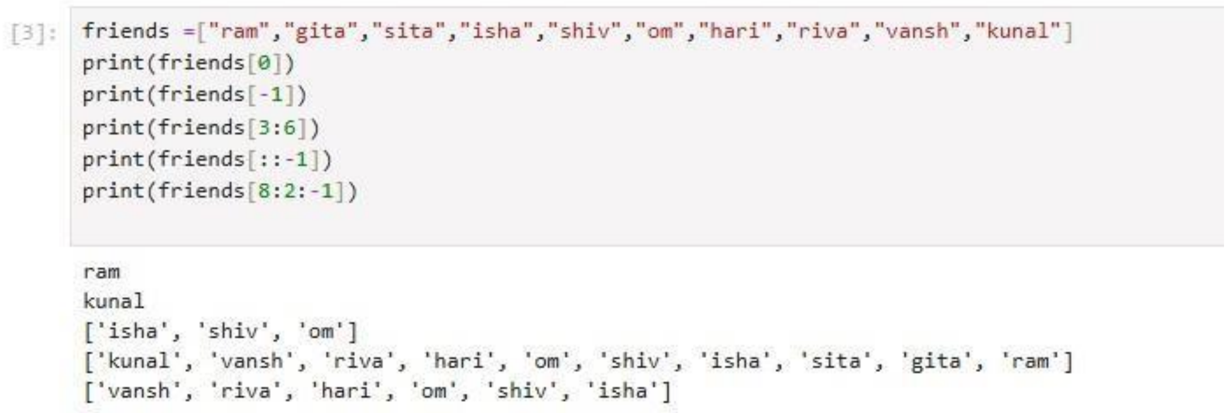
Kanishk Kaushik

23104015

B14

Practice assignment 2b

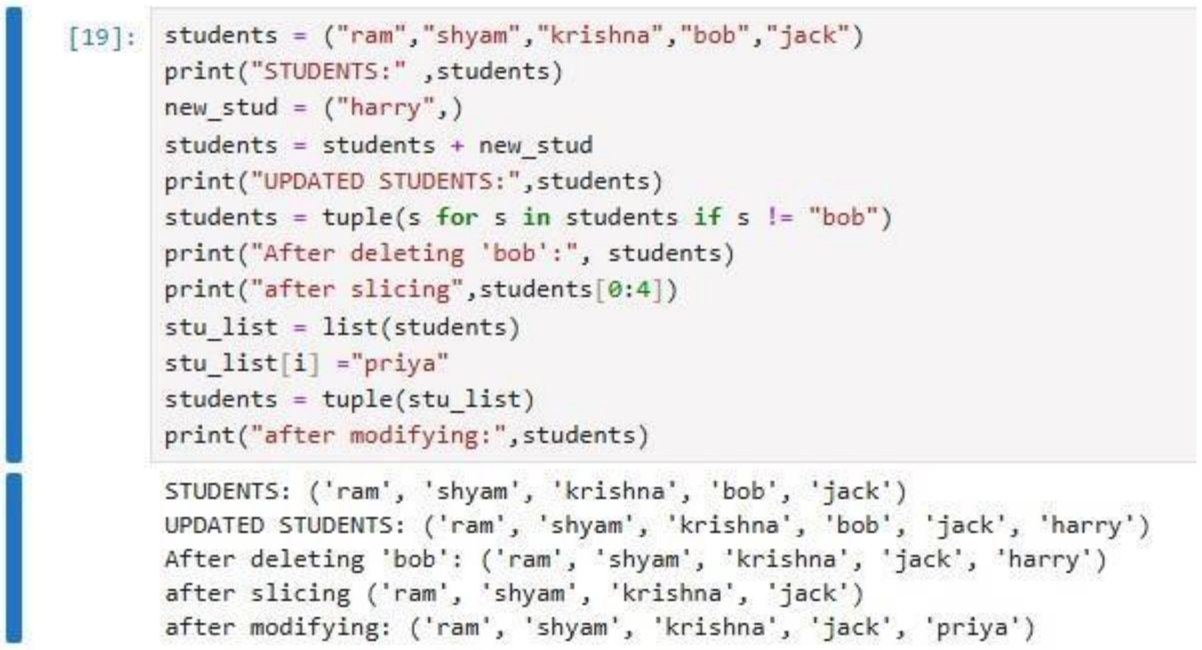
1. Write a python program to initialize a list with 10 names of your friends. Do the following :-
a. Print the name at the first index.
b. Print the name at the last index.
c. Use slicing to print the names from the third to the fifth index.
d. Use slicing to print the names in reverse order.
e. Use slicing to print the names from eight to third.



```
[3]: friends =["ram","gita","sita","isha","shiv","om","hari","riva","vansh","kunal"]
     print(friends[0])
     print(friends[-1])
     print(friends[3:6])
     print(friends[::-1])
     print(friends[8:2:-1])
```

