

ITW1
PYTHON ASSIGNMENT2

tarun.arora.cse20@itbhu.ac.in

Name: Tarun Arora

Roll No.: 20075092

Branch: CSE

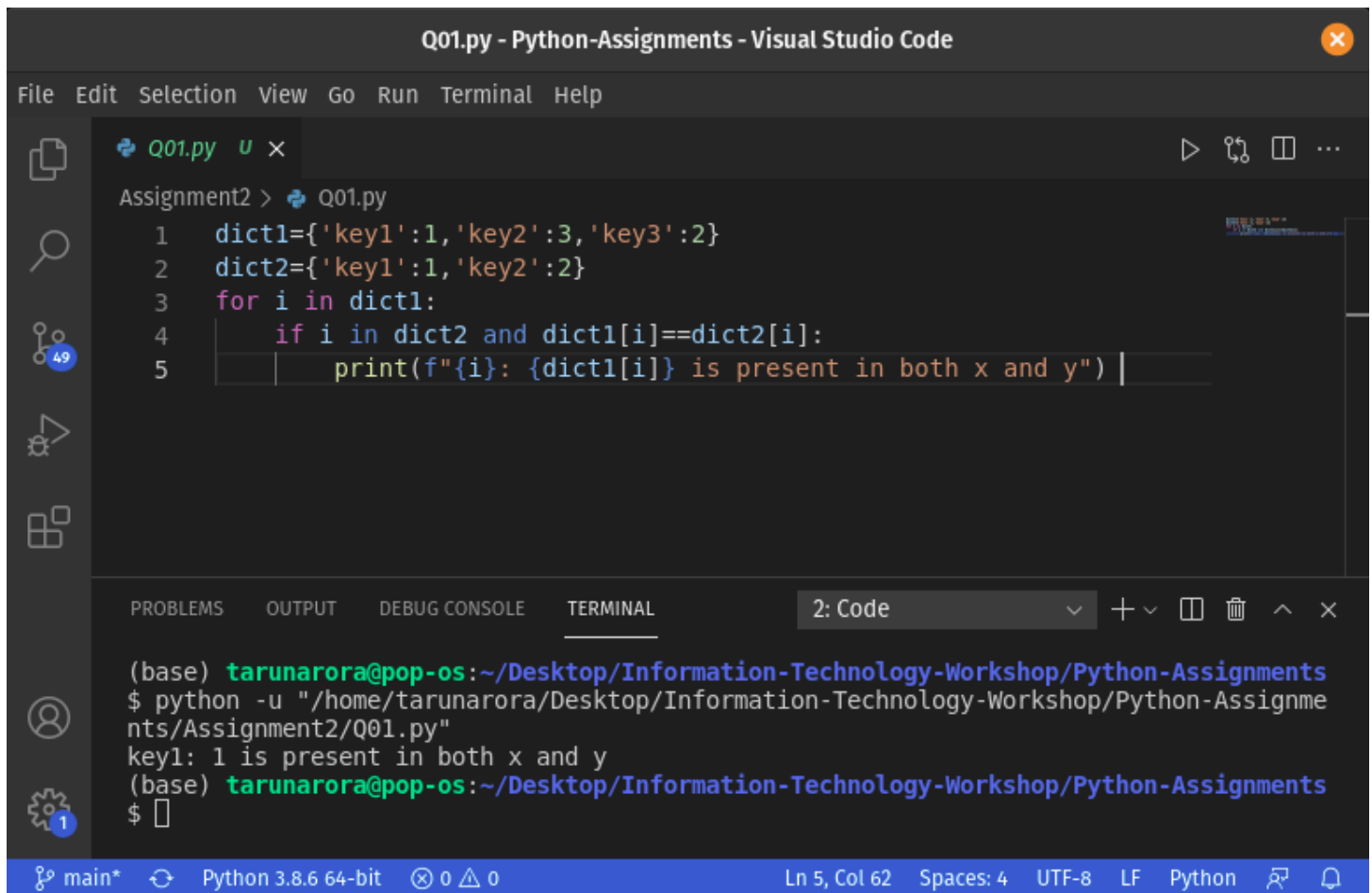
Date: July 20, 2021

1. Write a Python program to match key values in two dictionaries.

Sample dictionary: {'key1': 1, 'key2': 3, 'key3': 2}, {'key1': 1, 'key2': 2}

Expected output: key1: 1 is present in both x and y.

Solution: -



The screenshot shows the Visual Studio Code interface with a file named 'Q01.py' open. The code in the editor is as follows:

```
1 dict1={'key1':1,'key2':3,'key3':2}
2 dict2={'key1':1,'key2':2}
3 for i in dict1:
4     if i in dict2 and dict1[i]==dict2[i]:
5         print(f"{i}: {dict1[i]} is present in both x and y")
```

Below the code editor, the 'TERMINAL' panel is active, showing the command to run the script and its output:

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments
$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q01.py"
key1: 1 is present in both x and y
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments
$
```

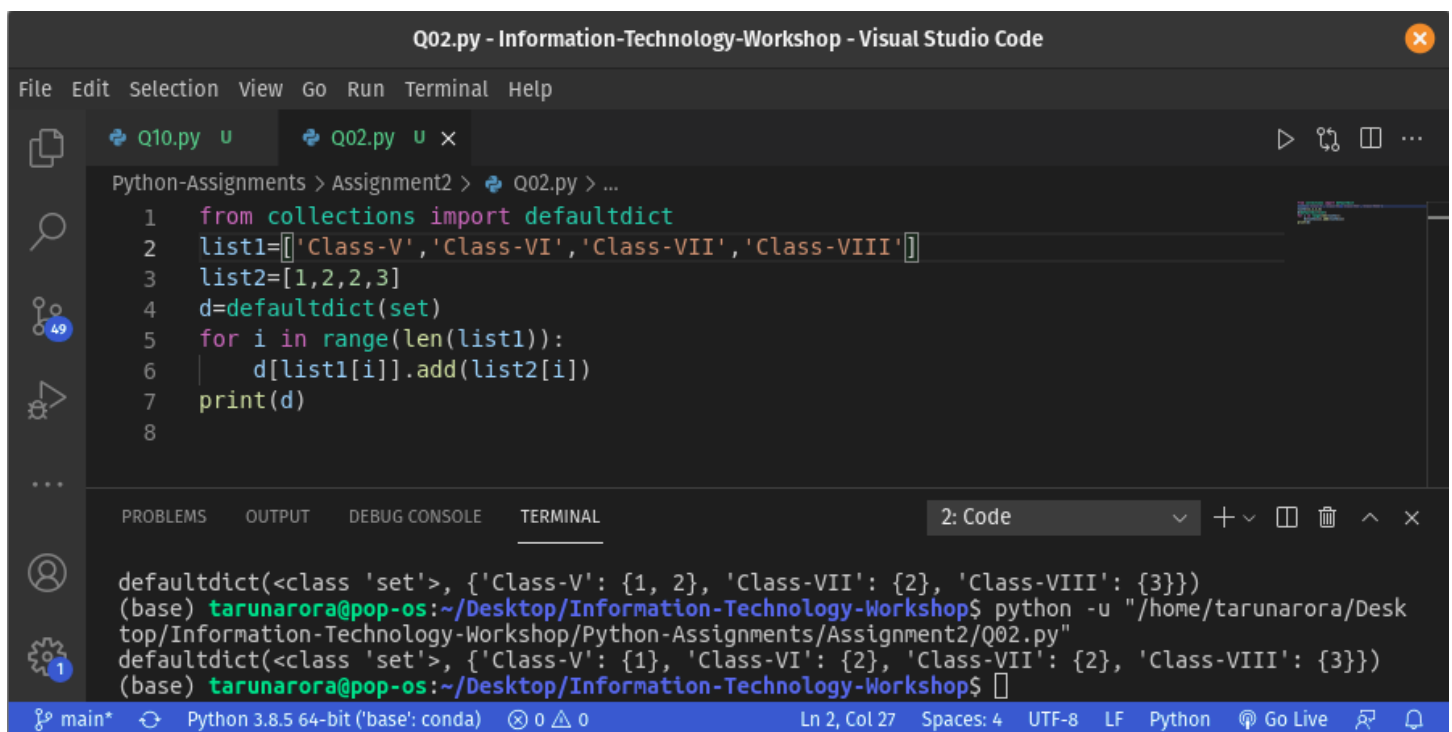
The status bar at the bottom indicates the current file is 'main*', the Python version is 'Python 3.8.6 64-bit', and the cursor is at 'Ln 5, Col 62'.

2. Write a Python program to create a dictionary from two lists without losing duplicate values.

Sample lists: ['Class-V', 'Class-VI', 'Class-VII', 'Class-VIII'], [1, 2, 2, 3]

Expected Output: defaultdict(<class 'set'>, {'Class-VII': {2}, 'Class-VI': {2}, 'Class-VIII': {3}, 'Class-V': {1}})

Solution: -



The screenshot shows a Visual Studio Code window titled "Q02.py - Information-Technology-Workshop - Visual Studio Code". The editor has two tabs: "Q10.py" and "Q02.py". The "Q02.py" tab is active, showing the following Python code:

```
1 from collections import defaultdict
2 list1=['Class-V','Class-VI','Class-VII','Class-VIII']
3 list2=[1,2,2,3]
4 d=defaultdict(set)
5 for i in range(len(list1)):
6     d[list1[i]].add(list2[i])
7 print(d)
8
```

The bottom panel of the editor shows the "TERMINAL" tab with the following output:

```
defaultdict(<class 'set'>, {'Class-V': {1, 2}, 'Class-VII': {2}, 'Class-VIII': {3}})
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q02.py"
defaultdict(<class 'set'>, {'Class-V': {1}, 'Class-VI': {2}, 'Class-VII': {2}, 'Class-VIII': {3}})
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$
```

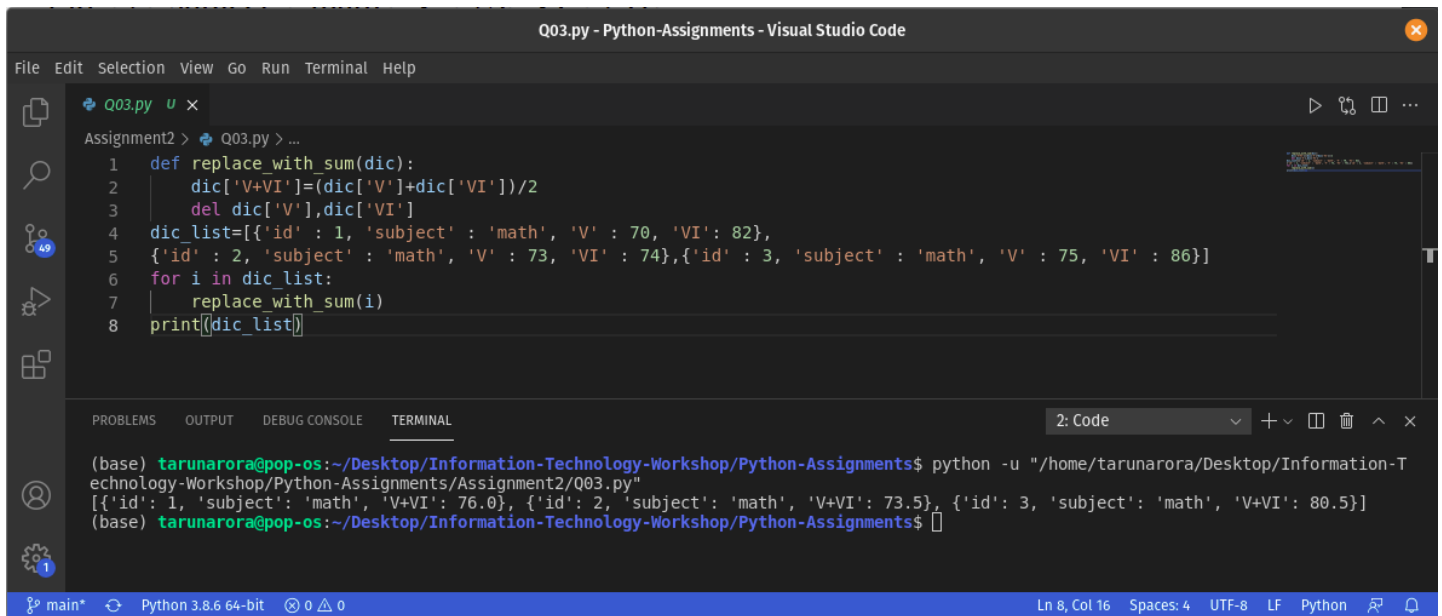
The status bar at the bottom indicates the file is "main*", the Python version is "Python 3.8.5 64-bit ('base': conda)", and the cursor is at "Ln 2, Col 27".

3. Write a Python program to replace dictionary values with their sum.

Example: Input: {'id' : 1, 'subject' : 'math', 'V' : 70, 'VI': 82},
 {'id' : 2, 'subject' : 'math', 'V' : 73, 'VI' : 74},
 {'id' : 3, 'subject' : 'math', 'V' : 75, 'VI' : 86}

Output: [{'subject': 'math', 'id': 1, 'V+VI': 76.0},
 {'subject': 'math', 'id': 2, 'V+VI': 73.5}
 {'subject': 'math', 'id': 3, 'V+VI': 80.5}]

Solution: -



The screenshot shows a Visual Studio Code window titled "Q03.py - Python-Assignments - Visual Studio Code". The editor displays a Python script named "Q03.py" with the following code:

```
1 def replace_with_sum(dic):
2     dic['V+VI']=(dic['V']+dic['VI'])/2
3     del dic['V'],dic['VI']
4 dic_list=[{'id' : 1, 'subject' : 'math', 'V' : 70, 'VI': 82},
5 {'id' : 2, 'subject' : 'math', 'V' : 73, 'VI' : 74},{ 'id' : 3, 'subject' : 'math', 'V' : 75, 'VI' : 86}]
6 for i in dic_list:
7     replace_with_sum(i)
8 print(dic_list)
```

Below the editor, the TERMINAL panel shows the command and output:

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q03.py"
[{'id': 1, 'subject': 'math', 'V+VI': 76.0}, {'id': 2, 'subject': 'math', 'V+VI': 73.5}, {'id': 3, 'subject': 'math', 'V+VI': 80.5}]
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments$
```

The status bar at the bottom indicates the file is "main*", the Python version is "Python 3.8.6 64-bit", and the cursor is at "Ln 8, Col 16".

