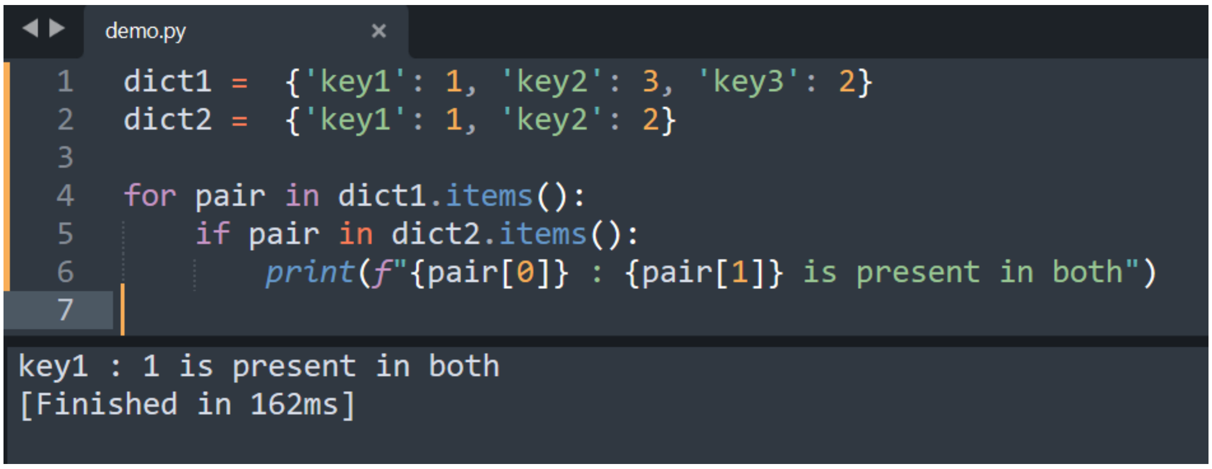
Python assignment 2

Name - Essamaraju Sasi kiran
Roll No - 20075031
Batch id - BA 1

---

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.

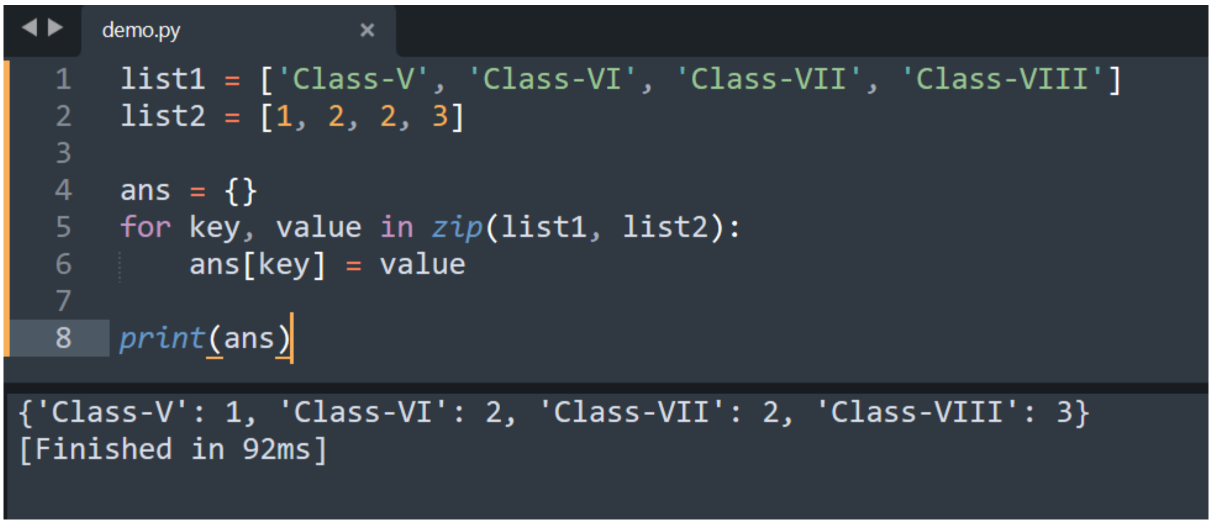


demo.py

```python
dict1 =  {'key1': 1, 'key2': 3, 'key3': 2}
dict2 =  {'key1': 1, 'key2': 2}

for pair in dict1.items():
    if pair in dict2.items():
        print(f"{pair[0]} : {pair[1]} is present in both")

```

```
key1 : 1 is present in both
[Finished in 162ms]
```

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-VII': {2}, 'Class-VI': {2}, 'Class-VIII': {3}, 'Class-V': {1}})

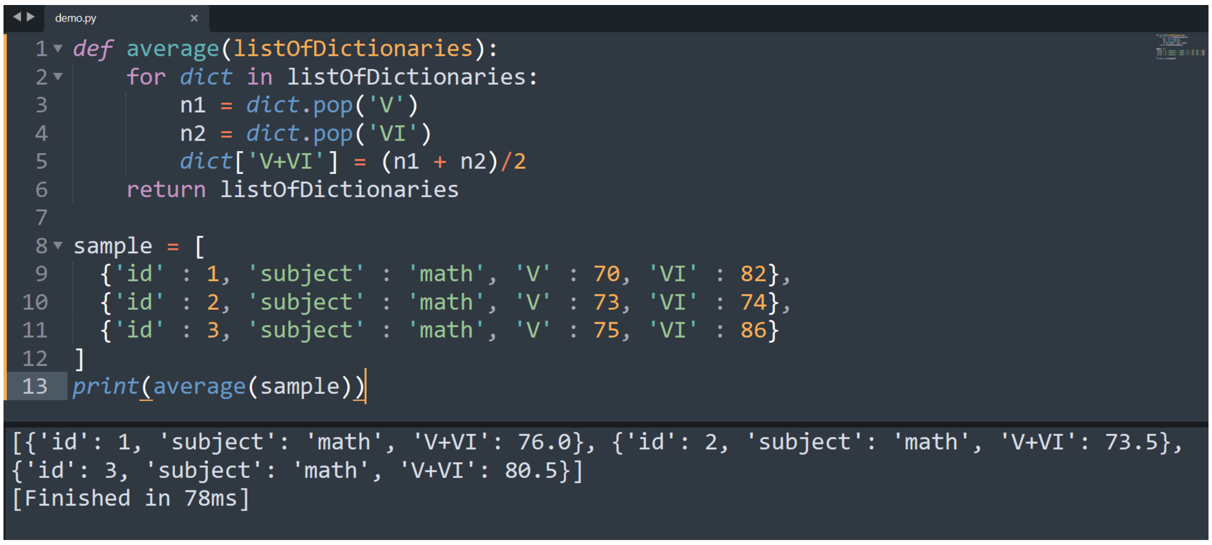


demo.py

```python
list1 = ['Class-V', 'Class-VI', 'Class-VII', 'Class-VIII']
list2 = [1, 2, 2, 3]

ans = {}
for key, value in zip(list1, list2):
    ans[key] = value

print(ans)
```

```
{'Class-V': 1, 'Class-VI': 2, 'Class-VII': 2, 'Class-VIII': 3}
[Finished in 92ms]
```

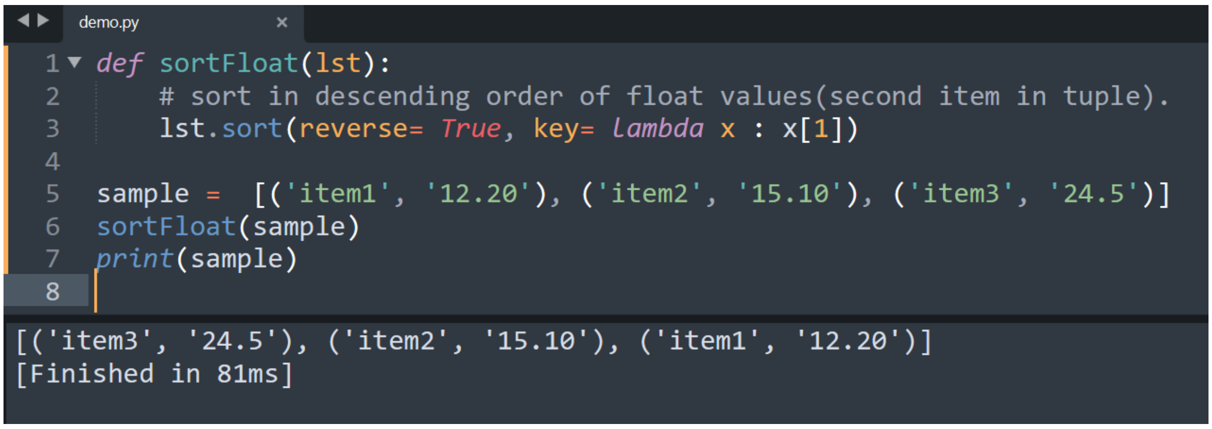
3. Write a Python program to replace dictionary values with their average.
Example: Input: [{'id' : 1, 'subject' : 'math', 'V' : 70, 'VI': 82},
{'id' : 2, 'subject' : 'math', 'V' : 73, 'VI' : 74},
{'id' : 3, 'subject' : 'math', 'V' : 75, 'VI' : 86}]
Output: [{'subject': 'math', 'id': 1, 'V+VI': 76.0},
{'subject': 'math', 'id': 2, 'V+VI': 73.5}
{'subject': 'math', 'id': 3, 'V+VI': 80.5}]



```python
def average(listOfDictionaries):
    for dict in listOfDictionaries:
        n1 = dict.pop('V')
        n2 = dict.pop('VI')
        dict['V+VI'] = (n1 + n2)/2
    return listOfDictionaries

sample = [
  {'id' : 1, 'subject' : 'math', 'V' : 70, 'VI' : 82},
  {'id' : 2, 'subject' : 'math', 'V' : 73, 'VI' : 74},
  {'id' : 3, 'subject' : 'math', 'V' : 75, 'VI' : 86}
]
print(average(sample))
```

```
[{'id': 1, 'subject': 'math', 'V+VI': 76.0}, {'id': 2, 'subject': 'math', 'V+VI': 73.5},
{'id': 3, 'subject': 'math', 'V+VI': 80.5}]
[Finished in 78ms]
```