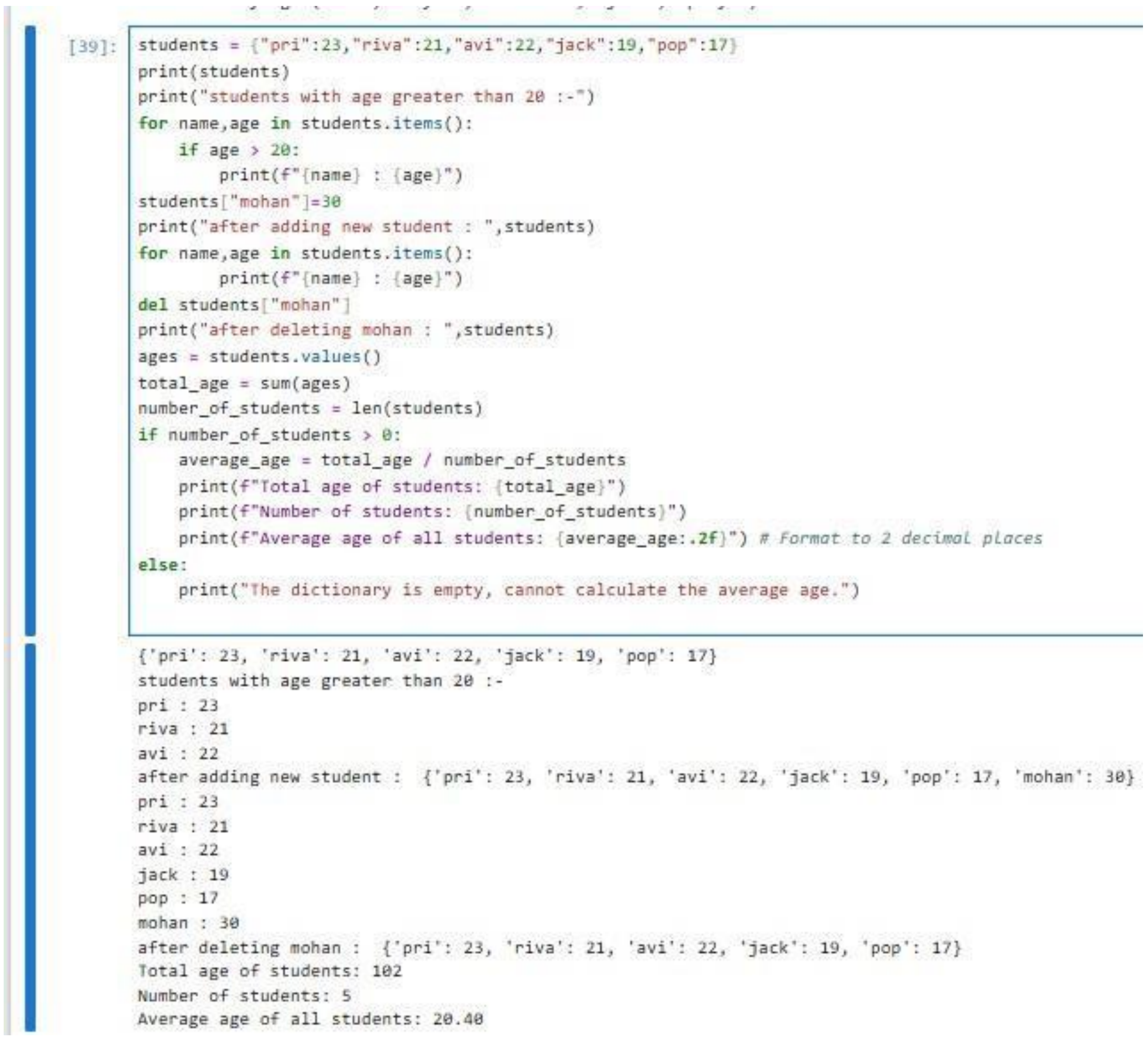
```
ram
kunal
['isha', 'shiv', 'om']
['kunal', 'vansh', 'riva', 'hari', 'om', 'shiv', 'isha', 'sita', 'gita', 'ram']
['vansh', 'riva', 'hari', 'om', 'shiv', 'isha']
```