4. Write a Python program to sort a tuple by its float element.

Sample data: [('item1', '12.20'), ('item2', '15.10'), ('item3', '24.5')]

Expected Output: [('item3', '24.5'), ('item2', '15.10'), ('item1', '12.20')]

Solution: -

```
Q04.py - Python-Assignments - Visual Studio Code
File Edit Selection View Go Run Terminal Help

Q04.py u x
Assignment2 > Q04.py > ...
1 def sort_by_float(e):
2     return float(e[1])
3 l=[('item1', '12.20'), ('item2', '15.10'), ('item3', '24.5')]
4 l.sort(reverse=True,key=sort_by_float)
5 print('Sorted List\n',l)

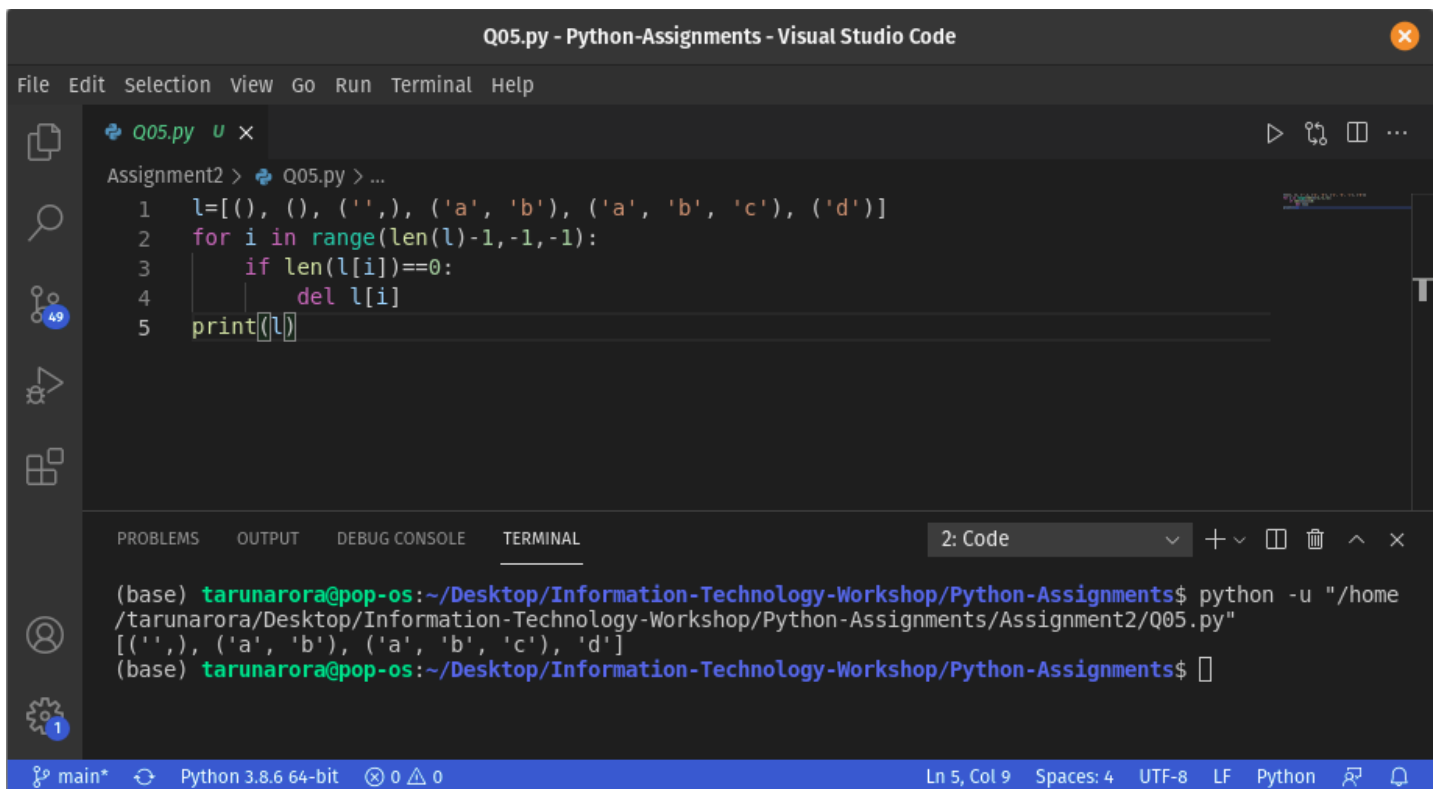
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Code
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q04.py"
Sorted List
[('item3', '24.5'), ('item2', '15.10'), ('item1', '12.20')]
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments$
```

5. Write a Python program to remove an empty tuple(s) from a list of tuples.

Sample data: `[(), (), ('',), ('a', 'b'), ('a', 'b', 'c'), ('d')]`

Expected output: `[('',), ('a', 'b'), ('a', 'b', 'c'), 'd']`

Solution: -



```
Q05.py - Python-Assignments - Visual Studio Code
File Edit Selection View Go Run Terminal Help

Q05.py u x
Assignment2 > Q05.py > ...
1 l=[(), (), ('',), ('a', 'b'), ('a', 'b', 'c'), ('d')]
2 for i in range(len(l)-1,-1,-1):
3     if len(l[i])==0:
4         del l[i]
5 print(l)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Code

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q05.py"
[('',), ('a', 'b'), ('a', 'b', 'c'), 'd']
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments$
```

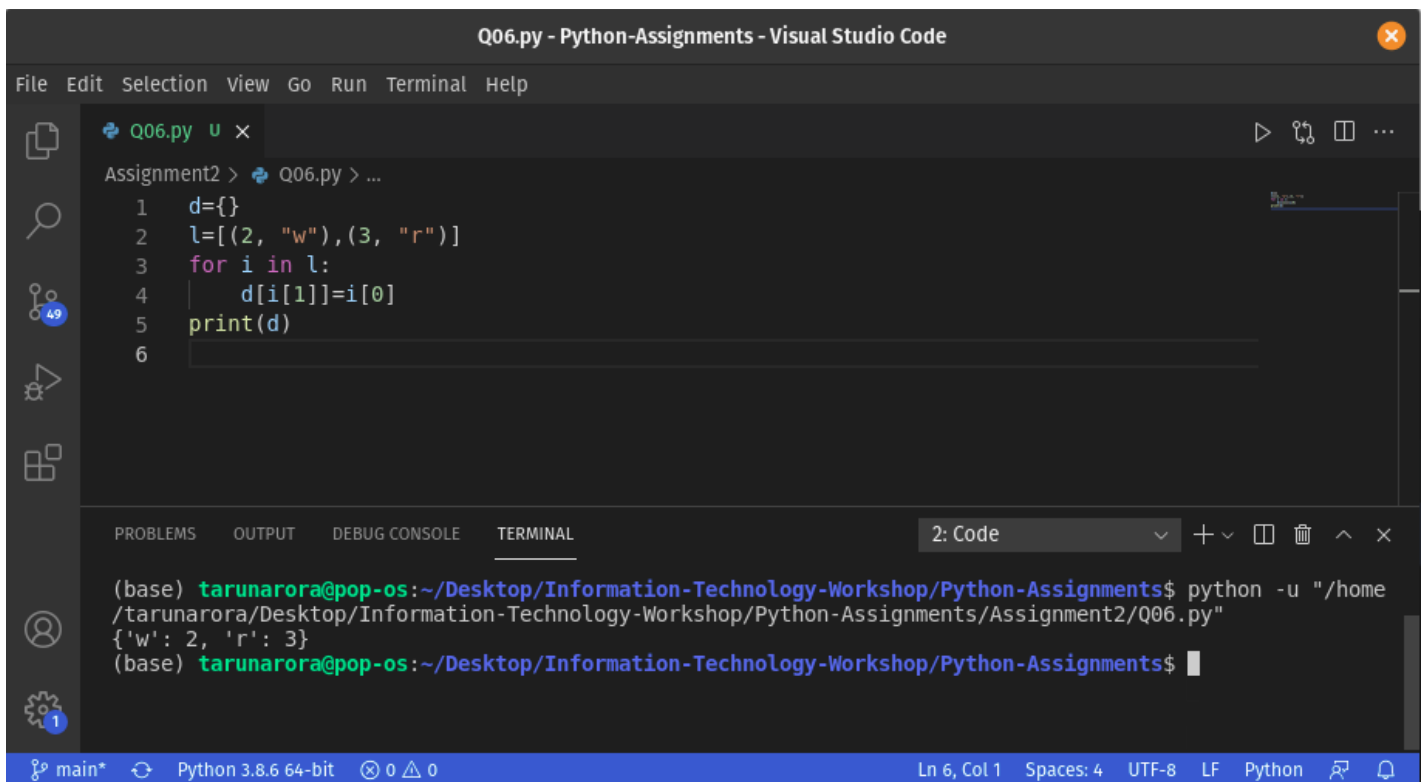
main* Python 3.8.6 64-bit 0 0 Ln 5, Col 9 Spaces: 4 UTF-8 LF Python

6. Write a Python program to convert a list of tuples into a dictionary.

Example: Input: ((2, "w"), (3, "r"))

Output: {'w': 2, 'r': 3}

Solution: -



```
Q06.py - Python-Assignments - Visual Studio Code
File Edit Selection View Go Run Terminal Help

Assignment2 > Q06.py > ...
1  d={}
2  l=[(2, "w"), (3, "r")]
3  for i in l:
4      d[i[1]]=i[0]
5  print(d)
6

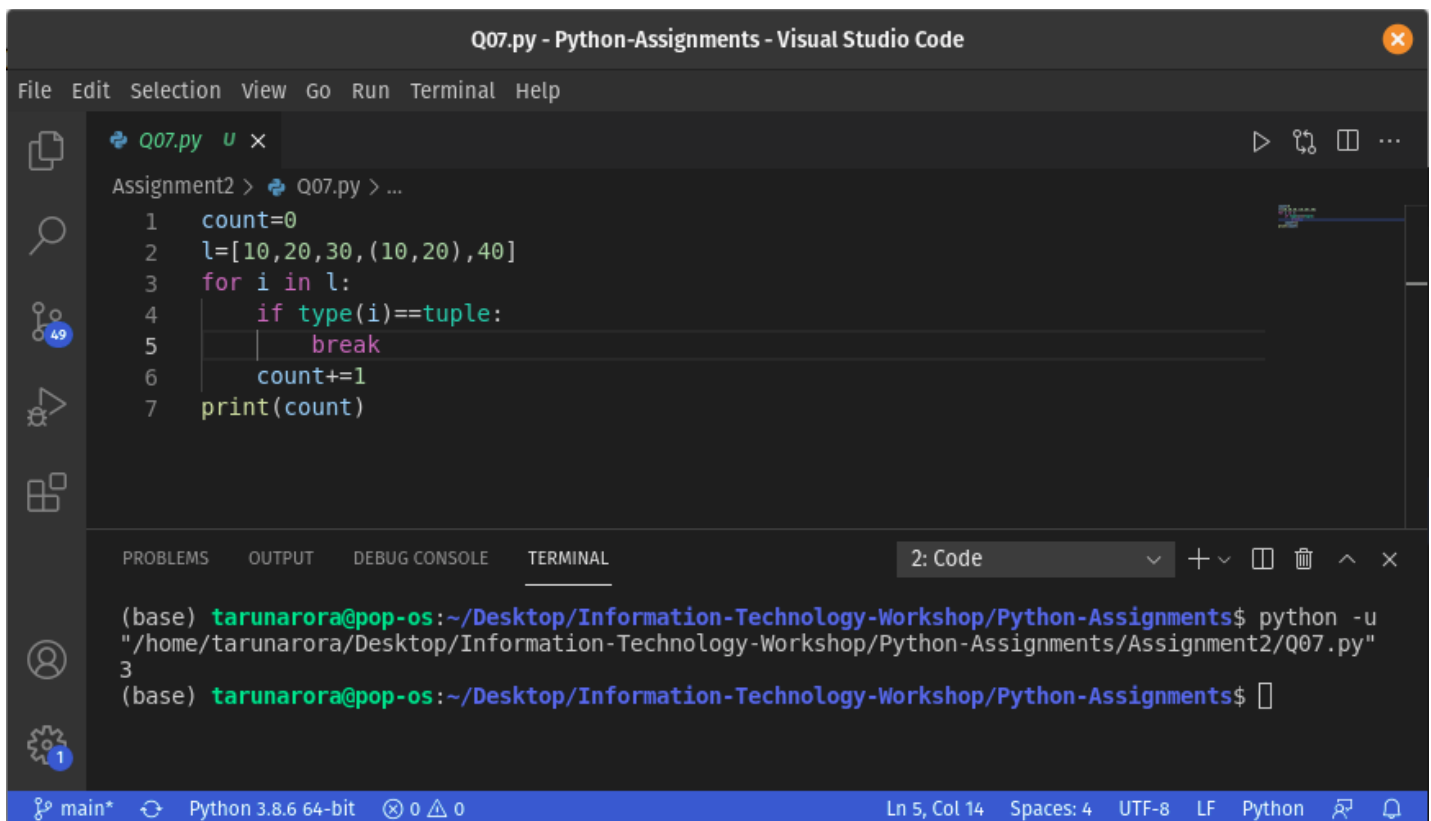
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q06.py"
{'w': 2, 'r': 3}
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments$
```

7. Write a Python program to count the elements in a list until an element is a tuple.

Example: Input: [10,20,30, (10,20),40]