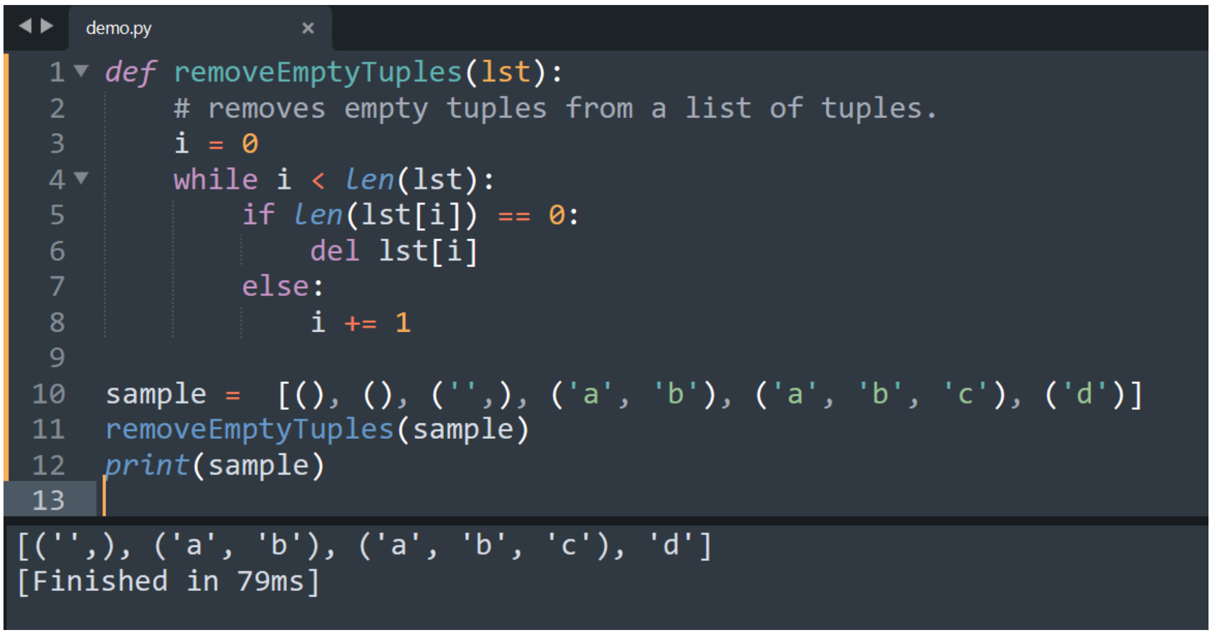
4. Write a Python program to sort a tuple by its float element.
Sample data: [('item1', '12.20'), ('item2', '15.10'), ('item3', '24.5')]
Expected Output: [('item3', '24.5'), ('item2', '15.10'), ('item1', '12.20')]



```python
def sortFloat(lst):
    # sort in descending order of float values(second item in tuple).
    lst.sort(reverse= True, key= lambda x : x[1])

sample =  [('item1', '12.20'), ('item2', '15.10'), ('item3', '24.5')]
sortFloat(sample)
print(sample)

```

```
[('item3', '24.5'), ('item2', '15.10'), ('item1', '12.20')]
[Finished in 81ms]
```

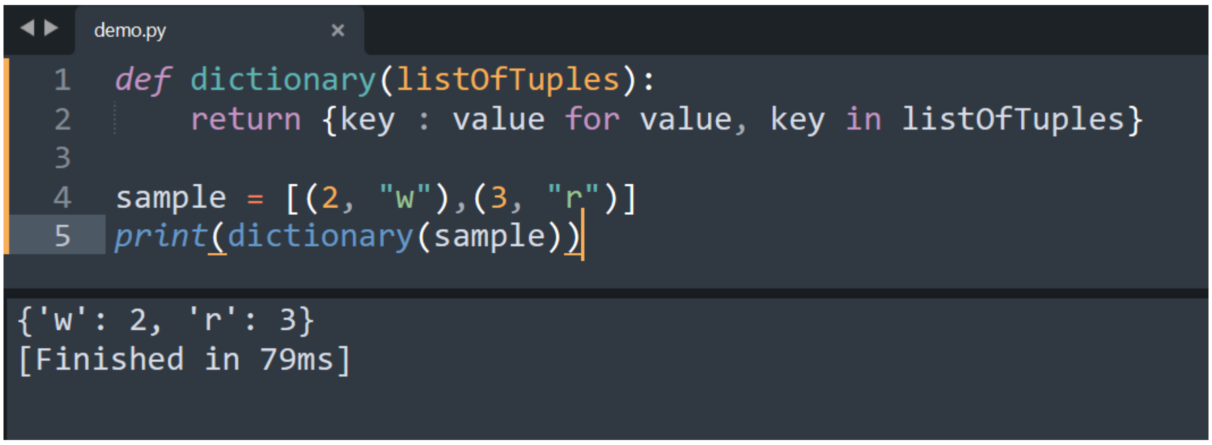
5. Write a Python program to remove an empty tuple(s) from a list of tuples.
Sample data: [(), (), ('',), ('a', 'b'), ('a', 'b', 'c'), ('d')]
Expected output: [('',), ('a', 'b'), ('a', 'b', 'c'), 'd']



```python
def removeEmptyTuples(lst):
    # removes empty tuples from a list of tuples.
    i = 0
    while i < len(lst):
        if len(lst[i]) == 0:
            del lst[i]
        else:
            i += 1

sample =  [(), (), ('',), ('a', 'b'), ('a', 'b', 'c'), ('d')]
removeEmptyTuples(sample)
print(sample)
```

```
[('',), ('a', 'b'), ('a', 'b', 'c'), 'd']
[Finished in 79ms]
```

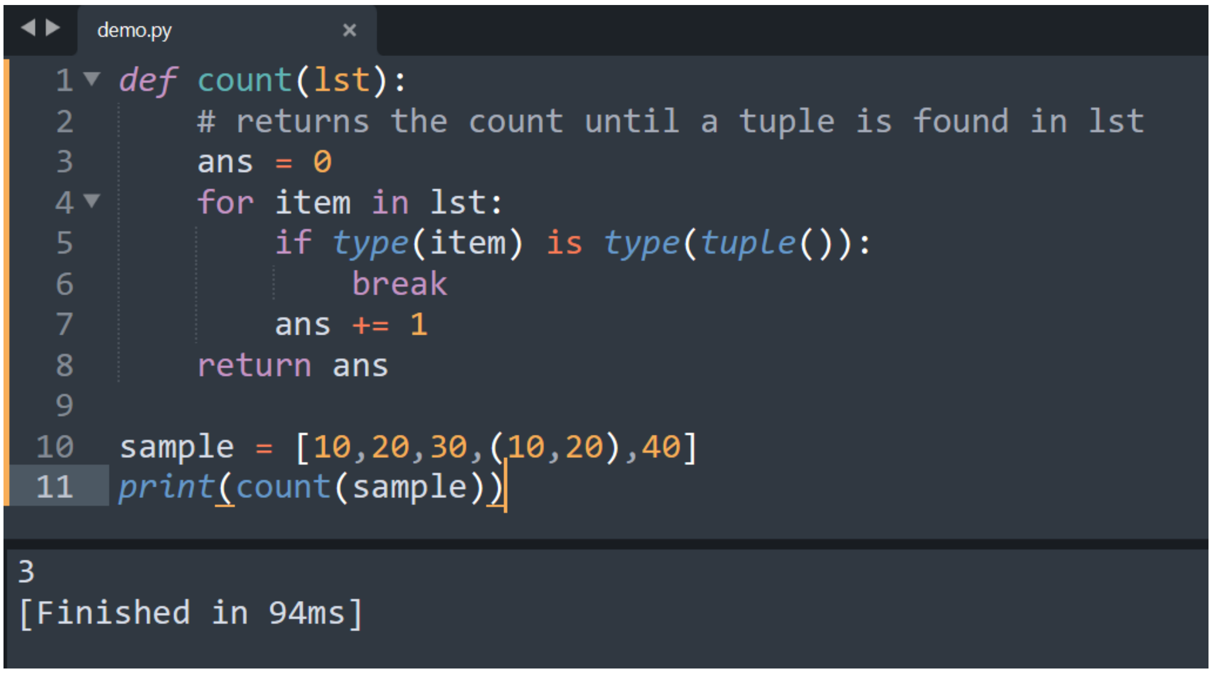
6. Write a Python program to convert a list of tuples into a dictionary.
Example: Input: ((2, "w"),(3, "r"))
Output: {'w': 2, 'r': 3}



```python
def dictionary(listOfTuples):
    return {key : value for value, key in listOfTuples}

sample = [(2, "w"),(3, "r")]
print(dictionary(sample))
```

```
{'w': 2, 'r': 3}
[Finished in 79ms]
```

