimit the replacement fields:

>>> name = "Jane"

>>> age = 25

>>> "Hello, {}! You're {} years old.".format(name, age)

"Hello, Jane! You're 25 years old."

For the .format() method to work, we provide replacement fields using curly brackets. If we use empty brackets, then the method interpolates its arguments into the target string based on position. We can manually specify the interpolation order by referencing the position of each argument to .format() using zero-based indices.

**F-strings:**

F-strings make the string interpolation process intuitive, quick, and concise. The syntax is similar to what we used with .format(), but it’s more compact. We only need to start our string literal with a lowercase or uppercase f and then embed our values, objects, or expressions in curly brackets at specific places:

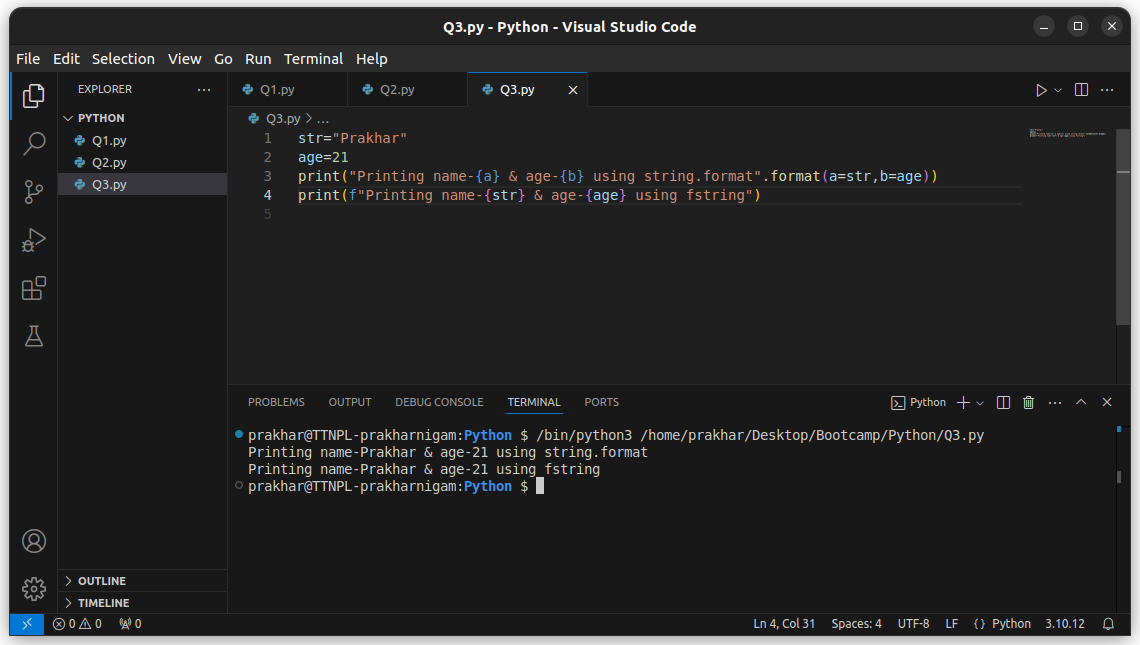
>>> name = "Jane"