Output: 3

Solution: -



The screenshot shows the Visual Studio Code interface with a file named 'Q07.py' open. The code in the editor is as follows:

```
1 count=0
2 l=[10,20,30,(10,20),40]
3 for i in l:
4     if type(i)==tuple:
5         break
6     count+=1
7 print(count)
```

Below the code editor, the 'TERMINAL' panel is active, showing the command to run the script and its output:

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments$ python -u
"/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q07.py"
3
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments$
```

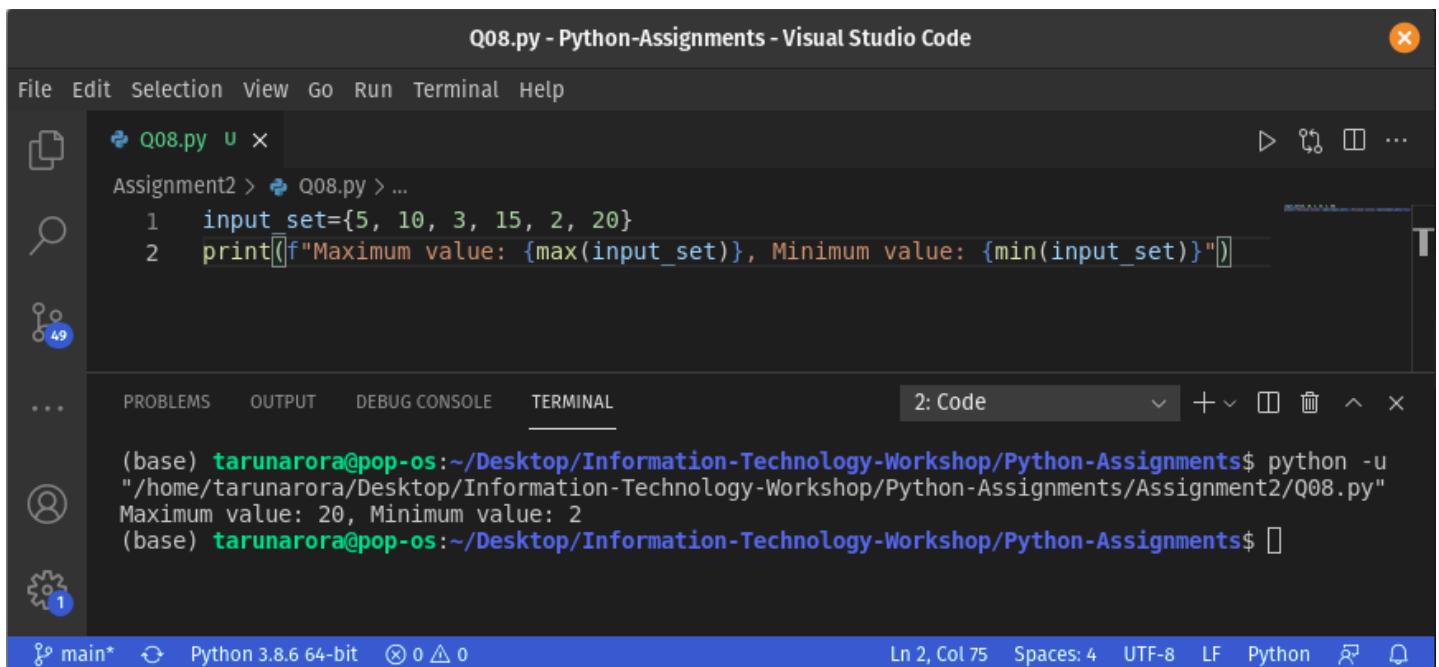
The status bar at the bottom indicates the current file is 'main*', the Python version is 'Python 3.8.6 64-bit', and the cursor is at 'Ln 5, Col 14'.

8. Write a Python program to find maximum and the minimum value in a set.

Example: Input: ([5, 10, 3, 15, 2, 20])

Output: Maximum value: 20, Minimum value: 2

Solution: -



The screenshot shows the Visual Studio Code interface with a file named 'Q08.py' open. The code in the editor is as follows:

```
1 input_set={5, 10, 3, 15, 2, 20}
2 print(f"Maximum value: {max(input_set)}, Minimum value: {min(input_set)}")
```

Below the code editor, the 'TERMINAL' panel is active, showing the command prompt output:

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments$ python -u
"/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q08.py"
Maximum value: 20, Minimum value: 2
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments$
```

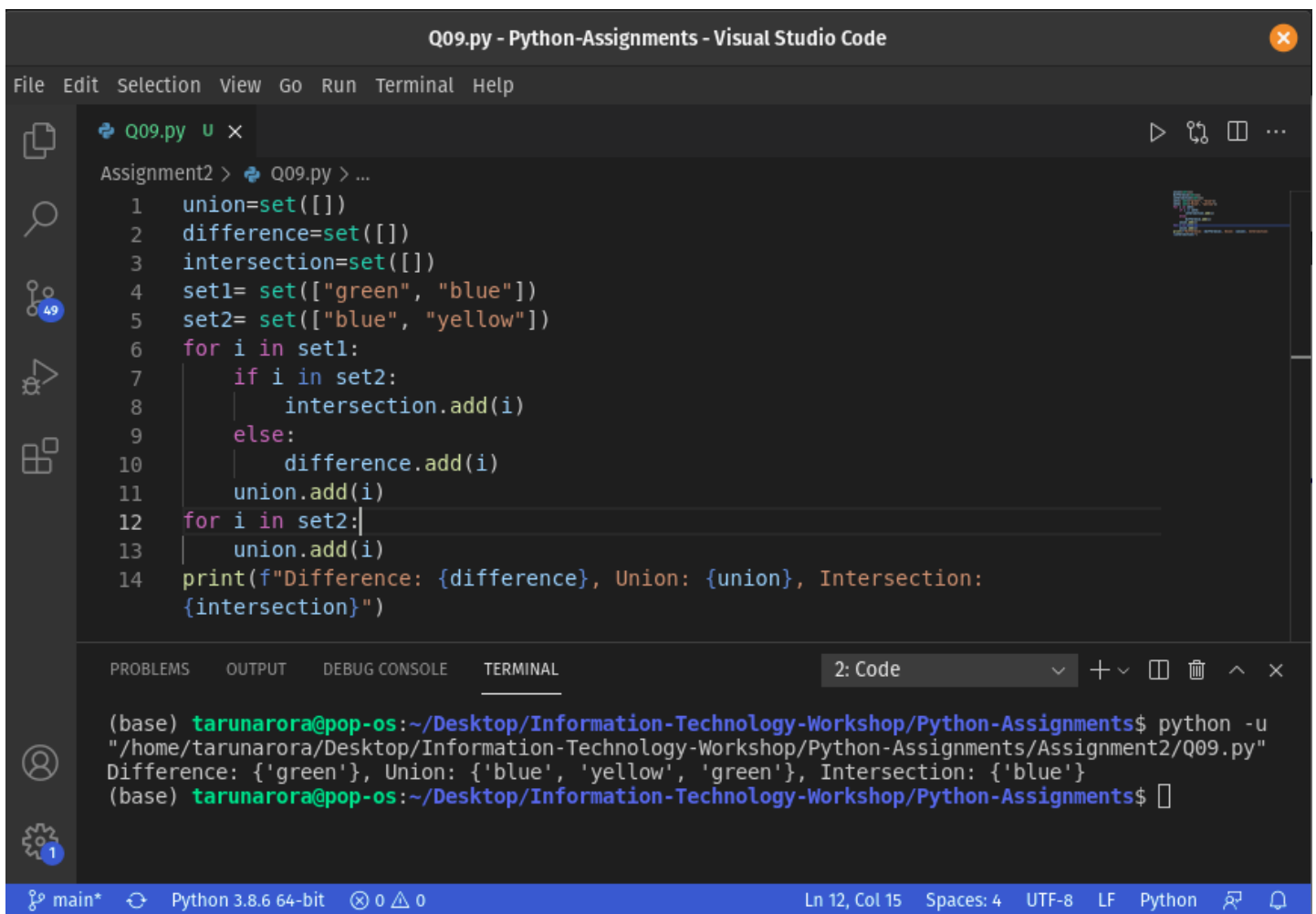
The status bar at the bottom indicates the file is 'main*', the Python version is 'Python 3.8.6 64-bit', and the cursor is at 'Ln 2, Col 75'.

9. Write a Python program to create set difference, union, and intersection of sets.

Example: Input: `set(["green", "blue"]), set(["blue", "yellow"])`

Output: Difference: {'blue'}, {'green'}, Union: {'yellow', 'green', 'blue'}, Intersection: {'blue'}

Solution: -



The screenshot displays the Visual Studio Code interface with a file named 'Q09.py'. The code defines three empty sets: 'union', 'difference', and 'intersection'. It then creates two sets: 'set1' containing 'green' and 'blue', and 'set2' containing 'blue' and 'yellow'. A loop iterates over 'set1', adding elements to 'intersection' if they are also in 'set2', or to 'difference' otherwise. Another loop iterates over 'set2', adding elements to 'union'. Finally, it prints the results for 'Difference', 'Union', and 'Intersection'.

```
1 union=set([])
2 difference=set([])
3 intersection=set([])
4 set1= set(["green", "blue"])
5 set2= set(["blue", "yellow"])
6 for i in set1:
7     if i in set2:
8         intersection.add(i)
9     else:
10        difference.add(i)
11        union.add(i)
12 for i in set2:
13     union.add(i)
14 print(f"Difference: {difference}, Union: {union}, Intersection: {intersection}")
```

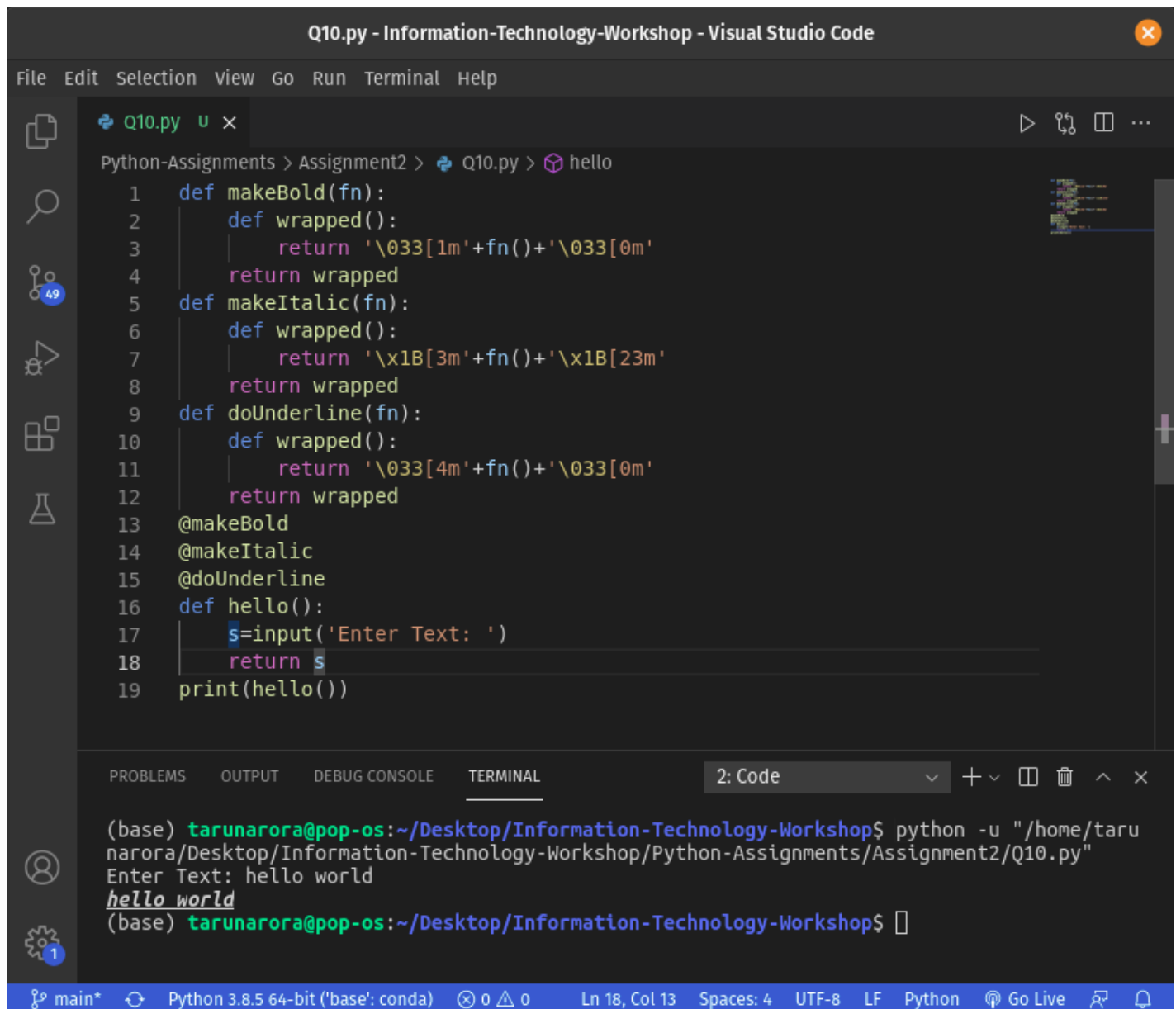
The terminal output shows the execution of the program, resulting in: Difference: {'green'}, Union: {'blue', 'yellow', 'green'}, Intersection: {'blue'}. The status bar at the bottom indicates the file is 'main*', the Python version is '3.8.6 64-bit', and the cursor is at 'Ln 12, Col 15'.