2. Create a tuple of 5 students. Do the following :-
a. Display all the students.
b. Add a new student.
c. Delete a student.
d. Use slicing to print students from the first index to third index.
e. Modify the second index value. Can you modify it?


```
[19]: students = ("ram","shyam","krishna","bob","jack")
      print("STUDENTS:" ,students)
      new_stud = ("harry",)
      students = students + new_stud
      print("UPDATED STUDENTS:",students)
      students = tuple(s for s in students if s != "bob")
      print("After deleting 'bob':", students)
      print("after slicing",students[0:4])
      stu_list = list(students)
      stu_list[i] ="priya"
      students = tuple(stu_list)
      print("after modifying:",students)
```

```
STUDENTS: ('ram', 'shyam', 'krishna', 'bob', 'jack')
UPDATED STUDENTS: ('ram', 'shyam', 'krishna', 'bob', 'jack', 'harry')
After deleting 'bob': ('ram', 'shyam', 'krishna', 'jack', 'harry')
after slicing ('ram', 'shyam', 'krishna', 'jack')
after modifying: ('ram', 'shyam', 'krishna', 'jack', 'priya')
```

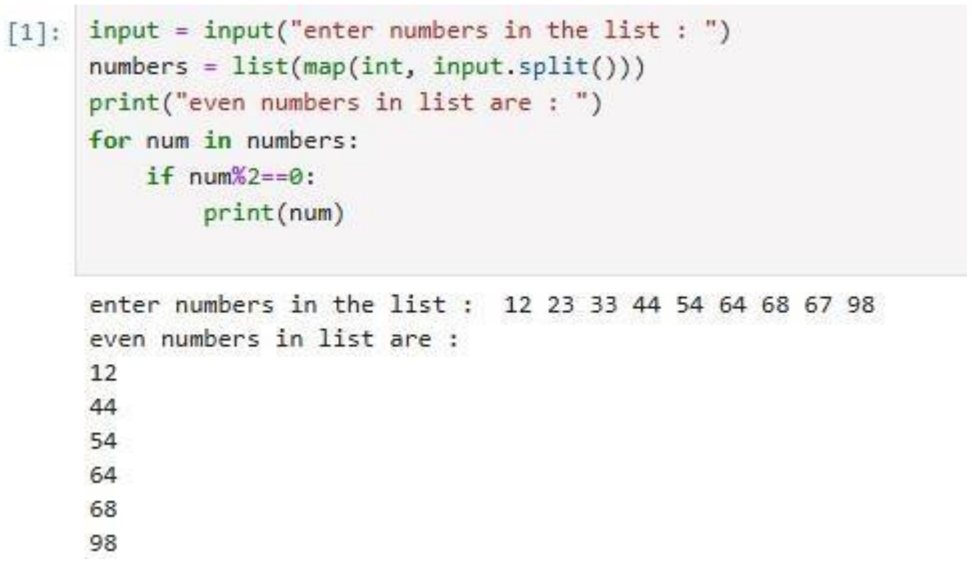
3. Create a dictionary with key value as name and age. Do the following:-
a. Display those students whose age is greater than 20.
b. Add one more student with age 30.
c. Display all the students using .items().
d. Delete a student.
e. Display the average age of all the students.



```
[39]: students = {"pri":23,"riva":21,"avi":22,"jack":19,"pop":17}
      print(students)
      print("students with age greater than 20 :-")
      for name,age in students.items():
          if age > 20:
              print(f"{name} : {age}")
      students["mohan"]=30
      print("after adding new student : ",students)
      for name,age in students.items():
              print(f"{name} : {age}")
      del students["mohan"]
      print("after deleting mohan : ",students)
      ages = students.values()
      total_age = sum(ages)
      number_of_students = len(students)
      if number_of_students > 0:
          average_age = total_age / number_of_students
          print(f"Total age of students: {total_age}")
          print(f"Number of students: {number_of_students}")
          print(f"Average age of all students: {average_age:.2f}") # Format to 2 decimal places
      else:
          print("The dictionary is empty, cannot calculate the average age.")
```

```
{'pri': 23, 'riva': 21, 'avi': 22, 'jack': 19, 'pop': 17}
students with age greater than 20 :-
pri : 23
riva : 21
avi : 22
after adding new student :  {'pri': 23, 'riva': 21, 'avi': 22, 'jack': 19, 'pop': 17, 'mohan': 30}
pri : 23
riva : 21
avi : 22
jack : 19
pop : 17
mohan : 30
after deleting mohan :  {'pri': 23, 'riva': 21, 'avi': 22, 'jack': 19, 'pop': 17}
Total age of students: 102
Number of students: 5
Average age of all students: 20.40
```