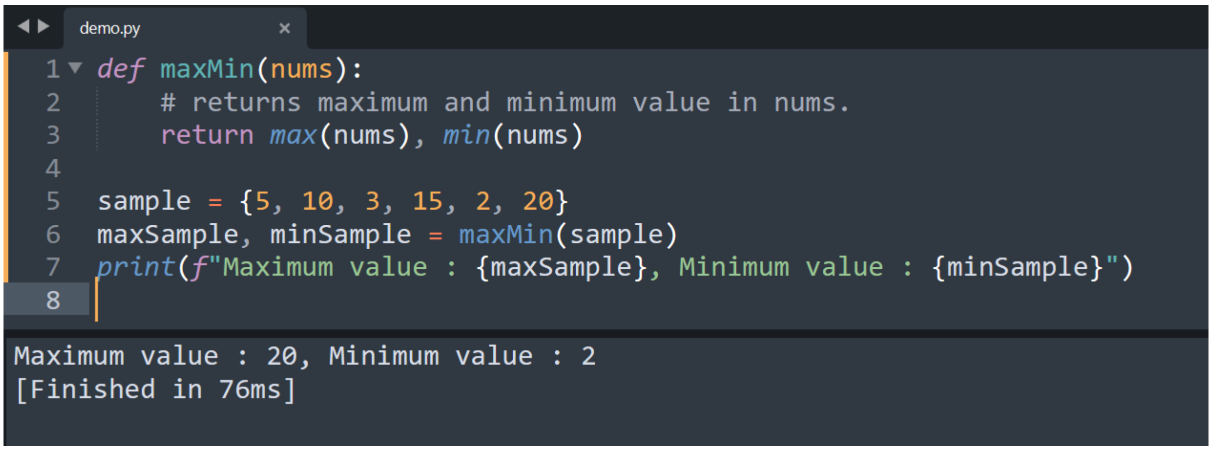
7. Write a Python program to count the elements in a list until an element is a tuple. Example: Input: [10,20,30,(10,20),40]
Output: 3



```python
def count(lst):
    # returns the count until a tuple is found in lst
    ans = 0
    for item in lst:
        if type(item) is type(tuple()):
            break
        ans += 1
    return ans

sample = [10,20,30,(10,20),40]
print(count(sample))
```

```
3
[Finished in 94ms]
```

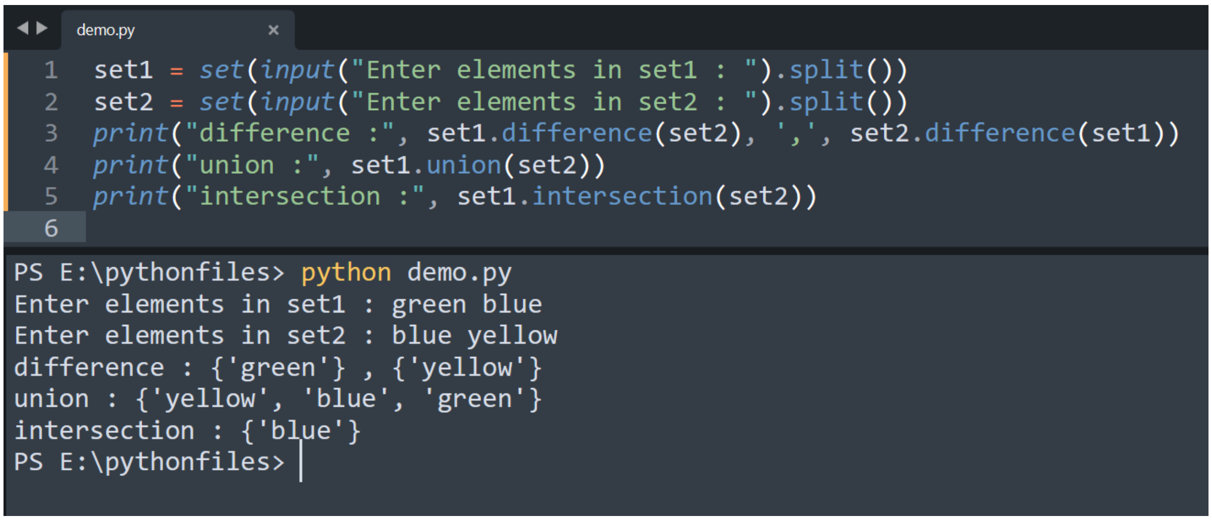
8. Write a Python program to find maximum and the minimum value in a set.
Example: Input: {5, 10, 3, 15, 2, 20}
Output: Maximum value: 20, Minimum value: 2



```python
def maxMin(nums):
    # returns maximum and minimum value in nums.
    return max(nums), min(nums)

sample = {5, 10, 3, 15, 2, 20}
maxSample, minSample = maxMin(sample)
print(f"Maximum value : {maxSample}, Minimum value : {minSample}")

```

```
Maximum value : 20, Minimum value : 2
[Finished in 76ms]
```

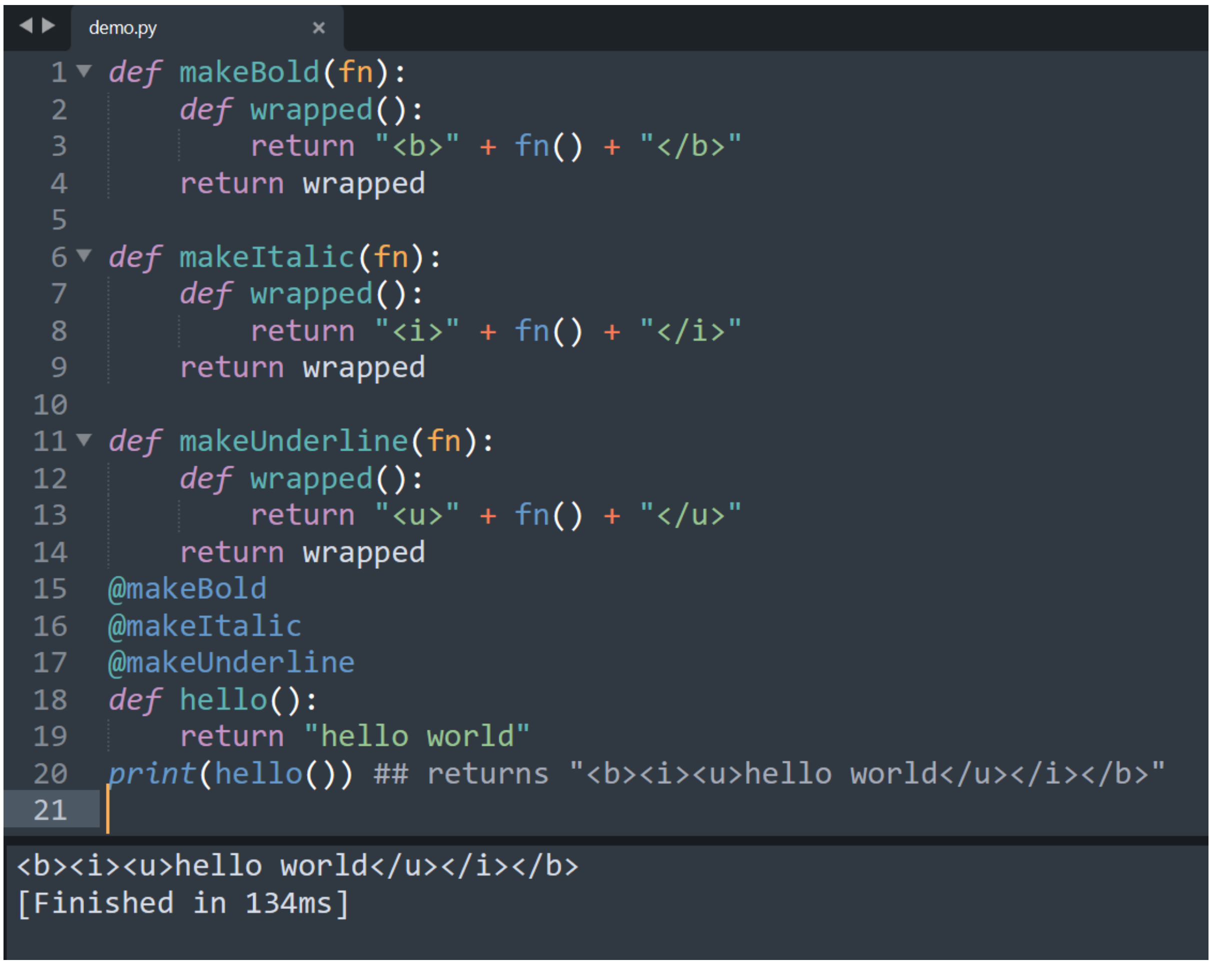
9. Write a Python program to create set difference, union, and intersection of sets. Example:
Input: set(["green", "blue"]), set(["blue", "yellow"])
Output: Difference: {'green'}, {'yellow'},
 Union: {'yellow', 'green', 'blue'}
 Intersection: {'blue'}



```python
set1 = set(input("Enter elements in set1 : ").split())
set2 = set(input("Enter elements in set2 : ").split())
print("difference :", set1.difference(set2), ',', set2.difference(set1))
print("union :", set1.union(set2))
print("intersection :", set1.intersection(set2))
```

```
PS E:\pythonfiles> python demo.py
Enter elements in set1 : green blue
Enter elements in set2 : blue yellow
difference : {'green'} , {'yellow'}
union : {'yellow', 'blue', 'green'}
intersection : {'blue'}
PS E:\pythonfiles>
```