10. Write a Python program to make a chain of function decorators (bold, italic, underline etc.).

Example: Input: hello world

Output: hello world

Solution: -



The screenshot shows a Visual Studio Code window titled "Q10.py - Information-Technology-Workshop - Visual Studio Code". The editor displays a Python file named "Q10.py" with the following code:

```
1 def makeBold(fn):
2     def wrapped():
3         return '\033[1m'+fn()+'\033[0m'
4     return wrapped
5 def makeItalic(fn):
6     def wrapped():
7         return '\x1B[3m'+fn()+'\x1B[23m'
8     return wrapped
9 def doUnderline(fn):
10    def wrapped():
11        return '\033[4m'+fn()+'\033[0m'
12    return wrapped
13 @makeBold
14 @makeItalic
15 @doUnderline
16 def hello():
17     s=input('Enter Text: ')
18     return s
19 print(hello())
```

The terminal at the bottom shows the execution of the program:

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q10.py"
Enter Text: hello world
hello world
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$
```

The status bar at the bottom indicates the file is "main*", the Python version is "Python 3.8.5 64-bit ('base': conda)", and the cursor is at "Ln 18, Col 13".

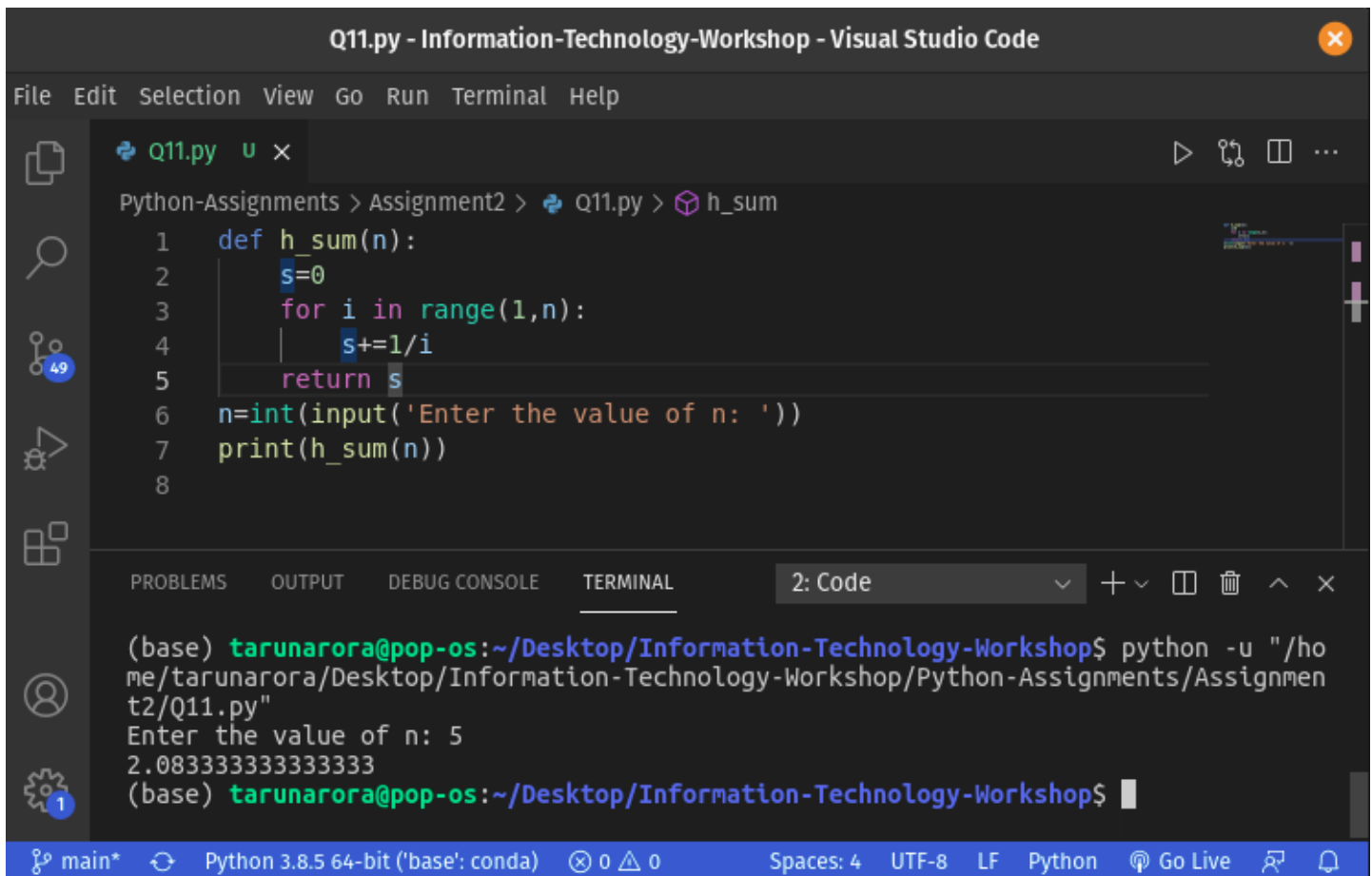
11. Write a Python program to calculate the harmonic sum of n-1.

Note: The harmonic sum is the sum of reciprocals of the positive integers.

Example :

$$1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \dots$$

Solution: -



```
Q11.py - Information-Technology-Workshop - Visual Studio Code
File Edit Selection View Go Run Terminal Help

Python-Assignments > Assignment2 > Q11.py > h_sum
1 def h_sum(n):
2     s=0
3     for i in range(1,n):
4         s+=1/i
5     return s
6 n=int(input('Enter the value of n: '))
7 print(h_sum(n))
8

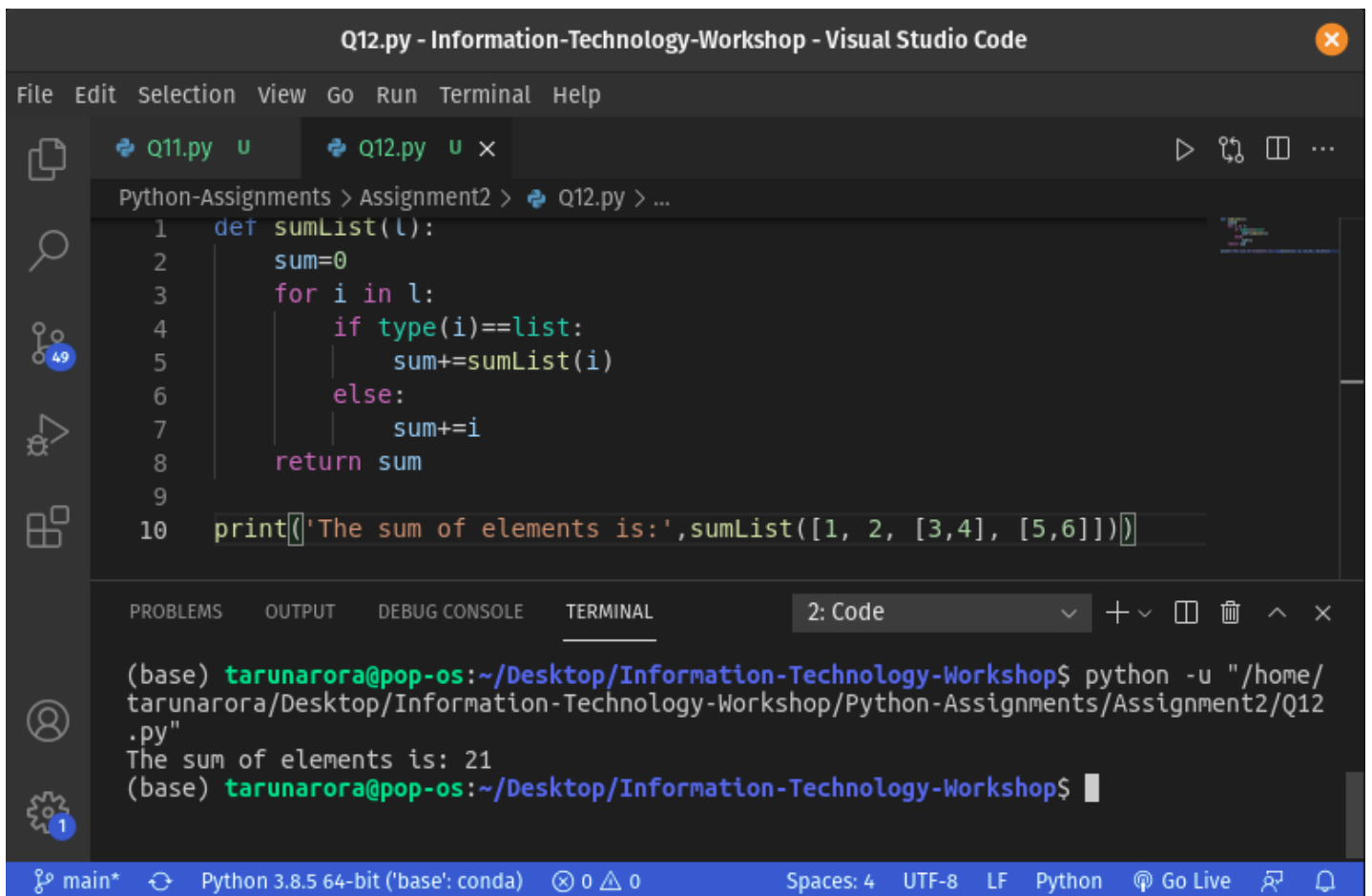
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Code
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q11.py"
Enter the value of n: 5
2.083333333333333
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$
```

12. Write a Python program of recursion list sum.

Test Data: [1, 2, [3,4], [5,6]]

Expected Result: 21

Solution: -



The screenshot shows the Visual Studio Code interface with a file named 'Q12.py' open. The code defines a recursive function 'sumList' that calculates the sum of elements in a list, including nested lists. The test data '[1, 2, [3,4], [5,6]]' is passed to the function, and the output 'The sum of elements is: 21' is printed. The terminal window at the bottom shows the command to run the script and the resulting output.

```
Q12.py - Information-Technology-Workshop - Visual Studio Code
File Edit Selection View Go Run Terminal Help
Python-Assignments > Assignment2 > Q12.py > ...
1 def sumList(l):
2     sum=0
3     for i in l:
4         if type(i)==list:
5             sum+=sumList(i)
6         else:
7             sum+=i
8     return sum
9
10 print('The sum of elements is:',sumList([1, 2, [3,4], [5,6]]))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Code

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q12.py"
The sum of elements is: 21
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$
```

main* Python 3.8.5 64-bit ('base': conda) 0 0 Spaces: 4 UTF-8 LF Python Go Live

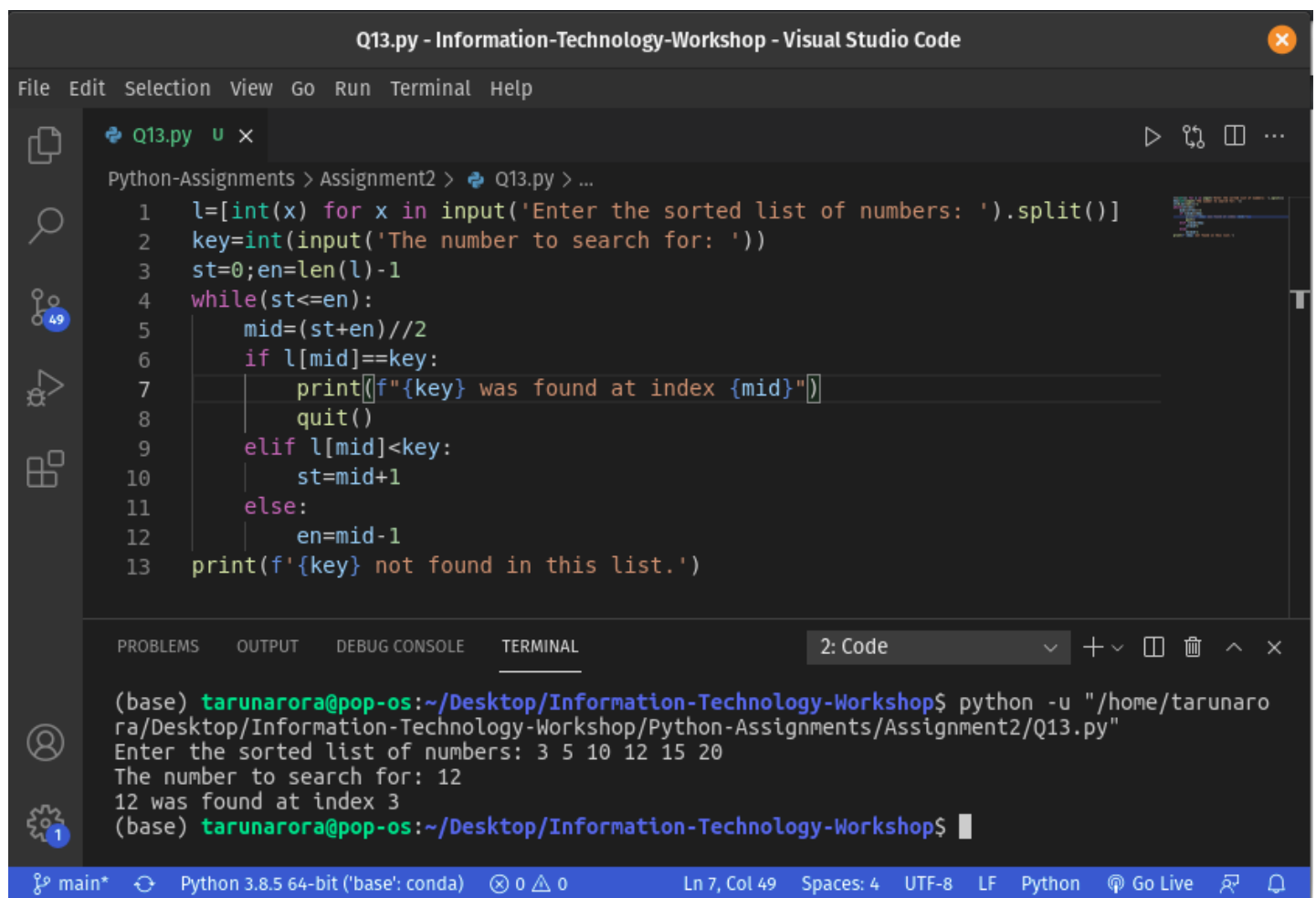
13. Write a Python program for binary search.

Example: Enter the sorted list of numbers: 3 5 10 12 15 20

The number to search for: 12

12 was found at index 3

Solution: -



The screenshot displays the Visual Studio Code interface. The editor window, titled 'Q13.py - Information-Technology-Workshop - Visual Studio Code', contains a Python script for binary search. The code is as follows:

```
1 l=[int(x) for x in input('Enter the sorted list of numbers: ').split()]
2 key=int(input('The number to search for: '))
3 st=0;en=len(l)-1
4 while(st<=en):
5     mid=(st+en)//2
6     if l[mid]==key:
7         print(f'{key} was found at index {mid}')
8         quit()
9     elif l[mid]<key:
10        st=mid+1
11    else:
12        en=mid-1
13 print(f'{key} not found in this list.')
```

Below the editor, the 'TERMINAL' panel shows the execution of the program. The user enters the sorted list '3 5 10 12 15 20' and the number to search for '12'. The program outputs '12 was found at index 3'.

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunaro
ra/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q13.py"
Enter the sorted list of numbers: 3 5 10 12 15 20
The number to search for: 12
12 was found at index 3
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$
```

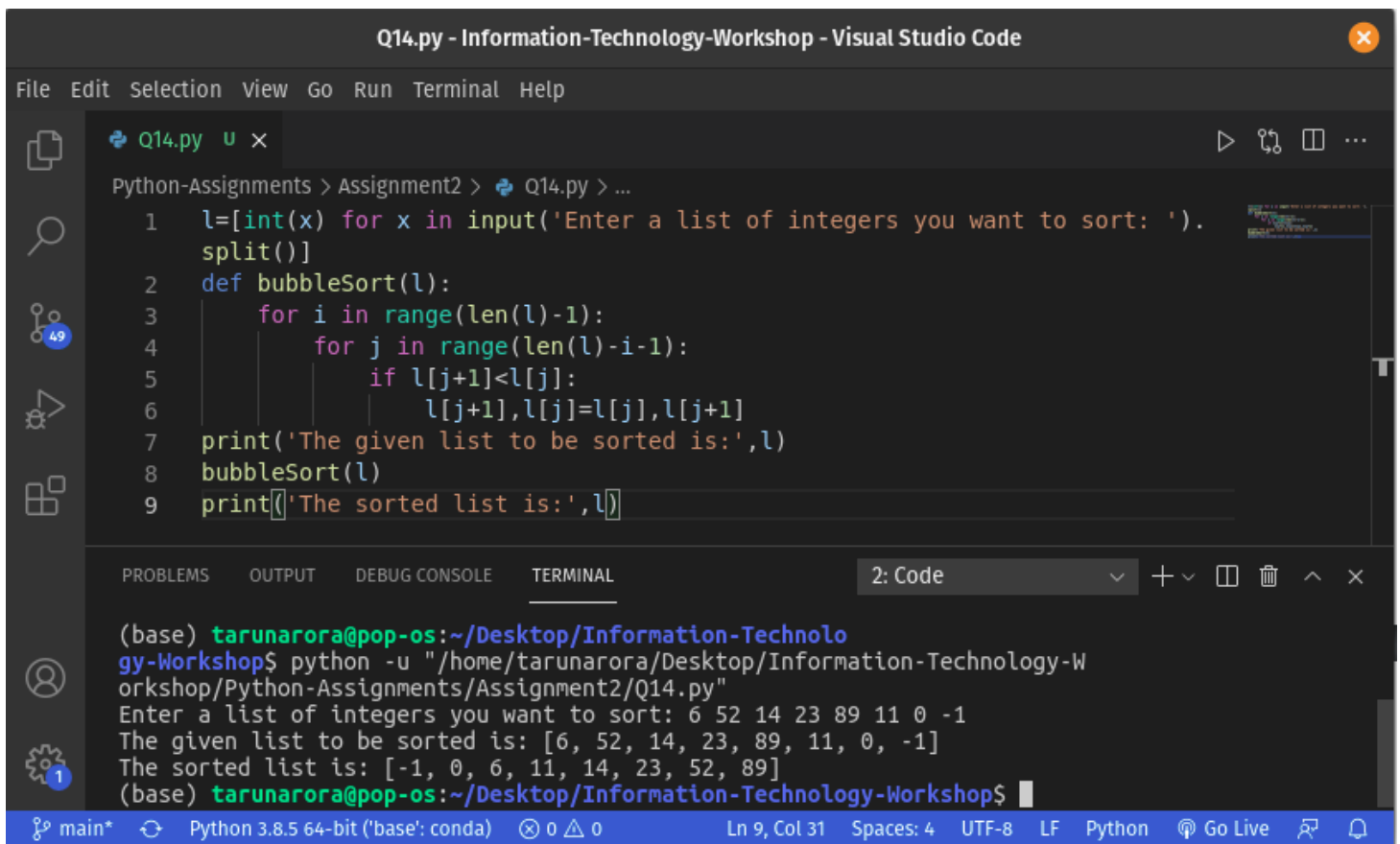
The status bar at the bottom indicates the file is 'main*', the Python version is 'Python 3.8.5 64-bit ('base': conda)', and the cursor is at 'Ln 7, Col 49'.

14. Write a Python program to sort a list of elements using the bubble sort algorithm.

Example: Sample Data: [14, 46, 43, 27, 57, 41, 45, 21, 70]

Expected Result: [14, 21, 27, 41, 43, 45, 46, 57, 70]

Solution: -



The screenshot shows a Visual Studio Code window titled "Q14.py - Information-Technology-Workshop - Visual Studio Code". The editor displays a Python file named "Q14.py" with the following code:

```
1 l=[int(x) for x in input('Enter a list of integers you want to sort: ').split()]
2 def bubbleSort(l):
3     for i in range(len(l)-1):
4         for j in range(len(l)-i-1):
5             if l[j+1]<l[j]:
6                 l[j+1],l[j]=l[j],l[j+1]
7 print('The given list to be sorted is:',l)
8 bubbleSort(l)
9 print('The sorted list is:',l)
```

The terminal output shows the execution of the program:

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q14.py"
Enter a list of integers you want to sort: 6 52 14 23 89 11 0 -1
The given list to be sorted is: [6, 52, 14, 23, 89, 11, 0, -1]
The sorted list is: [-1, 0, 6, 11, 14, 23, 52, 89]
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$
```

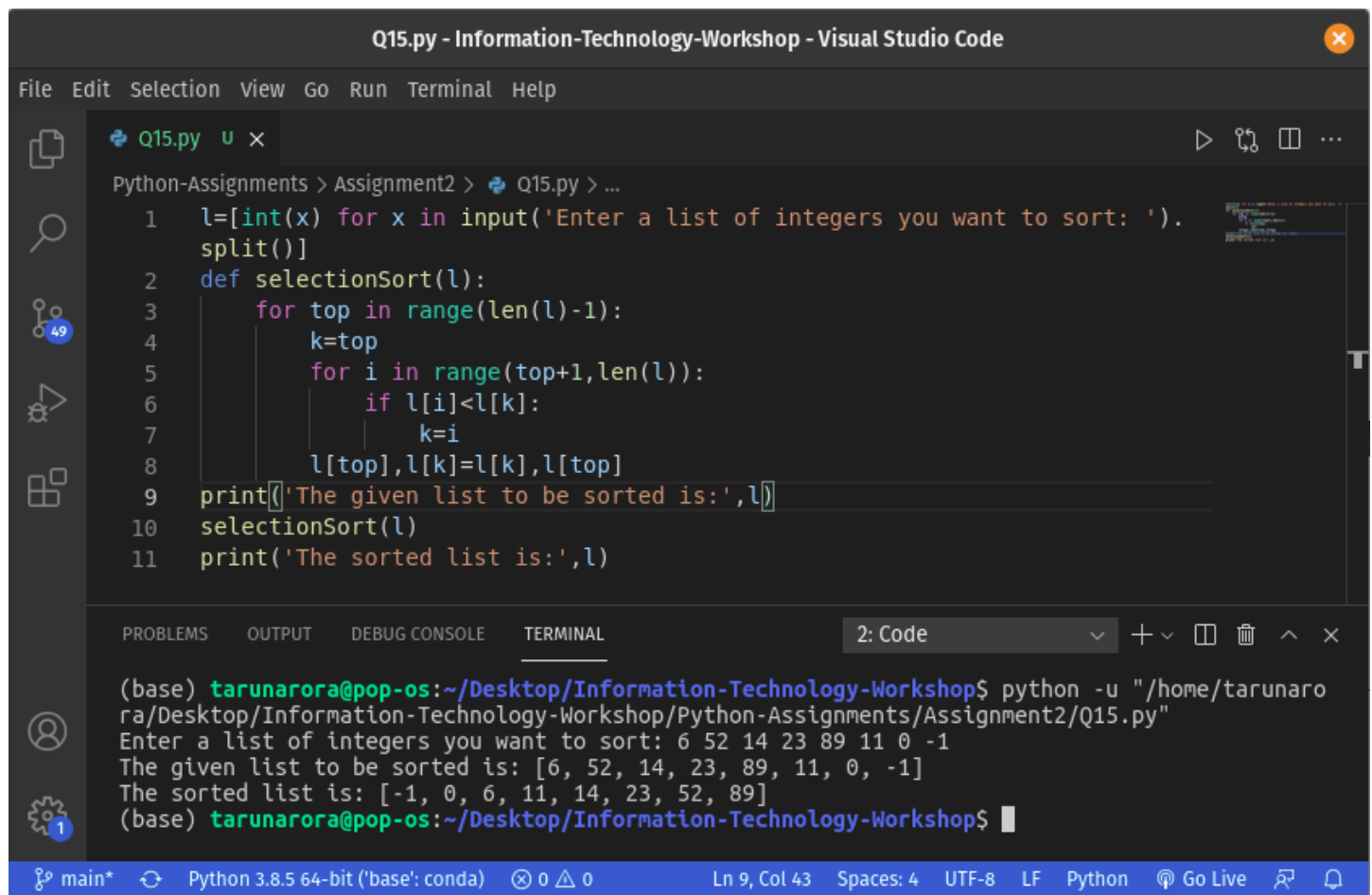
The status bar at the bottom indicates the current file is "main*", the Python version is "Python 3.8.5 64-bit ('base': conda)", and the cursor is at "Ln 9, Col 31".

15. Write a Python program to sort a list of elements using the selection sort algorithm.

Example: Sample Data: [14, 46, 43, 27, 57, 41, 45, 21, 70]

Expected Result: [14, 21, 27, 41, 43, 45, 46, 57, 70]

Solution: -



The screenshot shows a Visual Studio Code window titled "Q15.py - Information-Technology-Workshop - Visual Studio Code". The editor displays a Python file named "Q15.py" with the following code:

```
1 l=[int(x) for x in input('Enter a list of integers you want to sort: ').split()]
2 def selectionSort(l):
3     for top in range(len(l)-1):
4         k=top
5         for i in range(top+1,len(l)):
6             if l[i]<l[k]:
7                 k=i
8         l[top],l[k]=l[k],l[top]
9 print('The given list to be sorted is:',l)
10 selectionSort(l)
11 print('The sorted list is:',l)
```

The terminal at the bottom shows the execution of the program:

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q15.py"
Enter a list of integers you want to sort: 6 52 14 23 89 11 0 -1
The given list to be sorted is: [6, 52, 14, 23, 89, 11, 0, -1]
The sorted list is: [-1, 0, 6, 11, 14, 23, 52, 89]
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$
```

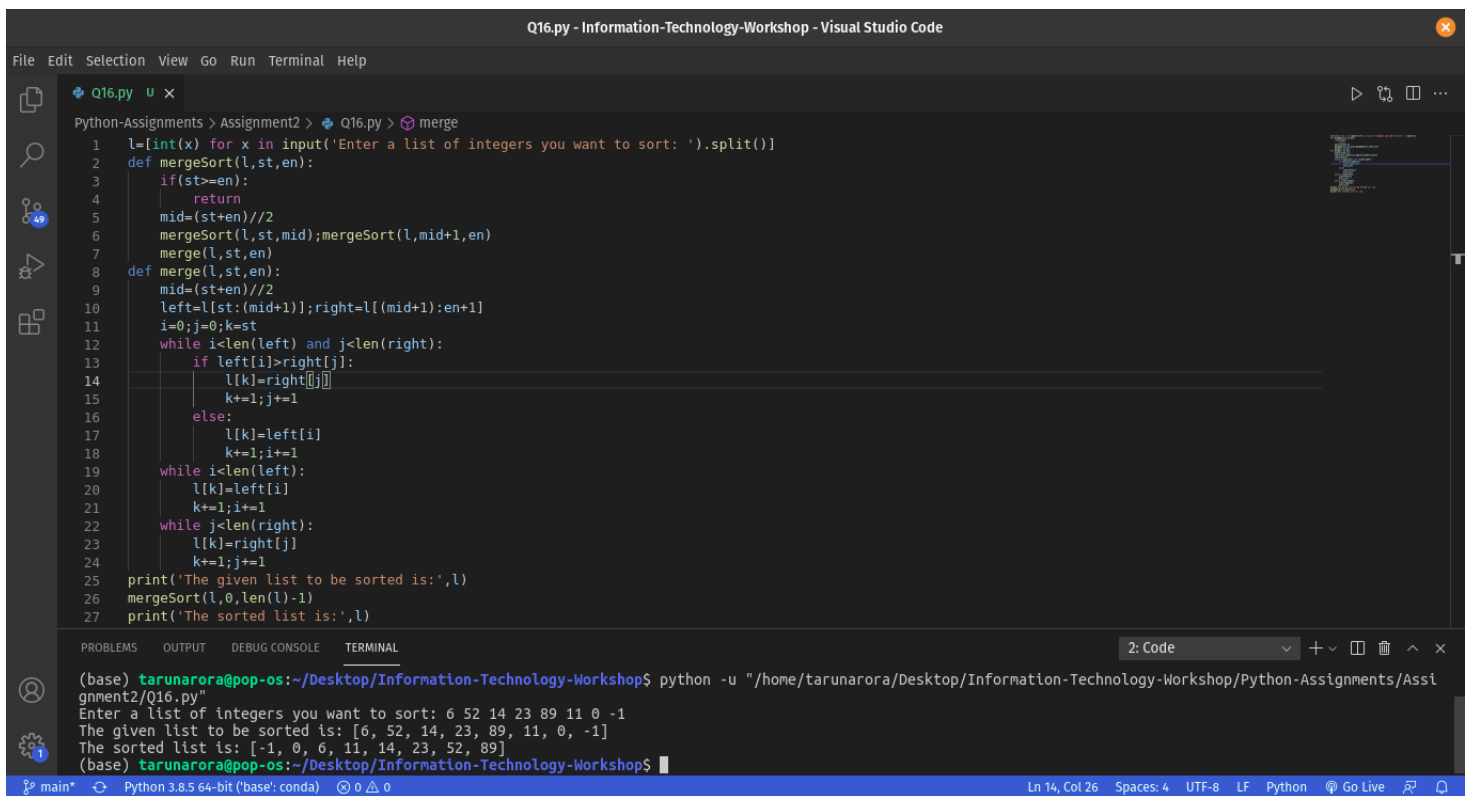
The status bar at the bottom indicates the current file is "main*", the Python version is "Python 3.8.5 64-bit ('base': conda)", and the cursor is at "Ln 9, Col 43".