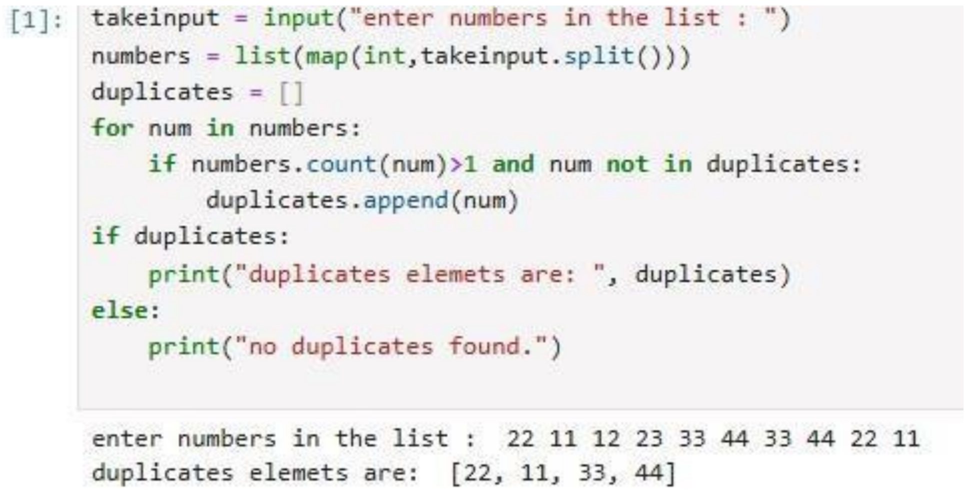
4. Write a python program that takes in a list with numbers in it and prints out all the even numbers.



```
[1]: input = input("enter numbers in the list : ")
     numbers = list(map(int, input.split()))
     print("even numbers in list are : ")
     for num in numbers:
         if num%2==0:
             print(num)
```

```
enter numbers in the list :  12 23 33 44 54 64 68 67 98
even numbers in list are :
12
44
54
64
68
98
```

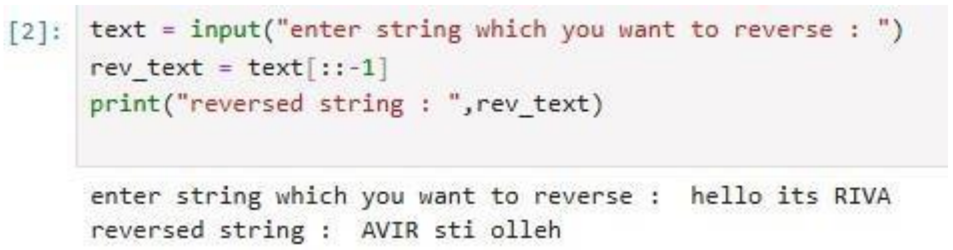
5. Write a python program to print all the duplicate elements in the list.



```
[1]: takeinput = input("enter numbers in the list : ")
     numbers = list(map(int,takeinput.split()))
     duplicates = []
     for num in numbers:
         if numbers.count(num)>1 and num not in duplicates:
             duplicates.append(num)
     if duplicates:
         print("duplicates elemets are: ", duplicates)
     else:
         print("no duplicates found.")
```

```
enter numbers in the list :  22 11 12 23 33 44 33 44 22 11
duplicates elemets are:  [22, 11, 33, 44]
```