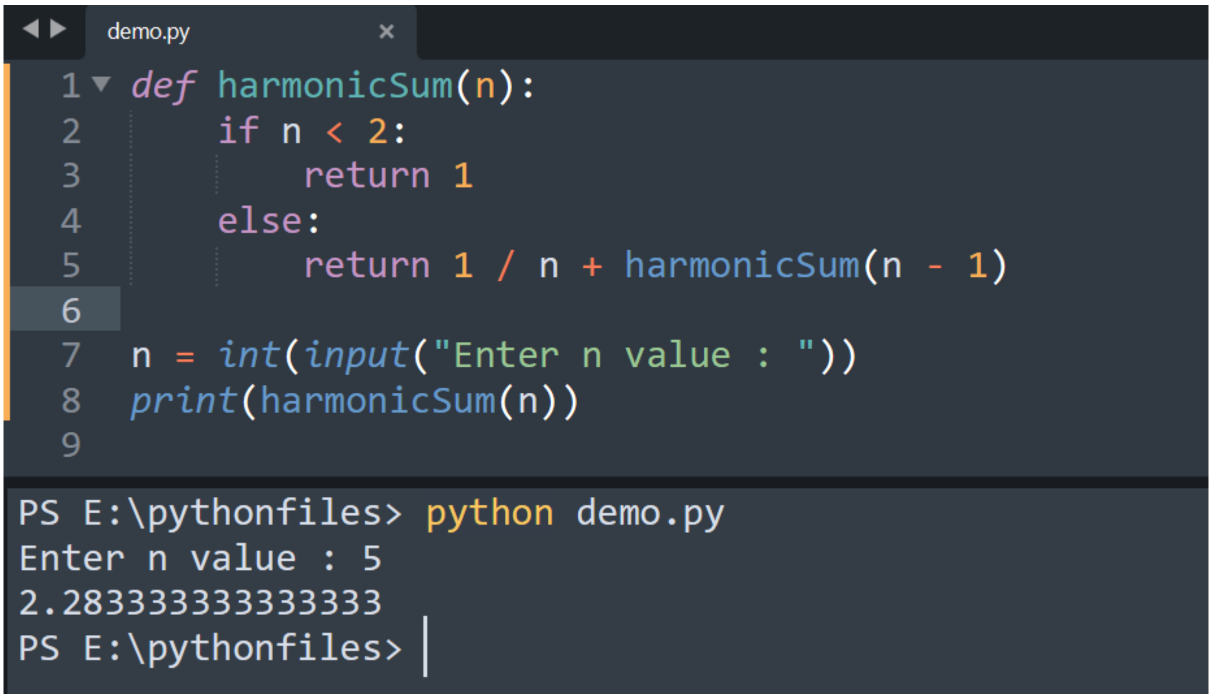
10. Write a Python program to make a chain of function decorators (bold, italic, underline etc.). Example: Input: hello world Output: *hello world*



```python
def makeBold(fn):
    def wrapped():
        return "<b>" + fn() + "</b>"
    return wrapped

def makeItalic(fn):
    def wrapped():
        return "<i>" + fn() + "</i>"
    return wrapped

def makeUnderline(fn):
    def wrapped():
        return "<u>" + fn() + "</u>"
    return wrapped
@makeBold
@makeItalic
@makeUnderline
def hello():
    return "hello world"
print(hello()) ## returns "<b><i><u>hello world</u></i></b>"

```

```
<b><i><u>hello world</u></i></b>
[Finished in 134ms]
```

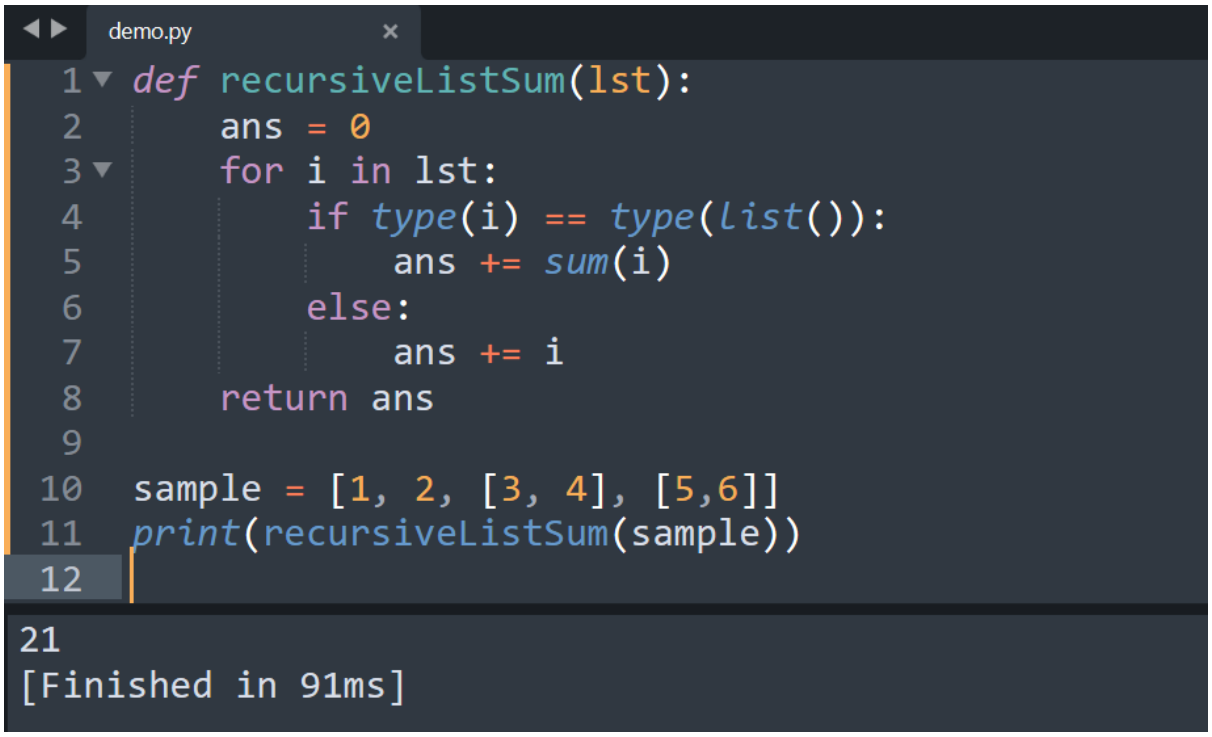
11. Write a Python program to calculate the harmonic sum of n-1.



```python
def harmonicSum(n):
    if n < 2:
        return 1
    else:
        return 1 / n + harmonicSum(n - 1)

n = int(input("Enter n value : "))
print(harmonicSum(n))
```

```
PS E:\pythonfiles> python demo.py
Enter n value : 5
2.283333333333333
PS E:\pythonfiles>
```

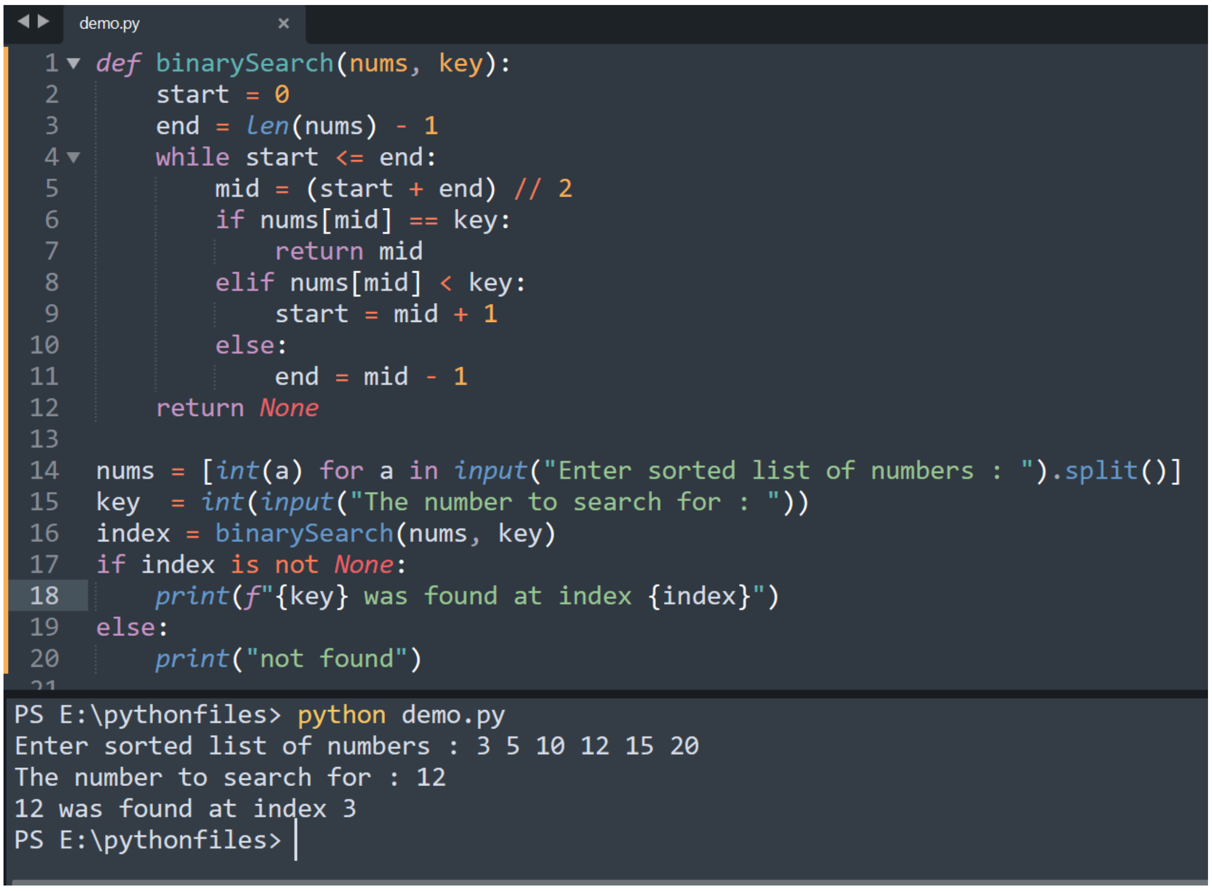
12. Write a Python program of recursion list sum.
Test Data: [1, 2, [3,4], [5,6]]
Expected Result: 21



```python
def recursiveListSum(lst):
    ans = 0
    for i in lst:
        if type(i) == type(list()):
            ans += sum(i)
        else:
            ans += i
    return ans

sample = [1, 2, [3, 4], [5,6]]
print(recursiveListSum(sample))
```

```
21
[Finished in 91ms]
```