16. Write a Python program to sort a list of elements using the merge sort algorithm.

Example: Split Sample Data: [14, 46, 43, 27, 57, 41, 45, 21, 70]

Merge and Sort(Expected Result): [14, 21, 27, 41, 43, 45, 46, 57, 70]

Solution: -



The screenshot shows a Visual Studio Code window titled "Q16.py - Information-Technology-Workshop - Visual Studio Code". The editor displays a Python program for merge sort. The code defines a `mergeSort` function that recursively splits a list and a `merge` function that merges the sorted sublists. The program prompts the user to enter a list of integers, sorts them, and prints the result.

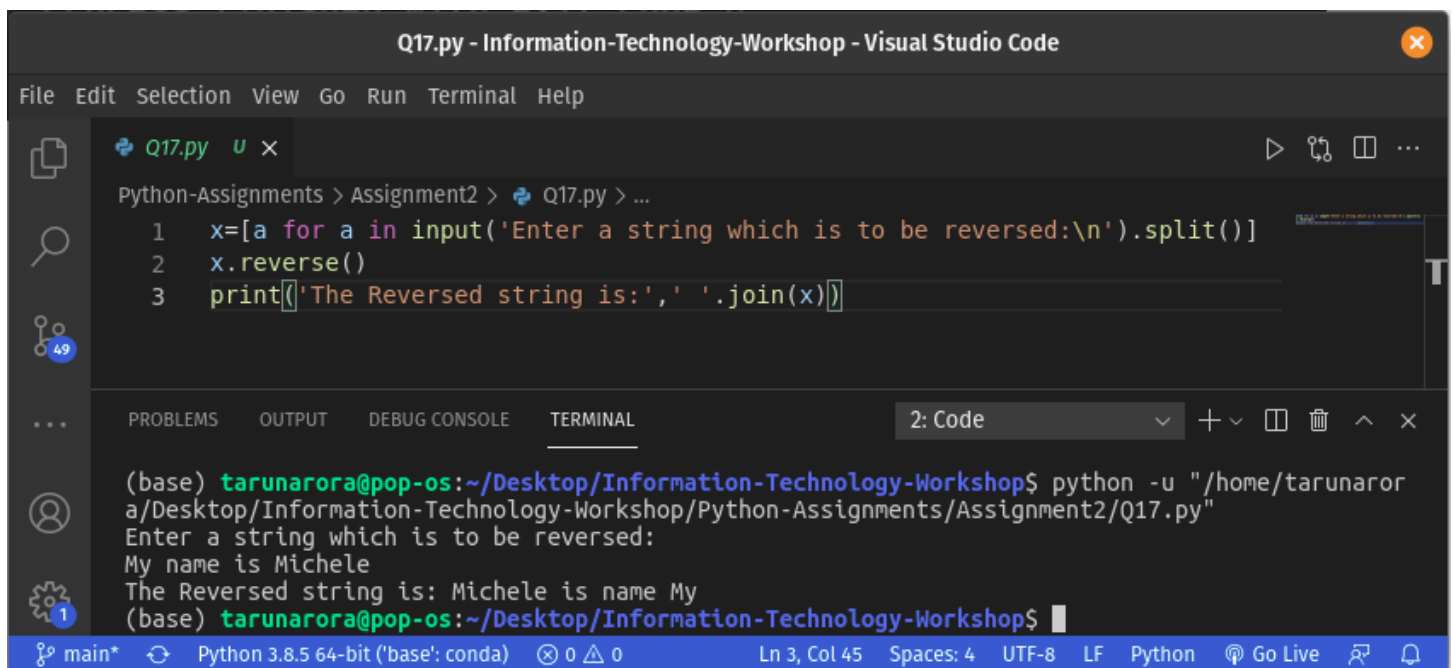
```
1 l=[int(x) for x in input('Enter a list of integers you want to sort: ').split()]
2 def mergeSort(l,st,en):
3     if(st>=en):
4         return
5     mid=(st+en)//2
6     mergeSort(l,st,mid);mergeSort(l,mid+1,en)
7     merge(l,st,en)
8 def merge(l,st,en):
9     mid=(st+en)//2
10    left=l[st:(mid+1)];right=l[(mid+1):en+1]
11    i=0;j=0;k=st
12    while i<len(left) and j<len(right):
13        if left[i]>right[j]:
14            l[k]=right[j]
15            k+=1;j+=1
16        else:
17            l[k]=left[i]
18            k+=1;i+=1
19    while i<len(left):
20        l[k]=left[i]
21        k+=1;i+=1
22    while j<len(right):
23        l[k]=right[j]
24        k+=1;j+=1
25    print('The given list to be sorted is:',l)
26    mergeSort(l,0,len(l)-1)
27    print('The sorted list is:',l)
```

The terminal output shows the program being executed. The user enters the list `6 52 14 23 89 11 0 -1`. The program outputs the unsorted list `[6, 52, 14, 23, 89, 11, 0, -1]` and the sorted list `[-1, 0, 6, 11, 14, 23, 52, 89]`.

17. Write a Python program using functions that asks the user for a long string containing multiple words. Print back to the user the same string, except with the words in backwards order.

For example, say I type the string: My name is Michele; Then I would see the string: Michele is name My; shown back to me.

Solution: -



The screenshot shows the Visual Studio Code interface with a file named 'Q17.py' open. The code in the editor is as follows:

```
1 x=[a for a in input('Enter a string which is to be reversed:\n').split()]
2 x.reverse()
3 print('The Reversed string is:', ' '.join(x))
```

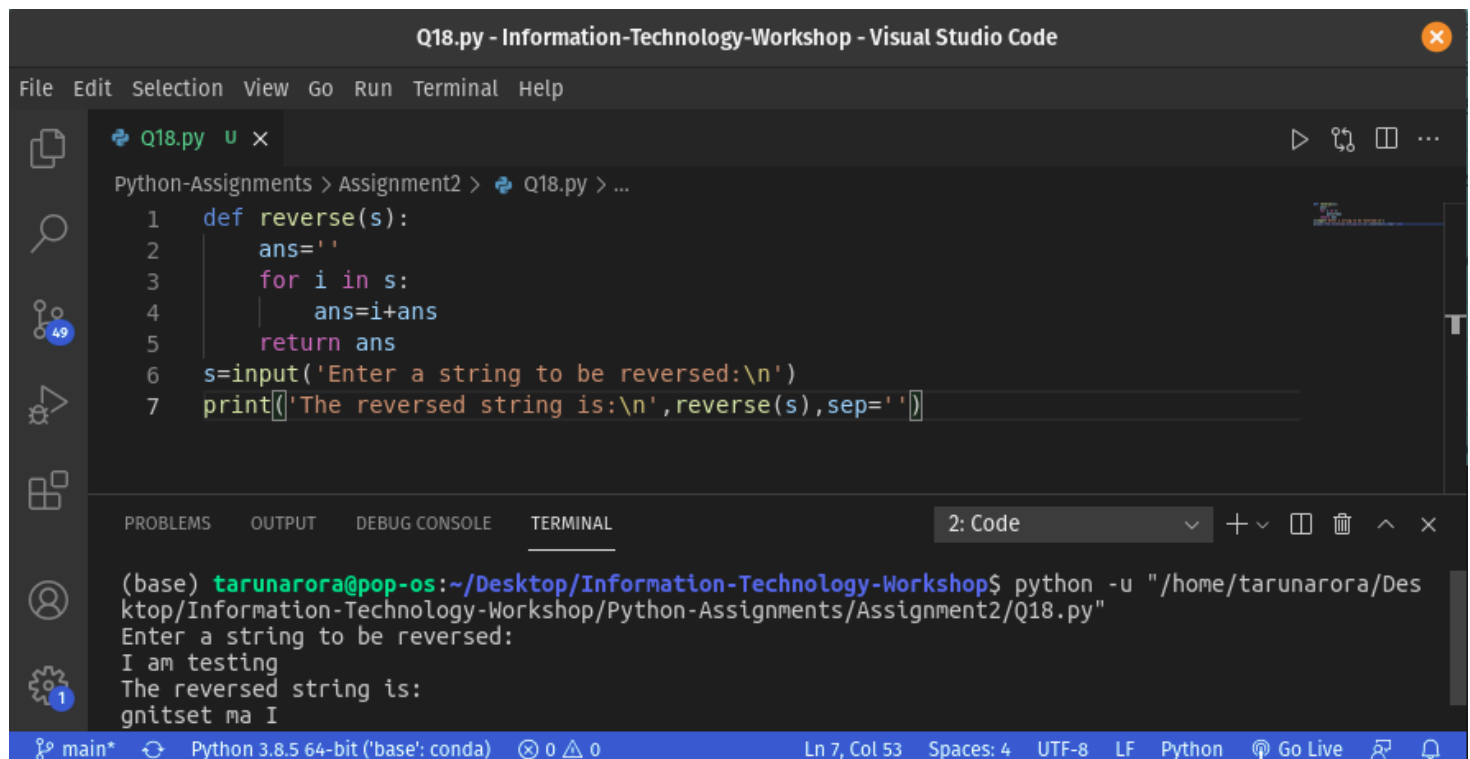
Below the code editor, the 'TERMINAL' panel is active, showing the execution of the program. The output is:

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q17.py"
Enter a string which is to be reversed:
My name is Michele
The Reversed string is: Michele is name My
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$
```

The status bar at the bottom indicates the current file is 'main*', the Python version is 'Python 3.8.5 64-bit (base: conda)', and the cursor is at 'Ln 3, Col 45'.

18. Define a function `reverse()` that computes the reversal of a string. For example, `reverse("I am testing")` should return the string "gnitset ma I".

Solution: -



The screenshot shows the Visual Studio Code interface with a file named `Q18.py` open. The code defines a `reverse(s)` function that iterates through each character of the string `s` and builds a new string `ans` by prepending each character. The main part of the script prompts the user to enter a string and prints the reversed string.

```
Python-Assignments > Assignment2 > Q18.py > ...
1 def reverse(s):
2     ans=''
3     for i in s:
4         ans=i+ans
5     return ans
6 s=input('Enter a string to be reversed:\n')
7 print('The reversed string is:\n',reverse(s),sep=' ')
```

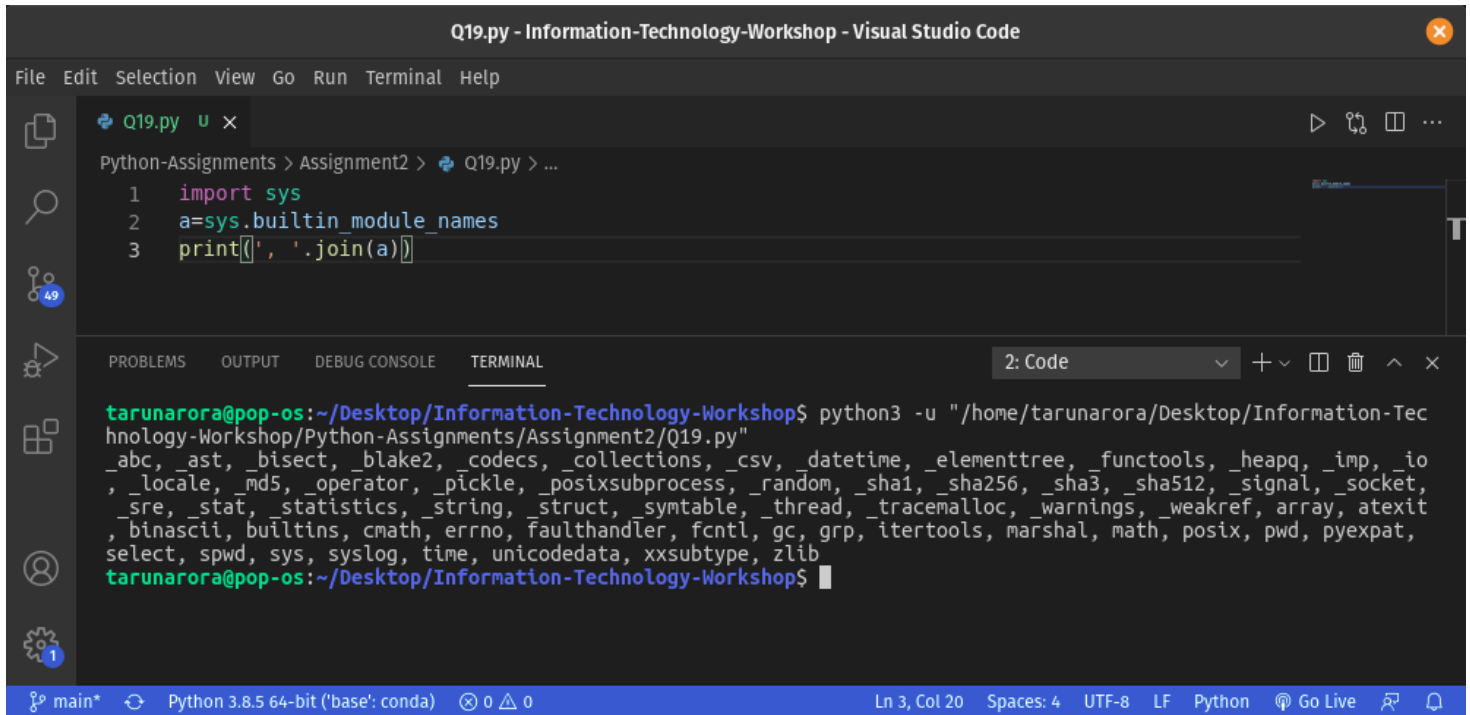
The terminal output shows the execution of the script, where the user entered "I am testing" and the program printed "gnitset ma I".

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q18.py"
Enter a string to be reversed:
I am testing
The reversed string is:
gnitset ma I
```

19. Write a Python program to find the available built-in modules.

Example: math, random, uuid, sys, syslog etc.