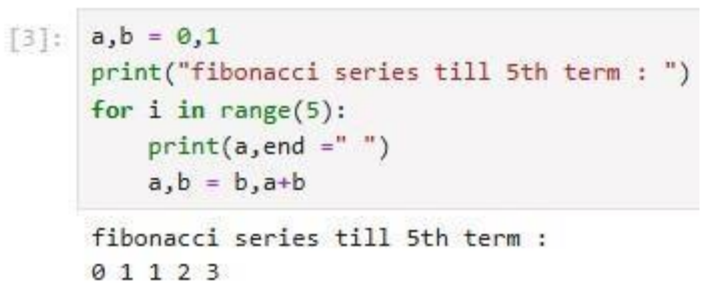
6. Write a python program to reverse a string.



```
[2]: text = input("enter string which you want to reverse : ")
     rev_text = text[::-1]
     print("reversed string : ",rev_text)
```

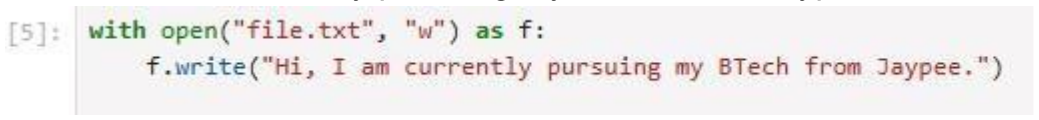
```
enter string which you want to reverse :  hello its RIVA
reversed string :  AVIR sti olleh
```

7. Write a python program to print fibonacci series till 5th term.



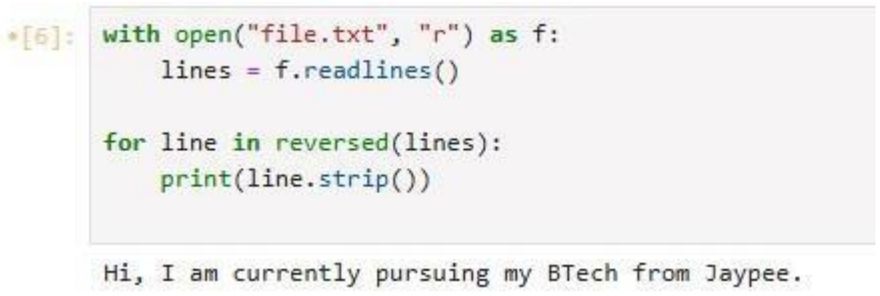
```
[3]: a,b = 0,1
     print("fibonacci series till 5th term : ")
     for i in range(5):
         print(a,end =" ")
         a,b = b,a+b

     fibonacci series till 5th term :
     0 1 1 2 3
```

8. Write a python program to read a txt file and replace the contents of the file with “Hi, I am currently pursuing my BTech from Jaypee.”



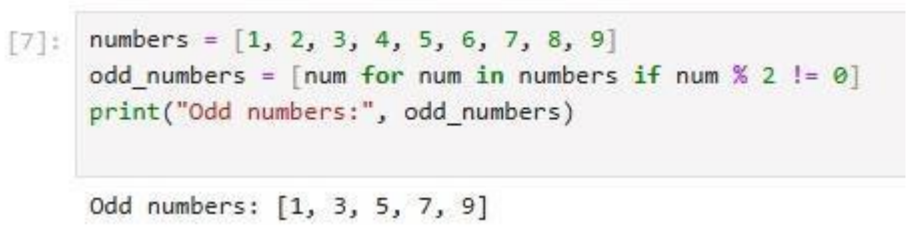
```
[5]: with open("file.txt", "w") as f:
         f.write("Hi, I am currently pursuing my BTech from Jaypee.")
```