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.

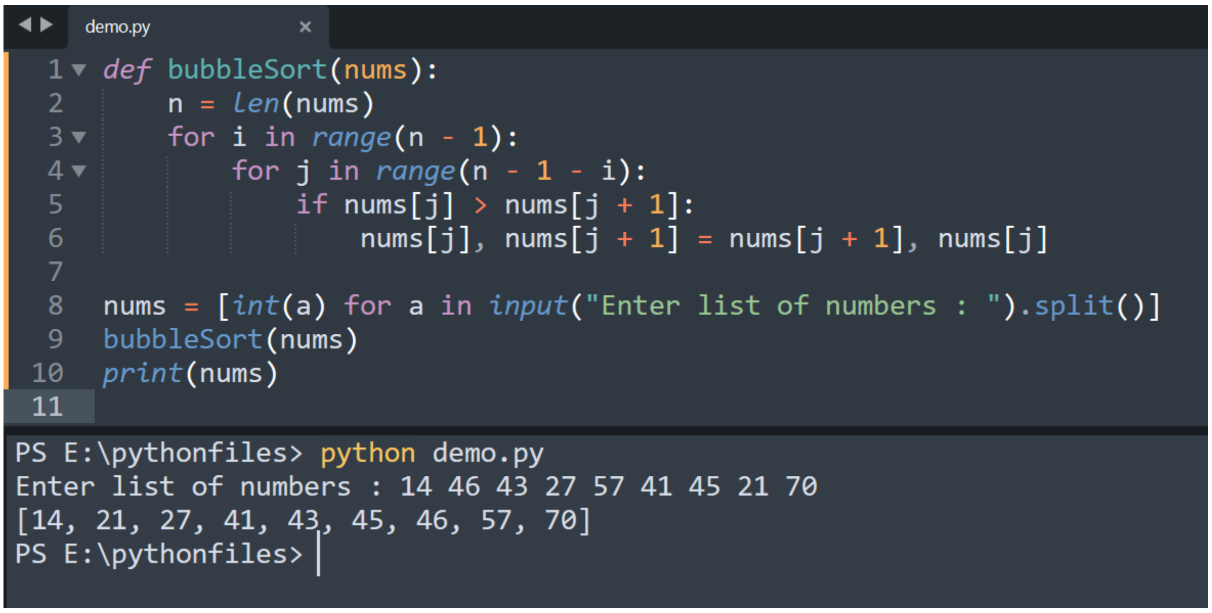


demo.py

```python
def binarySearch(nums, key):
    start = 0
    end = len(nums) - 1
    while start <= end:
        mid = (start + end) // 2
        if nums[mid] == key:
            return mid
        elif nums[mid] < key:
            start = mid + 1
        else:
            end = mid - 1
    return None

nums = [int(a) for a in input("Enter sorted list of numbers : ").split()]
key  = int(input("The number to search for : "))
index = binarySearch(nums, key)
if index is not None:
    print(f"{key} was found at index {index}")
else:
    print("not found")
```

```
PS E:\pythonfiles> python demo.py
Enter sorted list of numbers : 3 5 10 12 15 20
The number to search for : 12
12 was found at index 3
PS E:\pythonfiles>
```

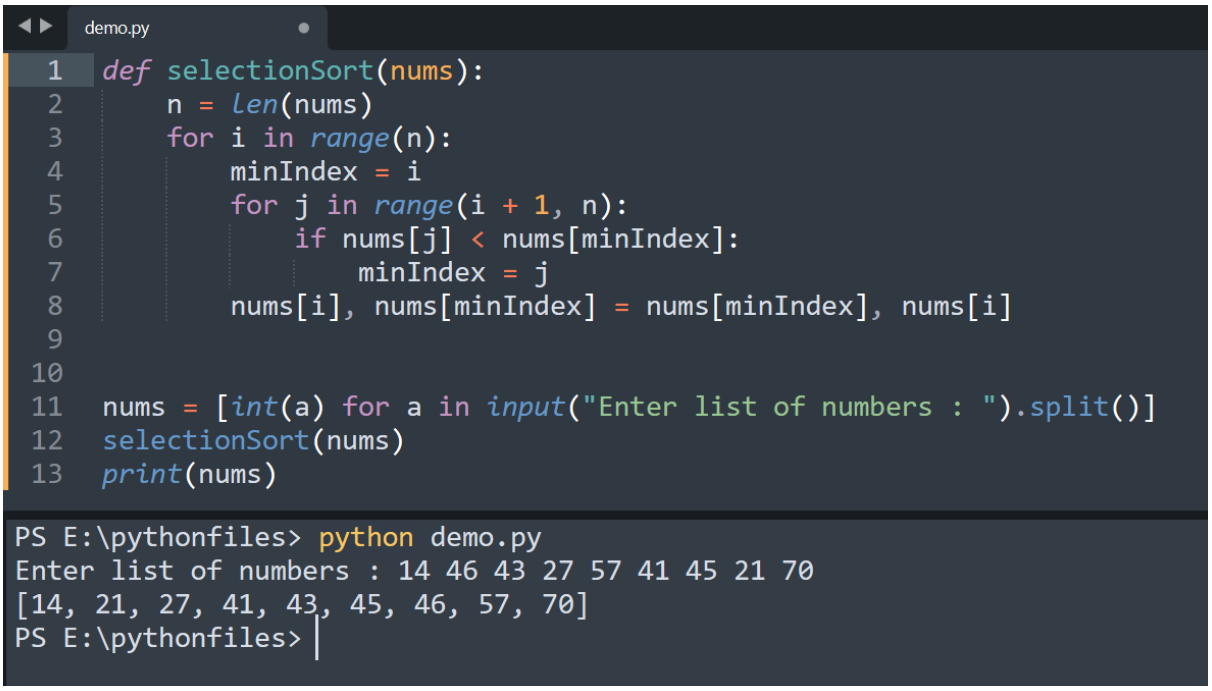
14. Write a Python program to sort a list of elements using the bubble sort algorithm.
Example: Sample Data: [14, 46, 43, 27, 57, 41, 45, 21, 70]
Expected Result: [14, 21, 27, 41, 43, 45, 46, 57, 70]


demo.py

```python
def bubbleSort(nums):
    n = len(nums)
    for i in range(n - 1):
        for j in range(n - 1 - i):
            if nums[j] > nums[j + 1]:
                nums[j], nums[j + 1] = nums[j + 1], nums[j]

nums = [int(a) for a in input("Enter list of numbers : ").split()]
bubbleSort(nums)
print(nums)

```

```
PS E:\pythonfiles> python demo.py
Enter list of numbers : 14 46 43 27 57 41 45 21 70
[14, 21, 27, 41, 43, 45, 46, 57, 70]
PS E:\pythonfiles>
```