>>> age = 25

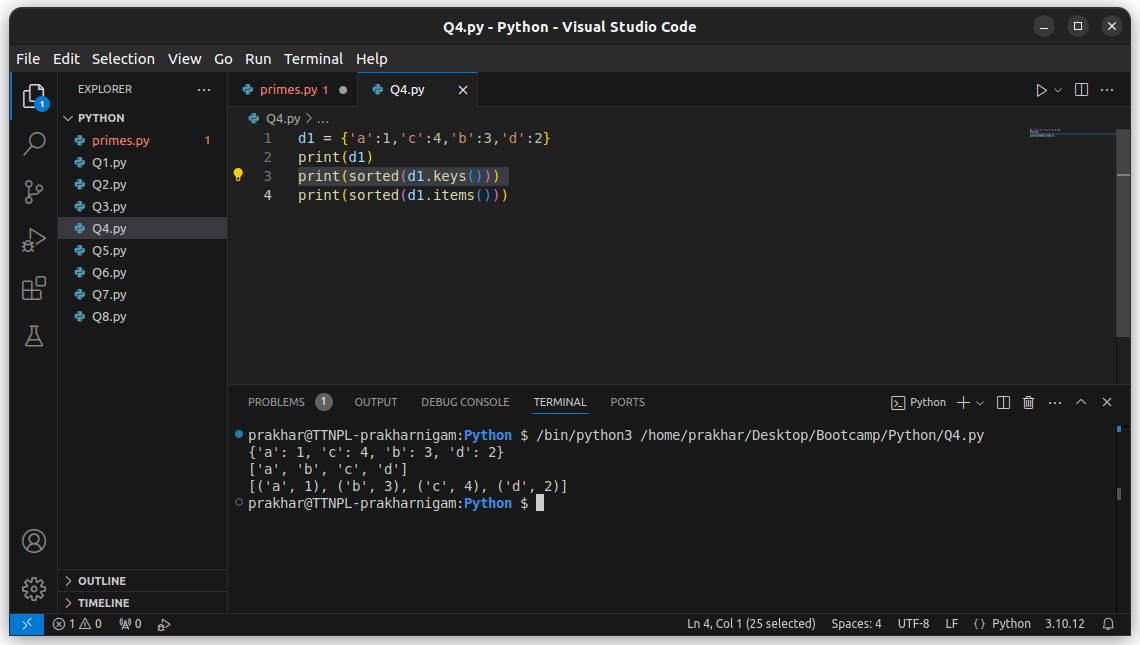
>>> f"Hello, {name}! You're {age} years old."

'Hello, Jane! You're 25 years old.'

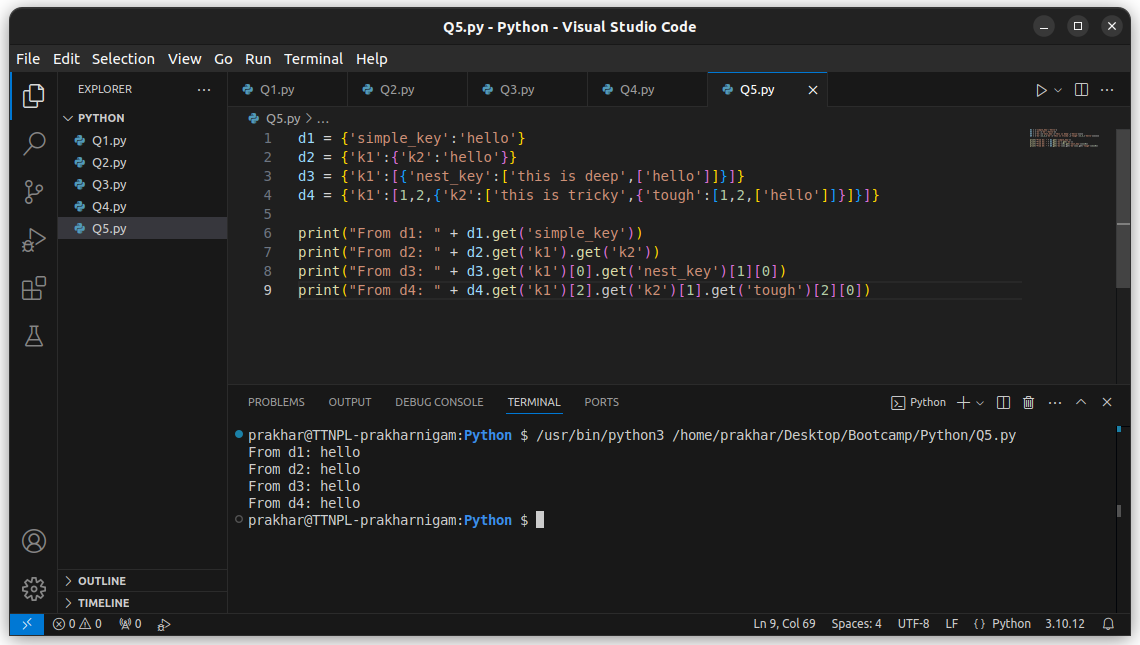
It makes the text more readable and concise. While using fstring we don’t need operators or methods anymore. We just embed the desired objects or expressions in our string literal using curly brackets. Below is an example of both.