9. Write a python program named ‘reverse.py’ to print lines of a file in reverse order.



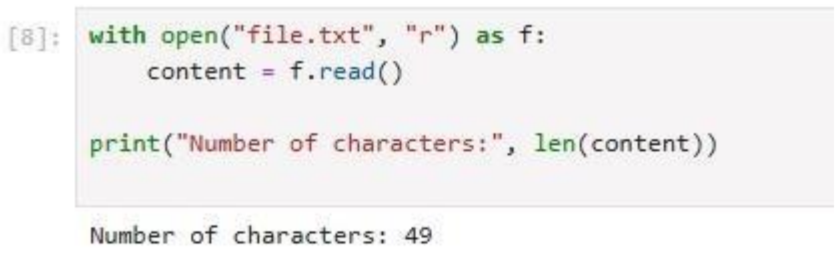
```
•[6]: with open("file.txt", "r") as f:
          lines = f.readlines()

      for line in reversed(lines):
          print(line.strip())

      Hi, I am currently pursuing my BTech from Jaypee.
```

10. Write a program to print odd elements from a list using list comprehension.



```
[7]: numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9]
     odd_numbers = [num for num in numbers if num % 2 != 0]
     print("Odd numbers:", odd_numbers)

     Odd numbers: [1, 3, 5, 7, 9]
```

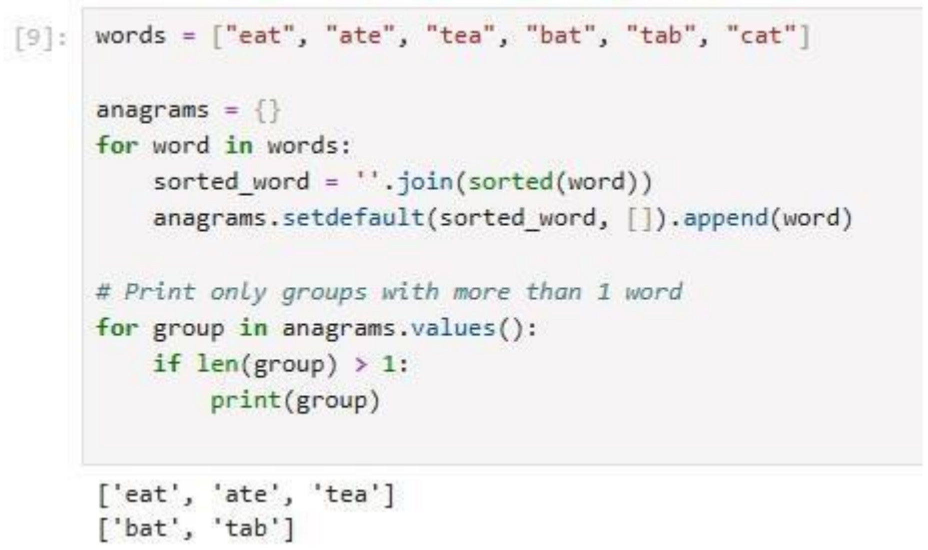
11. Write a program to read the file (from question 6) and count the number of characters in that file.



```
[8]: with open("file.txt", "r") as f:
         content = f.read()

     print("Number of characters:", len(content))

     Number of characters: 49
```

12. Write a program to find anagrams in a given list of words. Two words are called anagrams if one word can be formed by rearranging letters of another.

For example, ‘eat’, ‘ate’ and ‘tea’ are anagrams

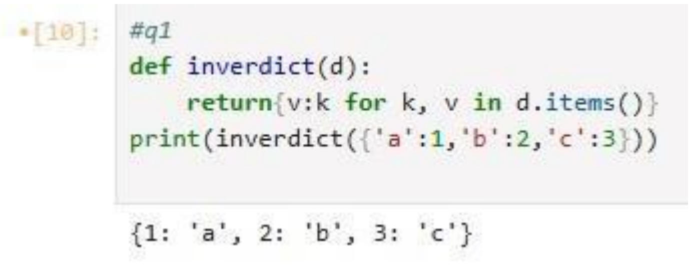


```
[9]: words = ["eat", "ate", "tea", "bat", "tab", "cat"]

     anagrams = {}
     for word in words:
         sorted_word = ''.join(sorted(word))
         anagrams.setdefault(sorted_word, []).append(word)

     # Print only groups with more than 1 word
     for group in anagrams.values():
         if len(group) > 1:
             print(group)
```

```
['eat', 'ate', 'tea']
['bat', 'tab']
```