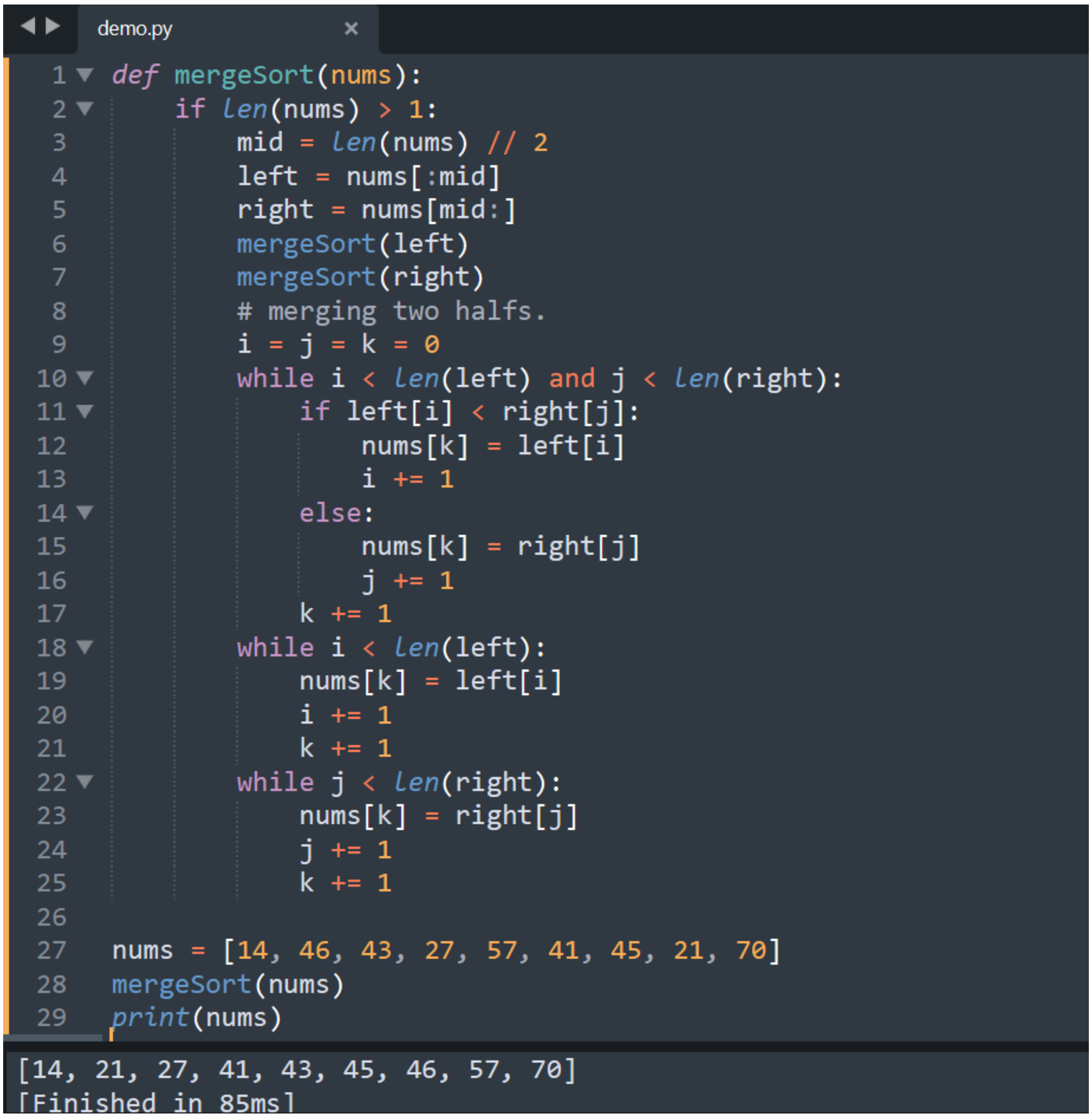
15. Write a Python program to sort a list of elements using the selection sort algorithm.
Example: Sample Data: [14, 46, 43, 27, 57, 41, 45, 21, 70]
Expected Result: [14, 21, 27, 41, 43, 45, 46, 57, 70]


demo.py

```python
def selectionSort(nums):
    n = len(nums)
    for i in range(n):
        minIndex = i
        for j in range(i + 1, n):
            if nums[j] < nums[minIndex]:
                minIndex = j
        nums[i], nums[minIndex] = nums[minIndex], nums[i]


nums = [int(a) for a in input("Enter list of numbers : ").split()]
selectionSort(nums)
print(nums)
```

```
PS E:\pythonfiles> python demo.py
Enter list of numbers : 14 46 43 27 57 41 45 21 70
[14, 21, 27, 41, 43, 45, 46, 57, 70]
PS E:\pythonfiles>
```

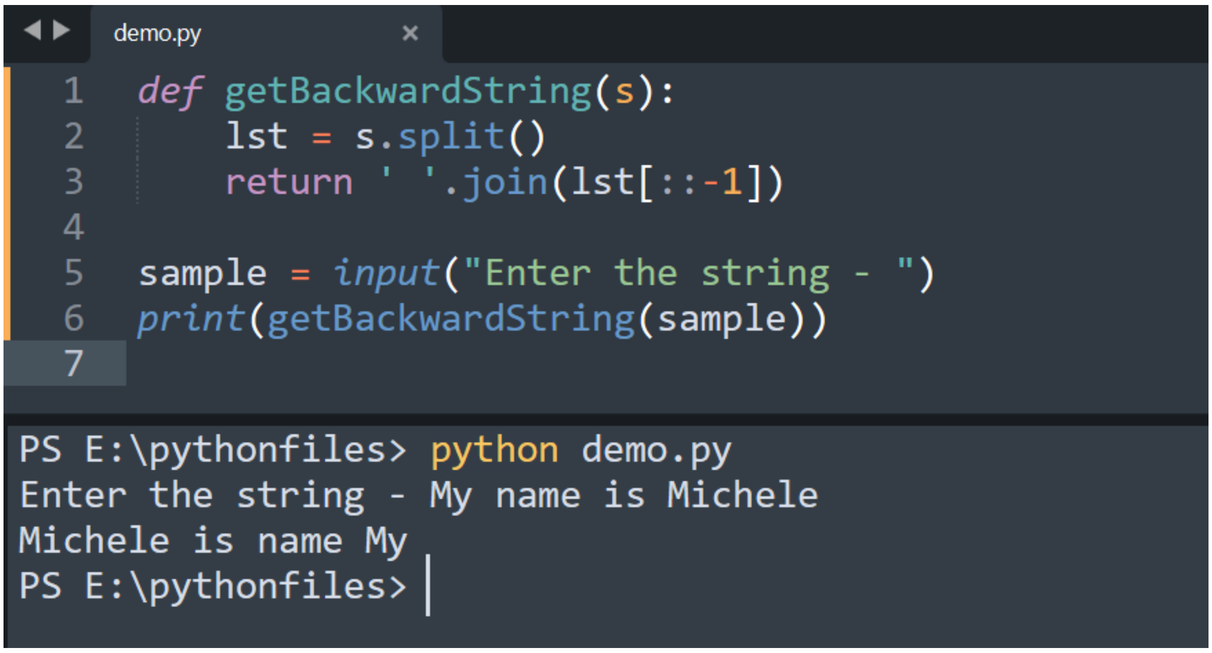
16. Write a Python program to sort a list of elements using the merge sort algorithm.
Example: Split Sample Data: [14, 46, 43, 27, 57, 41, 45, 21, 70]
Merge and Sort(Expected Result): [14, 21, 27, 41, 43, 45, 46, 57, 70]



```python
def mergeSort(nums):
    if len(nums) > 1:
        mid = len(nums) // 2
        left = nums[:mid]
        right = nums[mid:]
        mergeSort(left)
        mergeSort(right)
        # merging two halfs.
        i = j = k = 0
        while i < len(left) and j < len(right):
            if left[i] < right[j]:
                nums[k] = left[i]
                i += 1
            else:
                nums[k] = right[j]
                j += 1
            k += 1
        while i < len(left):
            nums[k] = left[i]
            i += 1
            k += 1
        while j < len(right):
            nums[k] = right[j]
            j += 1
            k += 1

nums = [14, 46, 43, 27, 57, 41, 45, 21, 70]
mergeSort(nums)
print(nums)
```

```
[14, 21, 27, 41, 43, 45, 46, 57, 70]
[Finished in 85ms]
```

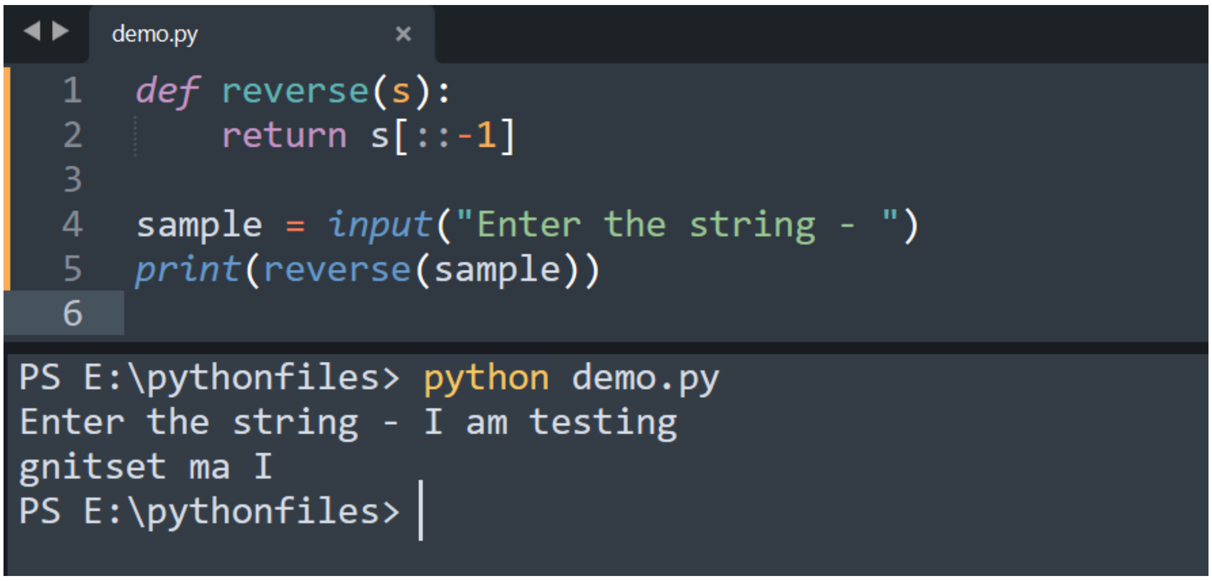
17. Write a Python program using functions that asks the user for a long string containing multiple words. Print back to the user the same string, except with the words in backwards order. For example, say I type the string: My name is Michele; Then I would see the string: Michele is name My; shown back to me.



```python
def getBackwardString(s):
    lst = s.split()
    return ' '.join(lst[::-1])

sample = input("Enter the string - ")
print(getBackwardString(sample))
```

```
PS E:\pythonfiles> python demo.py
Enter the string - My name is Michele
Michele is name My
PS E:\pythonfiles>
```