**Q4. Can we sort a dictionary? Why or why not?**  
  
Yes, dictionary is sortable, but not traditionally. It is unordered in its original nature, however we can explicitly sort it. [Python](https://www.javatpoint.com/python-tutorial) offers the [built-in keys functions](https://www.javatpoint.com/python-built-in-functions) keys() and values() to help us sort the dictionary. We use ‘sorted’ function to sort. It takes any iterable as an argument and returns the sorted list of keys. We can use the keys to sort the dictionary in ascending/descendin order.

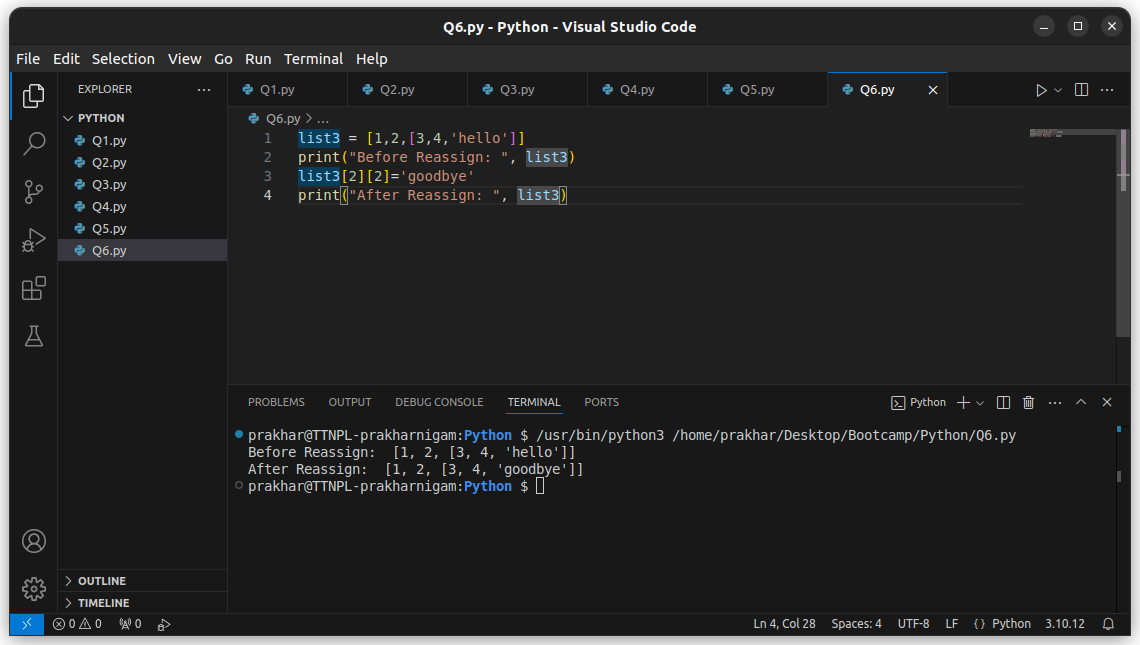


**Q5. Using keys and indexing, grab the 'hello' from the following dictionaries:**

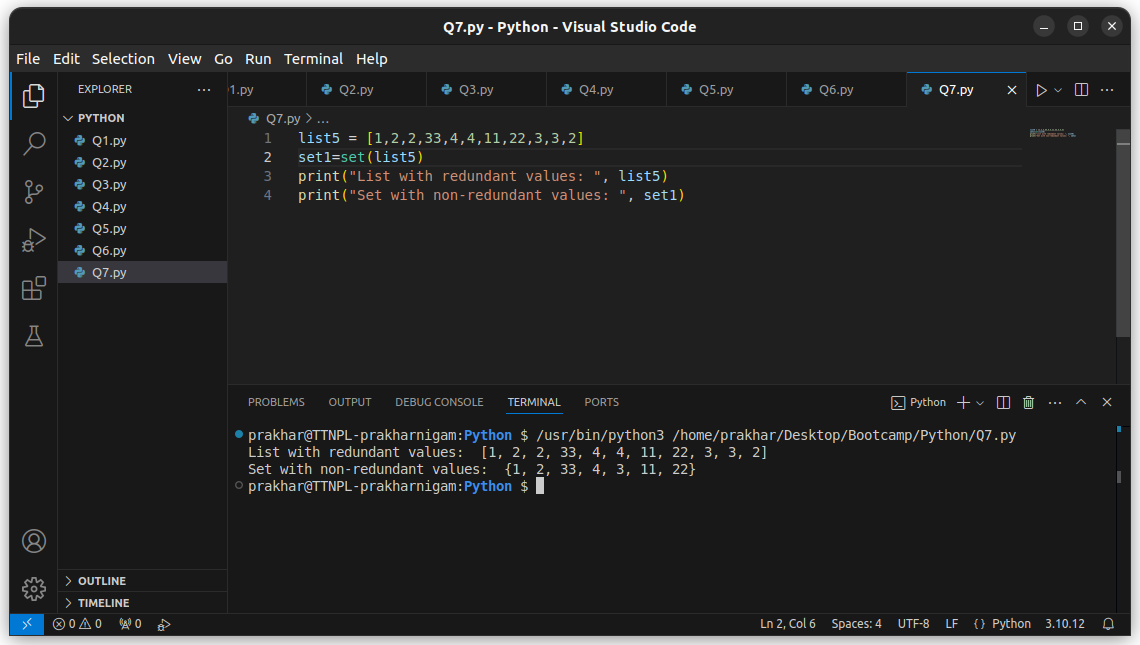


*Note: Instead of using dict.get(), I could have also used just dict[‘key’]*

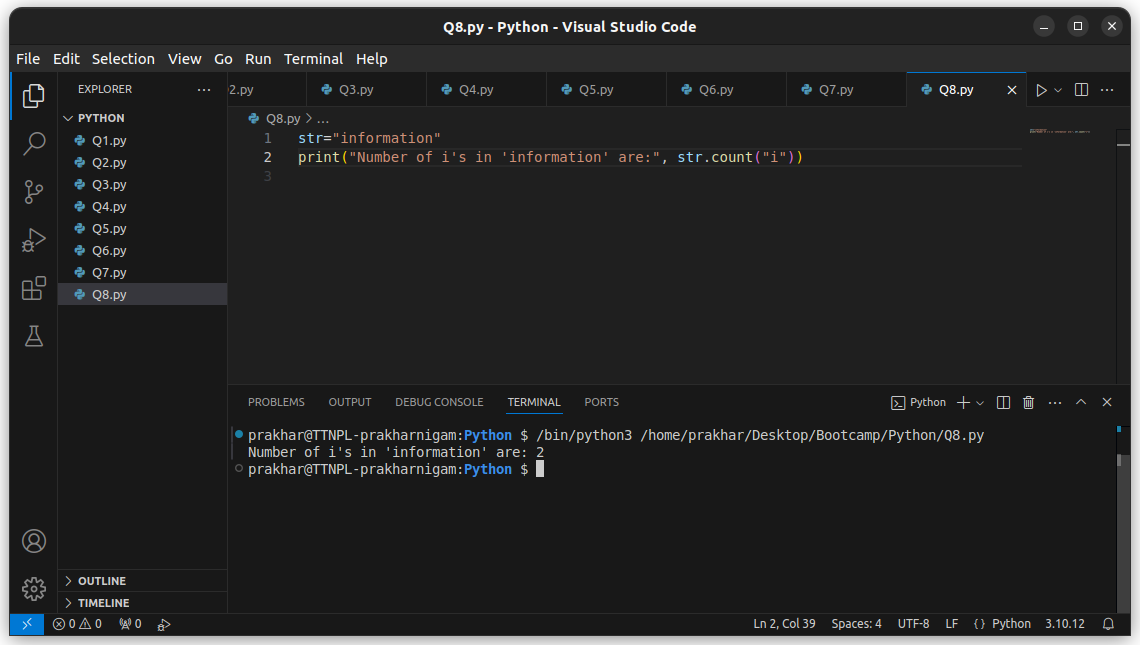
**Q6. Reassign 'hello' in this nested list to say 'goodbye' instead:**