Solution: -



The screenshot shows the Visual Studio Code interface with a file named `Q19.py` open. The code in the editor is as follows:

```
1 import sys
2 a=sys.builtin_module_names
3 print(' '.join(a))
```

The terminal at the bottom shows the command being executed and the resulting output:

```
tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python3 -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q19.py"
_abc, _ast, _bisect, _blake2, _codecs, _collections, _csv, _datetime, _elementtree, _functools, _heapq, _imp, _io, _locale, _md5, _operator, _pickle, _posixsubprocess, _random, _sha1, _sha256, _sha3, _sha512, _signal, _socket, _sre, _stat, _statistics, _string, _struct, _symtable, _thread, _tracemalloc, _warnings, _weakref, array, atexit, binascii, builtins, cmath, errno, faulthandler, fcntl, gc, grp, itertools, marshal, math, posix, pwd, pyexpat, select, spwd, sys, syslog, time, unicodedata, xxsubtype, zlib
tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$
```

The status bar at the bottom indicates the file is `main*`, the Python version is `Python 3.8.5 64-bit ('base': conda)`, and the cursor is at `Ln 3, Col 20`.

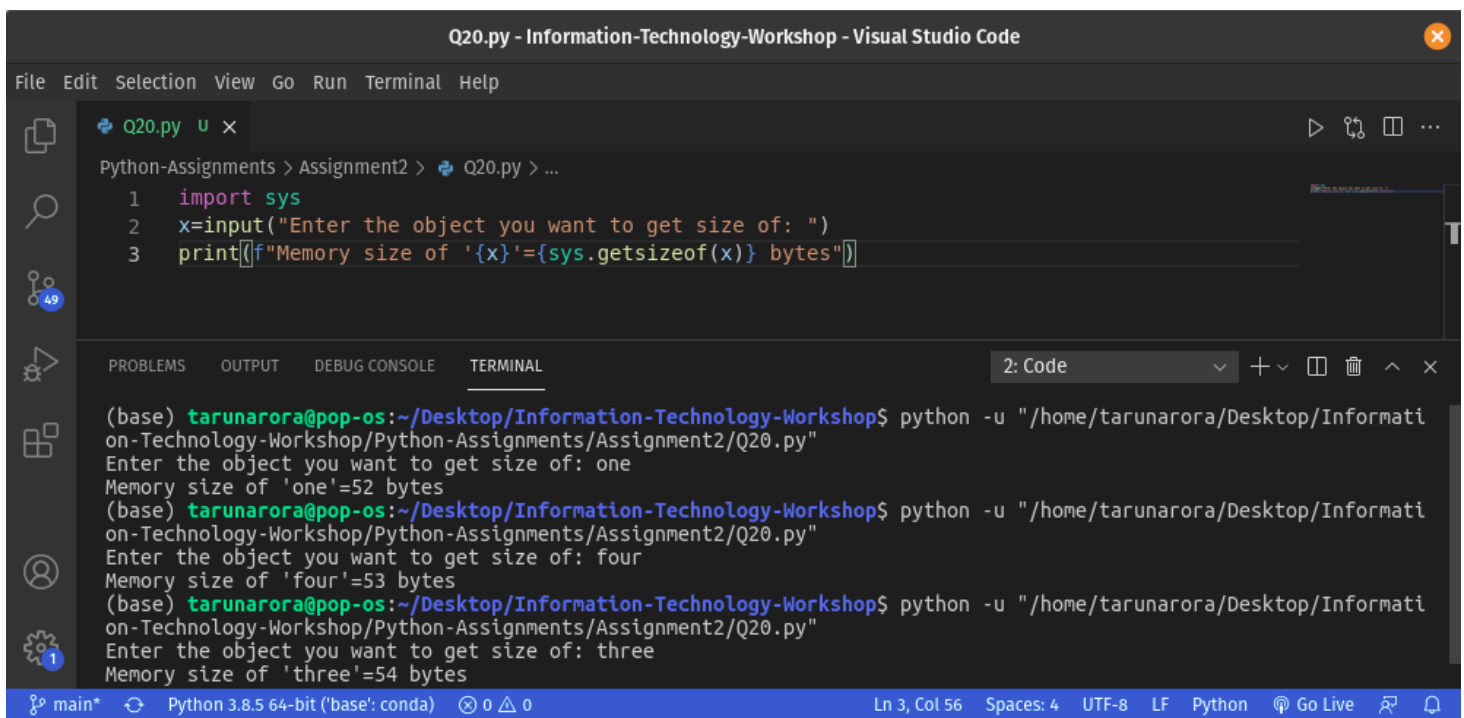
20. Write a Python program to get the size of an object in bytes by using module "sys".

Example: Memory size of 'one' = 52 bytes

Memory size of 'four' = 53 bytes

Memory size of 'three' = 54 bytes

Solution: -



The screenshot shows the Visual Studio Code interface with a file named 'Q20.py' open. The code in the editor is as follows:

```
1 import sys
2 x=input("Enter the object you want to get size of: ")
3 print(f"Memory size of '{x}'={sys.getsizeof(x)} bytes")
```

The terminal window at the bottom shows the execution of the program three times, each time with a different input and the corresponding output:

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q20.py"
Enter the object you want to get size of: one
Memory size of 'one'=52 bytes
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q20.py"
Enter the object you want to get size of: four
Memory size of 'four'=53 bytes
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q20.py"
Enter the object you want to get size of: three
Memory size of 'three'=54 bytes
```

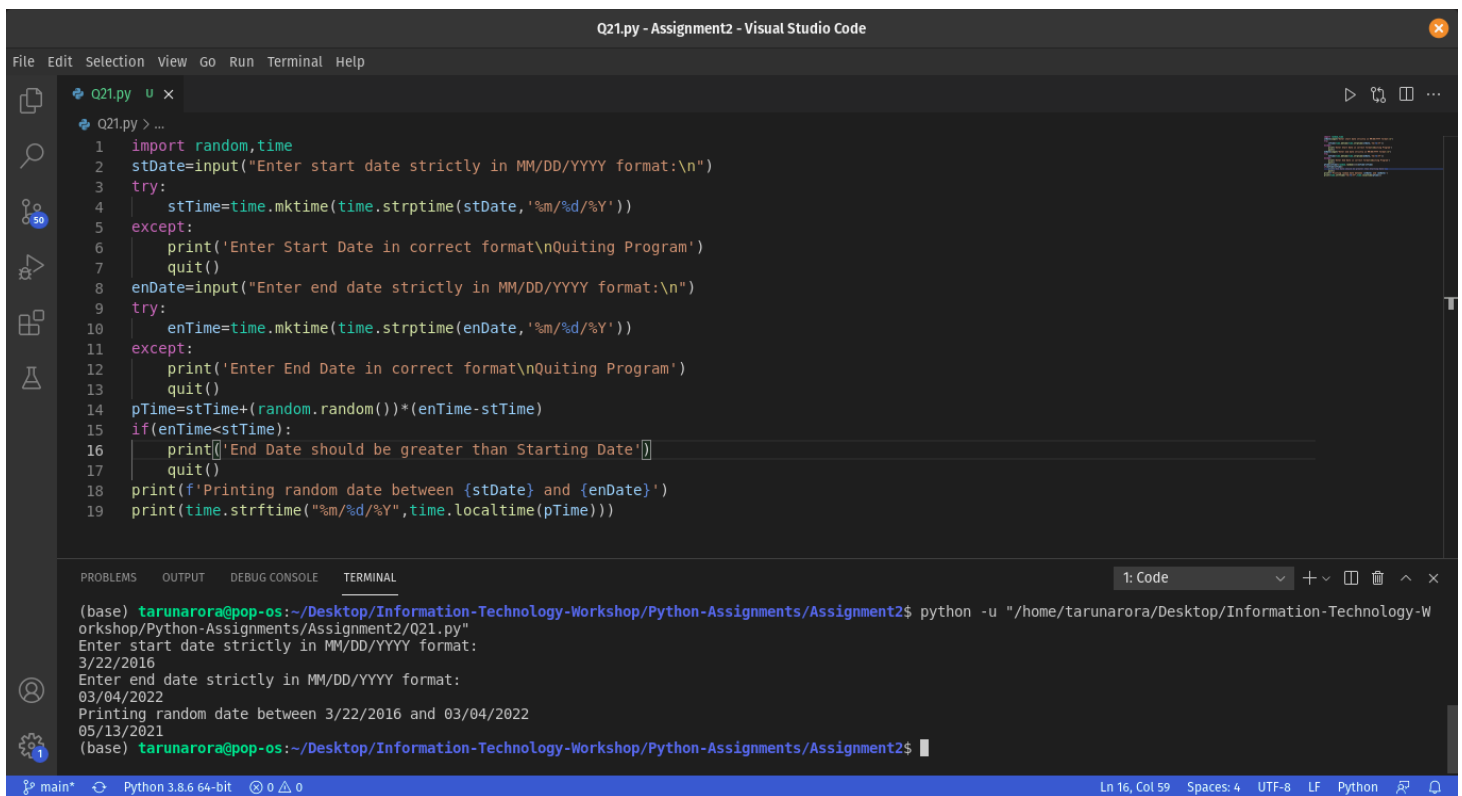
The status bar at the bottom indicates the current file is 'main*', the Python version is 'Python 3.8.5 64-bit (base: conda)', and the cursor is at 'Ln 3, Col 56'.

21. Using the module random and time in python generate a random date between given start and end dates.

Example: Printing random date between 1/1/2016 and 3/23/2018

Random Date = 02/25/2016

Solution: -



The screenshot shows a Visual Studio Code window titled "Q21.py - Assignment2 - Visual Studio Code". The editor displays a Python script named Q21.py. The script imports the 'random' and 'time' modules. It prompts the user to enter a start date in MM/DD/YYYY format. It then prompts for an end date in the same format. The script calculates a random date between the start and end dates using the formula: `pTime = stTime + (random.random() * (enTime - stTime))`. It includes error handling for incorrect date formats and ensures the end date is greater than the start date. Finally, it prints the generated random date.