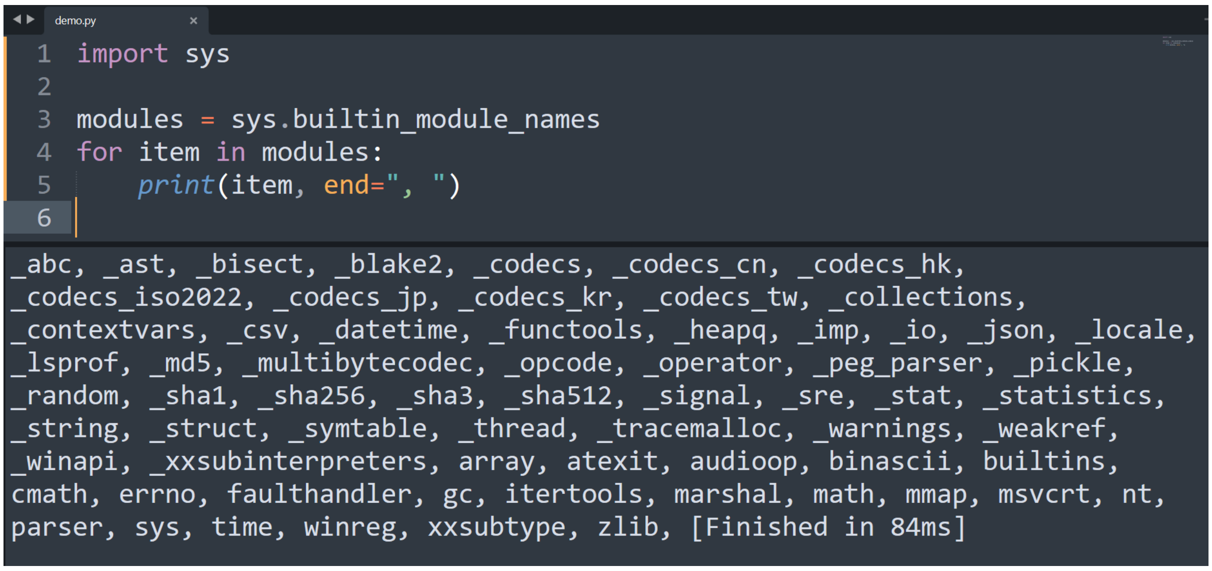
18. Define a function reverse() that computes the reversal of a string.
For example, reverse(“I am testing”) should return the string ”gnitset ma I”.



```python
def reverse(s):
    return s[::-1]

sample = input("Enter the string - ")
print(reverse(sample))
```

```
PS E:\pythonfiles> python demo.py
Enter the string - I am testing
gnitset ma I
PS E:\pythonfiles>
```

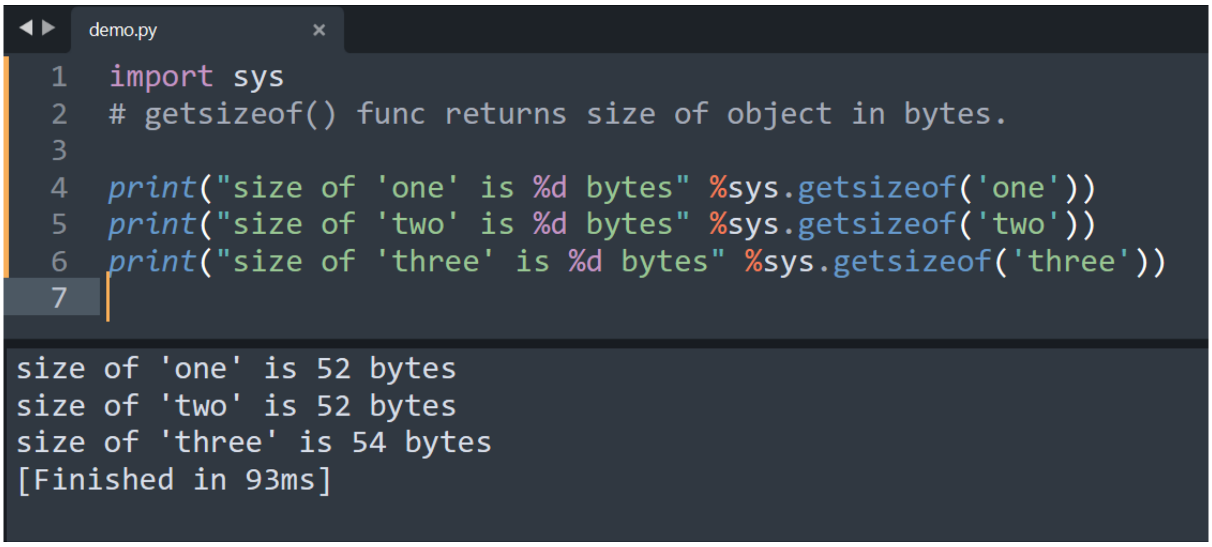
19. Write a Python program to find the available built-in modules.
Example: math, random, uuid, sys, syslog etc.



```python
import sys

modules = sys.builtin_module_names
for item in modules:
    print(item, end=", ")

```

```
_abc, _ast, _bisect, _blake2, _codecs, _codecs_cn, _codecs_hk,
_codecs_iso2022, _codecs_jp, _codecs_kr, _codecs_tw, _collections,
_contextvars, _csv, _datetime, _functools, _heapq, _imp, _io, _json, _locale,
_lsprof, _md5, _multibytecodec, _opcode, _operator, _peg_parser, _pickle,
_random, _sha1, _sha256, _sha3, _sha512, _signal, _sre, _stat, _statistics,
_string, _struct, _symtable, _thread, _tracemalloc, _warnings, _weakref,
_winapi, _xxsubinterpreters, array, atexit, audioop, binascii, builtins,
cmath, errno, faulthandler, gc, itertools, marshal, math, mmap, msvcrt, nt,
parser, sys, time, winreg, xxsubtype, zlib, [Finished in 84ms]
```

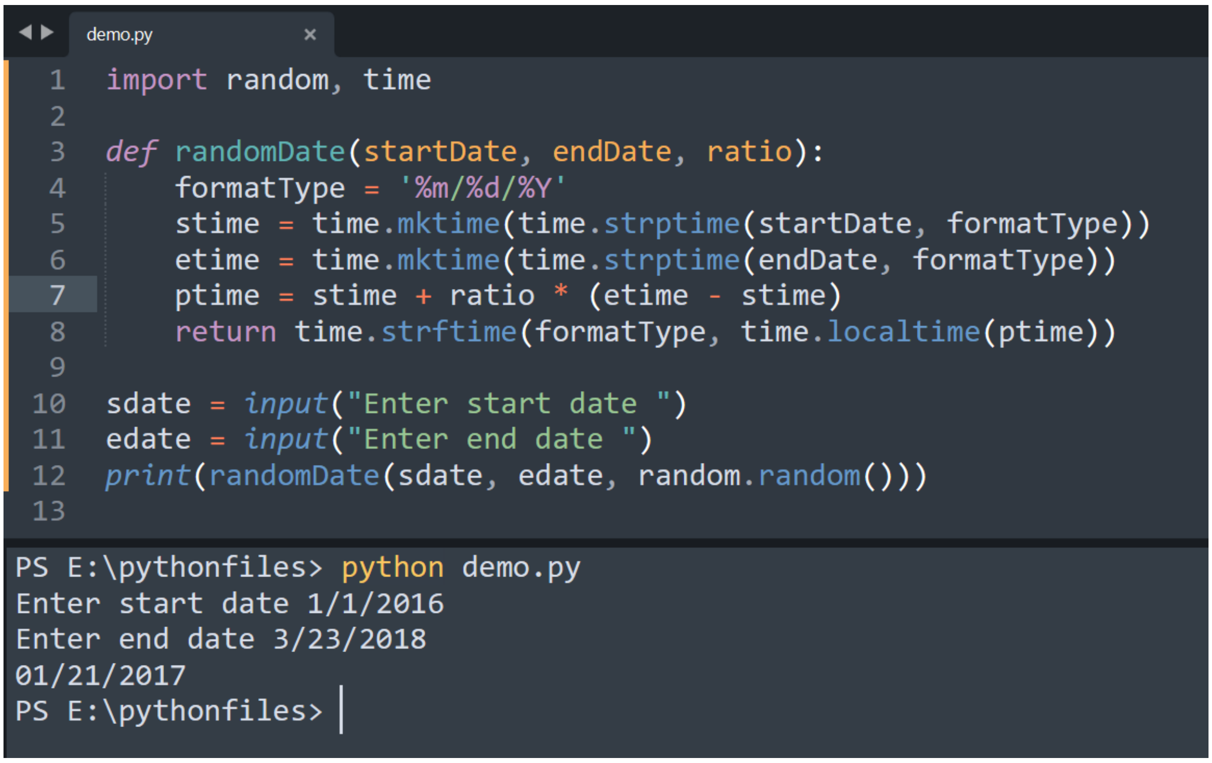
20. Write a Python program to get the size of an object in bytes by using module “sys”.
Example: Memory size of 'one' = 52 bytes
Memory size of 'four' = 53 bytes
Memory size of 'three' = 54 bytes



```python
import sys
# getsizeof() func returns size of object in bytes.

print("size of 'one' is %d bytes" %sys.getsizeof('one'))
print("size of 'two' is %d bytes" %sys.getsizeof('two'))
print("size of 'three' is %d bytes" %sys.getsizeof('three'))