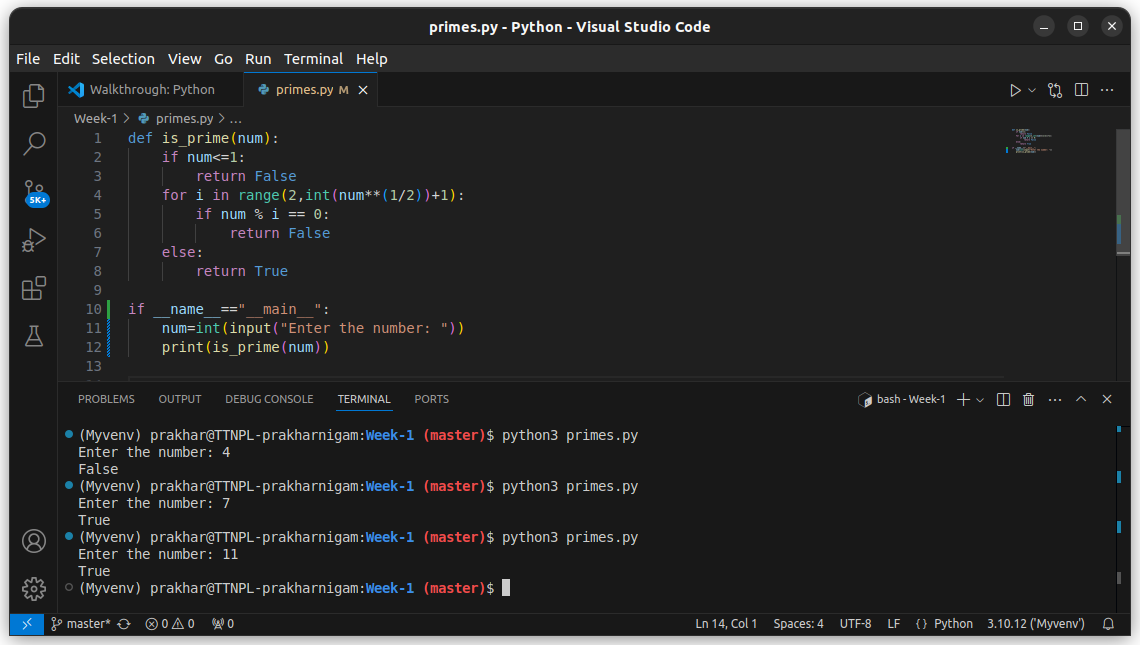
**Q7. From the given list list5 create a set: *list5 = [1,2,2,33,4,4,11,22,3,3,2]***



**Q8. In string information count the total number of i.**



**Q9. Write a Python script to test if a number is prime or not? - The Script name: primes.py - Add a functions is\_prime() which return boolean True or False - Program should accept a number from console**



*Used terminal to run the script and to enter the number.*

**Q10. Write a code to print binary, octal or hexa-decimal presentation of a number. Do not use any third party library.**