```
1 import random,time
2 startDate=input("Enter start date strictly in MM/DD/YYYY format:\n")
3 try:
4     stTime=time.mktime(time.strptime(startDate,'%m/%d/%Y'))
5 except:
6     print('Enter Start Date in correct format\nQuiting Program')
7     quit()
8 endDate=input("Enter end date strictly in MM/DD/YYYY format:\n")
9 try:
10    enTime=time.mktime(time.strptime(endDate,'%m/%d/%Y'))
11 except:
12    print('Enter End Date in correct format\nQuiting Program')
13    quit()
14 pTime=stTime+(random.random()*(enTime-stTime))
15 if(enTime<stTime):
16     print('End Date should be greater than Starting Date')
17     quit()
18 print(f'Printing random date between {startDate} and {endDate}')
19 print(time.strftime("%m/%d/%Y",time.localtime(pTime)))
```

The terminal output shows the execution of the script. The user enters '3/22/2016' for the start date and '03/04/2022' for the end date. The script prints: "Printing random date between 3/22/2016 and 03/04/2022" followed by the generated random date "05/13/2021".

(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2\$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q21.py"
Enter start date strictly in MM/DD/YYYY format:
3/22/2016
Enter end date strictly in MM/DD/YYYY format:
03/04/2022
Printing random date between 3/22/2016 and 03/04/2022
05/13/2021
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2\$

22. Generate three random password string of length 10 with special characters, letters, and digits by using python modules (random and string).

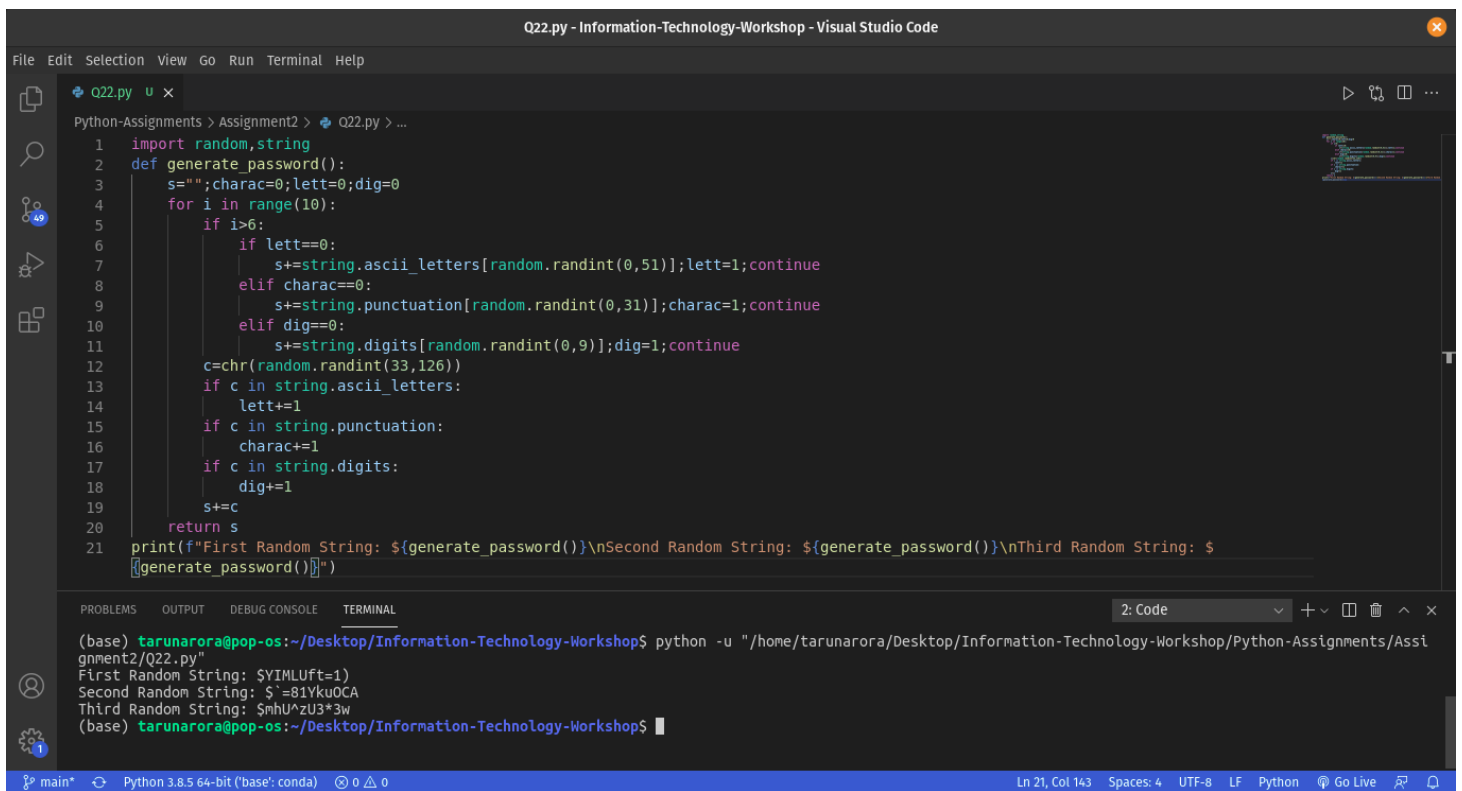
Example:

First Random String: yrjmcyi^VS

Second Random String: |}Hd]!^>~1

Third Random String: 3^a93@x=|Z

Solution: -



The screenshot shows a Visual Studio Code window titled "Q22.py - Information-Technology-Workshop - Visual Studio Code". The editor displays a Python script named "Q22.py" with the following code:

```
1 import random,string
2 def generate_password():
3     s="";charac=0;lett=0;dig=0
4     for i in range(10):
5         if i>6:
6             if lett==0:
7                 s+=string.ascii_letters[random.randint(0,51)];lett=1;continue
8             elif charac==0:
9                 s+=string.punctuation[random.randint(0,31)];charac=1;continue
10            elif dig==0:
11                s+=string.digits[random.randint(0,9)];dig=1;continue
12            c=chr(random.randint(33,126))
13            if c in string.ascii_letters:
14                lett+=1
15            if c in string.punctuation:
16                charac+=1
17            if c in string.digits:
18                dig+=1
19            s+=c
20        return s
21 print(f"First Random String: ${generate_password()}\nSecond Random String: ${generate_password()}\nThird Random String: ${generate_password()}")
```

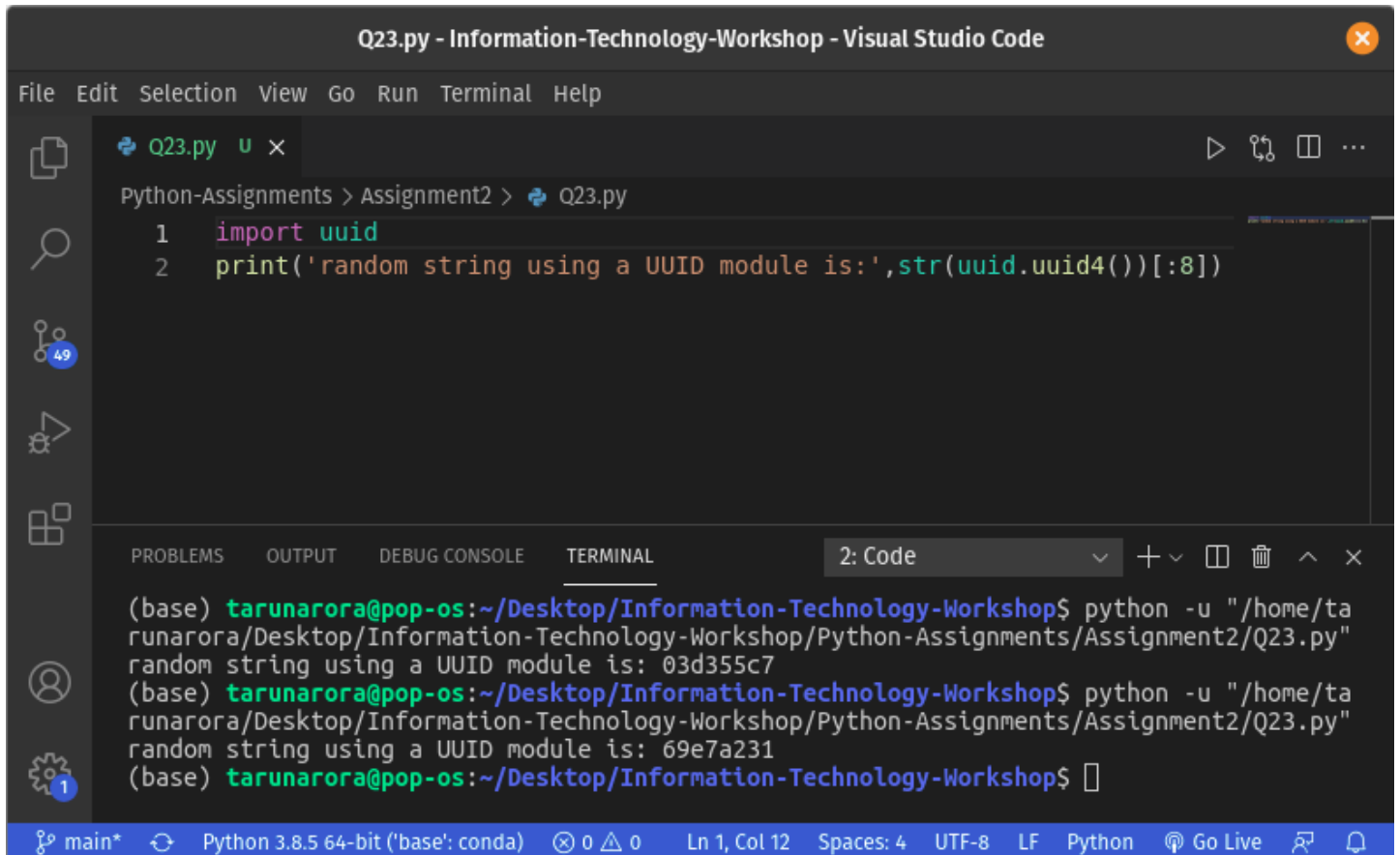
The terminal at the bottom shows the execution of the script using the command: `python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q22.py"`. The output is:

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q22.py"
First Random String: SYIMLUft=1)
Second Random String: $'81YkuOCA
Third Random String: $mhU^zU3*3w
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$
```

23. Write a python code using module "uuid" to generate universally unique secure random string id of length 8.

Example: random string using a UUID module is: 9C8E13FF

random string using a UUID module is: 9cb3561d



The screenshot shows the Visual Studio Code interface. The editor window displays a file named `Q23.py` with the following code:

```
1 import uuid
2 print('random string using a UUID module is:',str(uuid.uuid4())[:8])
```

The terminal window at the bottom shows the execution of the script using the command `python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q23.py"`. The output shows two different random strings generated by the script:

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q23.py"
random string using a UUID module is: 03d355c7
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q23.py"
random string using a UUID module is: 69e7a231
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$
```


24. Write a python code using module "random" to generate a 100 Lottery tickets and pick two lucky tickets from it as a winner.

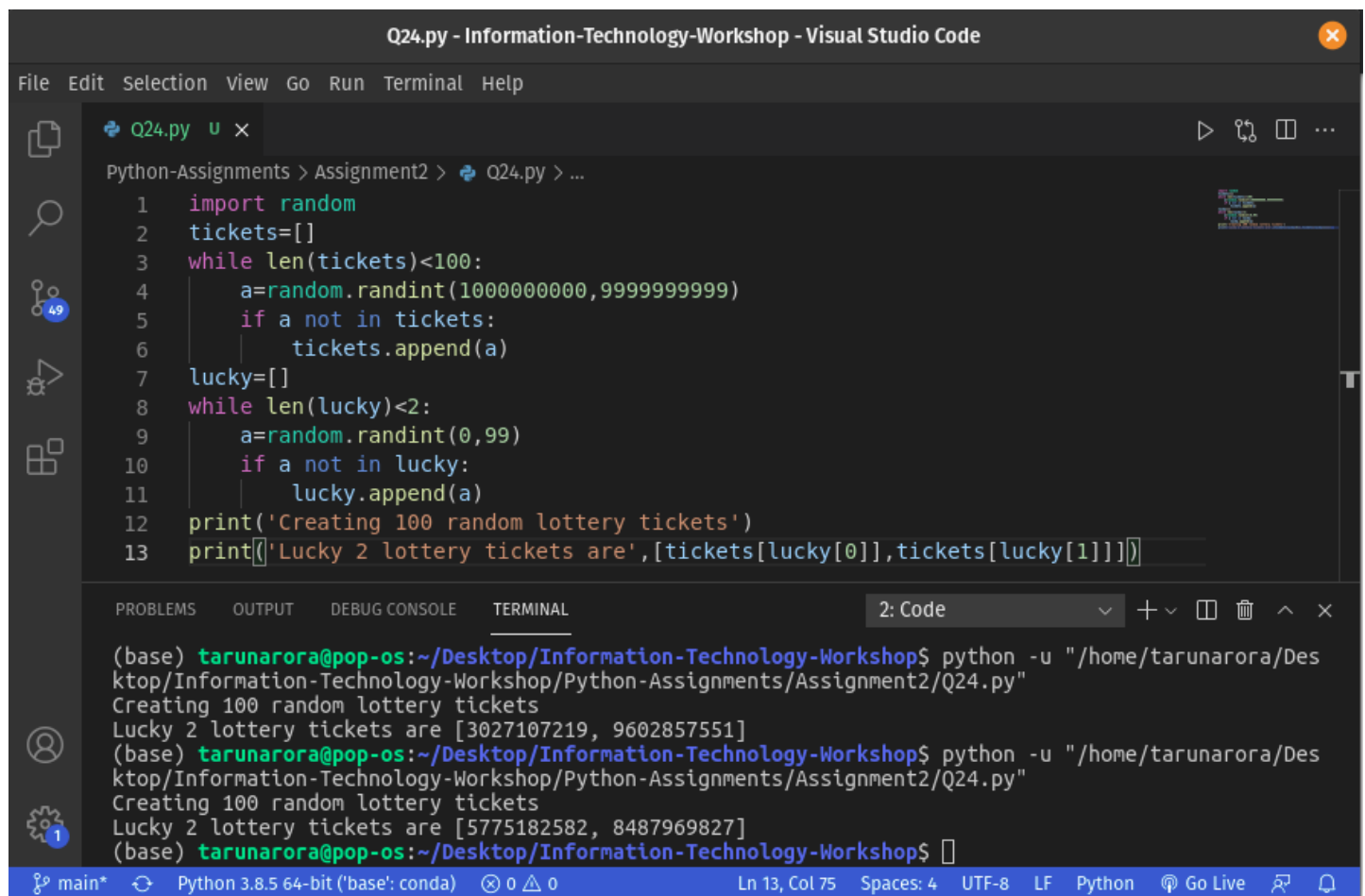
Note: You must adhere to the following conditions:

1. Lottery number must be 10 digits long.
2. All 100 ticket number must be unique.

Example: Creating 100 random lottery tickets

Lucky 2 lottery tickets are [7184805696, 7380986204]

Solution: -



The screenshot shows a Visual Studio Code window titled "Q24.py - Information-Technology-Workshop - Visual Studio Code". The editor displays a Python script in "Q24.py" with the following code:

```
1 import random
2 tickets=[]
3 while len(tickets)<100:
4     a=random.randint(1000000000,9999999999)
5     if a not in tickets:
6         tickets.append(a)
7 lucky=[]
8 while len(lucky)<2:
9     a=random.randint(0,99)
10    if a not in lucky:
11        lucky.append(a)
12 print('Creating 100 random lottery tickets')
13 print('Lucky 2 lottery tickets are',[tickets[lucky[0]],tickets[lucky[1]]])
```

The terminal at the bottom shows the execution of the script. It runs the command `python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q24.py"` twice, producing the following output:

```
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q24.py"
Creating 100 random lottery tickets
Lucky 2 lottery tickets are [3027107219, 9602857551]
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$ python -u "/home/tarunarora/Desktop/Information-Technology-Workshop/Python-Assignments/Assignment2/Q24.py"
Creating 100 random lottery tickets
Lucky 2 lottery tickets are [5775182582, 8487969827]
(base) tarunarora@pop-os:~/Desktop/Information-Technology-Workshop$
```

The status bar at the bottom indicates the file is "main*", the Python version is "Python 3.8.5 64-bit ('base': conda)", and the cursor is at "Ln 13, Col 75".

***** EOF *****