```

```
size of 'one' is 52 bytes
size of 'two' is 52 bytes
size of 'three' is 54 bytes
[Finished in 93ms]
```

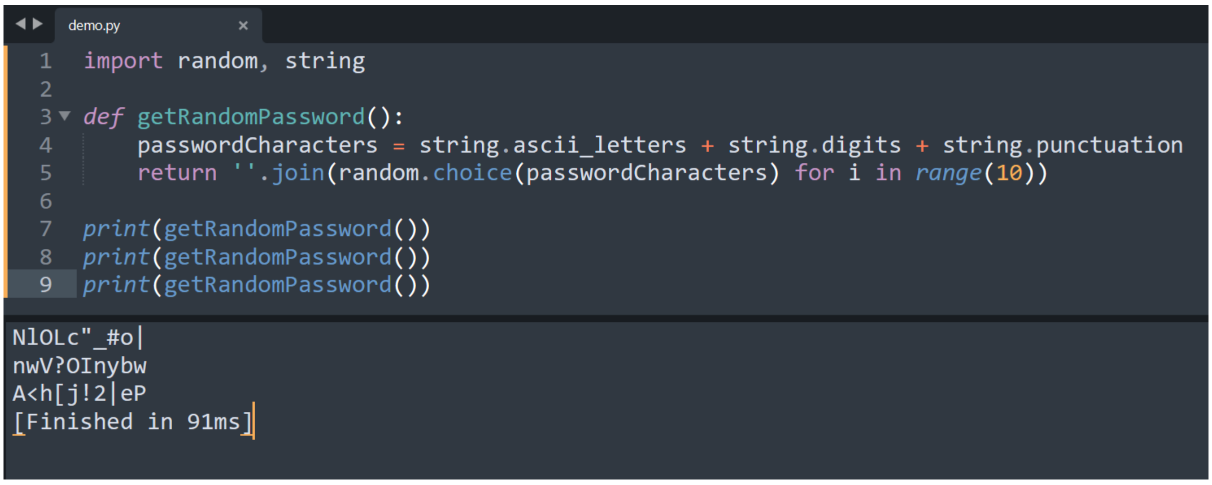
21. Using the module random and time in python generate a random date between given start and end dates.
Example: Printing random date between 1/1/2016 and 3/23/2018
Random Date = 02/25/2016



```python
import random, time

def randomDate(startDate, endDate, ratio):
    formatType = '%m/%d/%Y'
    stime = time.mktime(time.strptime(startDate, formatType))
    etime = time.mktime(time.strptime(endDate, formatType))
    ptime = stime + ratio * (etime - stime)
    return time.strftime(formatType, time.localtime(ptime))

sdate = input("Enter start date ")
edate = input("Enter end date ")
print(randomDate(sdate, edate, random.random()))

```

```
PS E:\pythonfiles> python demo.py
Enter start date 1/1/2016
Enter end date 3/23/2018
01/21/2017
PS E:\pythonfiles>
```

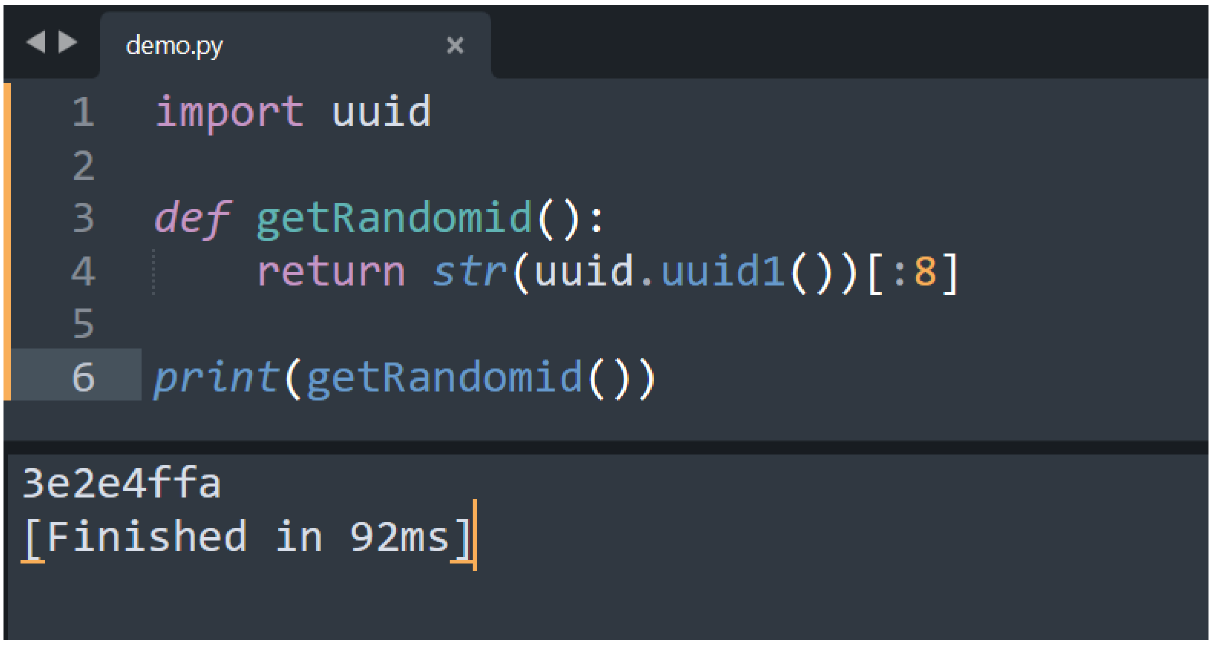
22. Generate three random password string of length 10 with special characters, letters, and digits by using python modules (random and string).
Example: First Random String: yrjmcyi^VS
Second Random String: |}Hd]!^>~I
Third Random String: 3^a93@x=|Z



```python
import random, string

def getRandomPassword():
    passwordCharacters = string.ascii_letters + string.digits + string.punctuation
    return ''.join(random.choice(passwordCharacters) for i in range(10))

print(getRandomPassword())
print(getRandomPassword())
print(getRandomPassword())
```

```
NlOLc"_#o|
nwV?OInybw
A<h[j!2|eP
[Finished in 91ms]
```

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

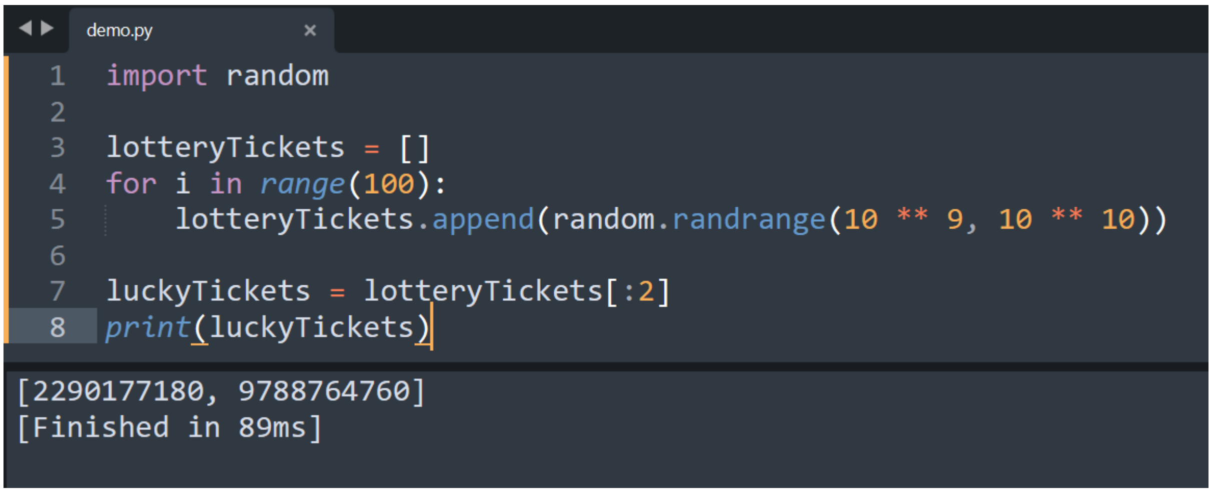


demo.py

```python
import uuid

def getRandomid():
    return str(uuid.uuid1())[:8]

print(getRandomid())
```

```
3e2e4ffa
[Finished in 92ms]
```

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]



demo.py

```python
import random

lotteryTickets = []
for i in range(100):
    lotteryTickets.append(random.randrange(10 ** 9, 10 ** 10))

luckyTickets = lotteryTickets[:2]
print(luckyTickets)
```

```
[2290177180, 9788764760]
[Finished in 89ms]
```