ASSIGNMENT 3

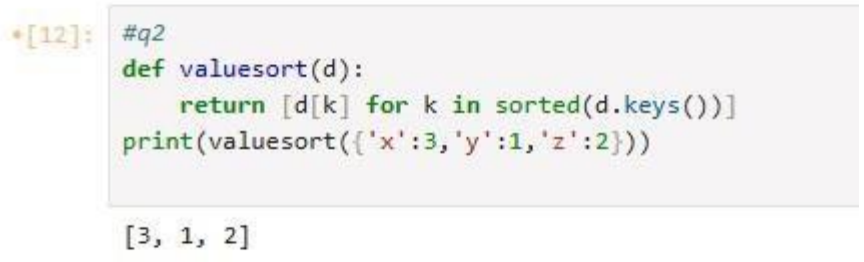
Q1. Write a function invertdict to interchange keys and values in a dictionary. For simplicity,assume that all values are unique.



```
•[10]: #q1
       def inverdict(d):
           return{v:k for k, v in d.items()}
       print(inverdict({'a':1,'b':2,'c':3}))
```

```
{1: 'a', 2: 'b', 3: 'c'}
```

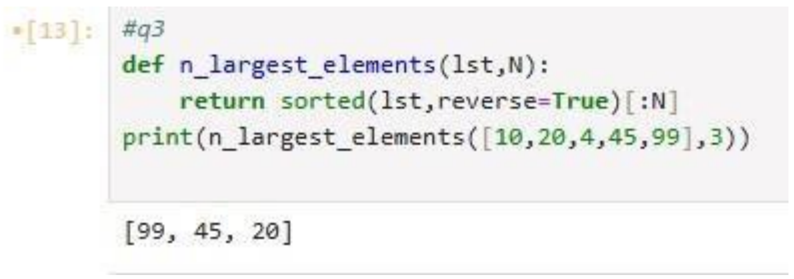
2. Write a function valuesort to sort values of a dictionary based on the key.



```
•[12]: #q2
       def valuesort(d):
           return [d[k] for k in sorted(d.keys())]
       print(valuesort({'x':3,'y':1,'z':2}))
```

```
[3, 1, 2]
```

3. Write a python program to find N largest elements assuming size of list is greater than or equal to N.

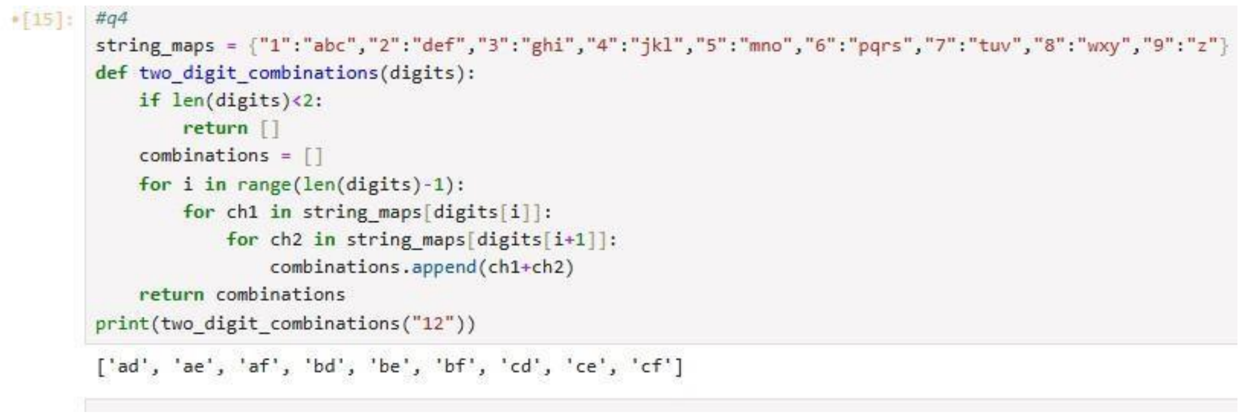


```
[13]: #q3
def n_largest_elements(lst,N):
    return sorted(lst,reverse=True)[:N]
print(n_largest_elements([10,20,4,45,99],3))
```

```
[99, 45, 20]
```

4. Write a Python program to get all possible two-digit letter combinations from a digit (1 to 9) string. string_maps = {

"1": "abc",
"2": "def",
"3": "ghi",
"4": "jkl",
"5": "mno", "6":
"pqrs",
"7": "tuv",
"8": "wxy",
"9": "z"
}



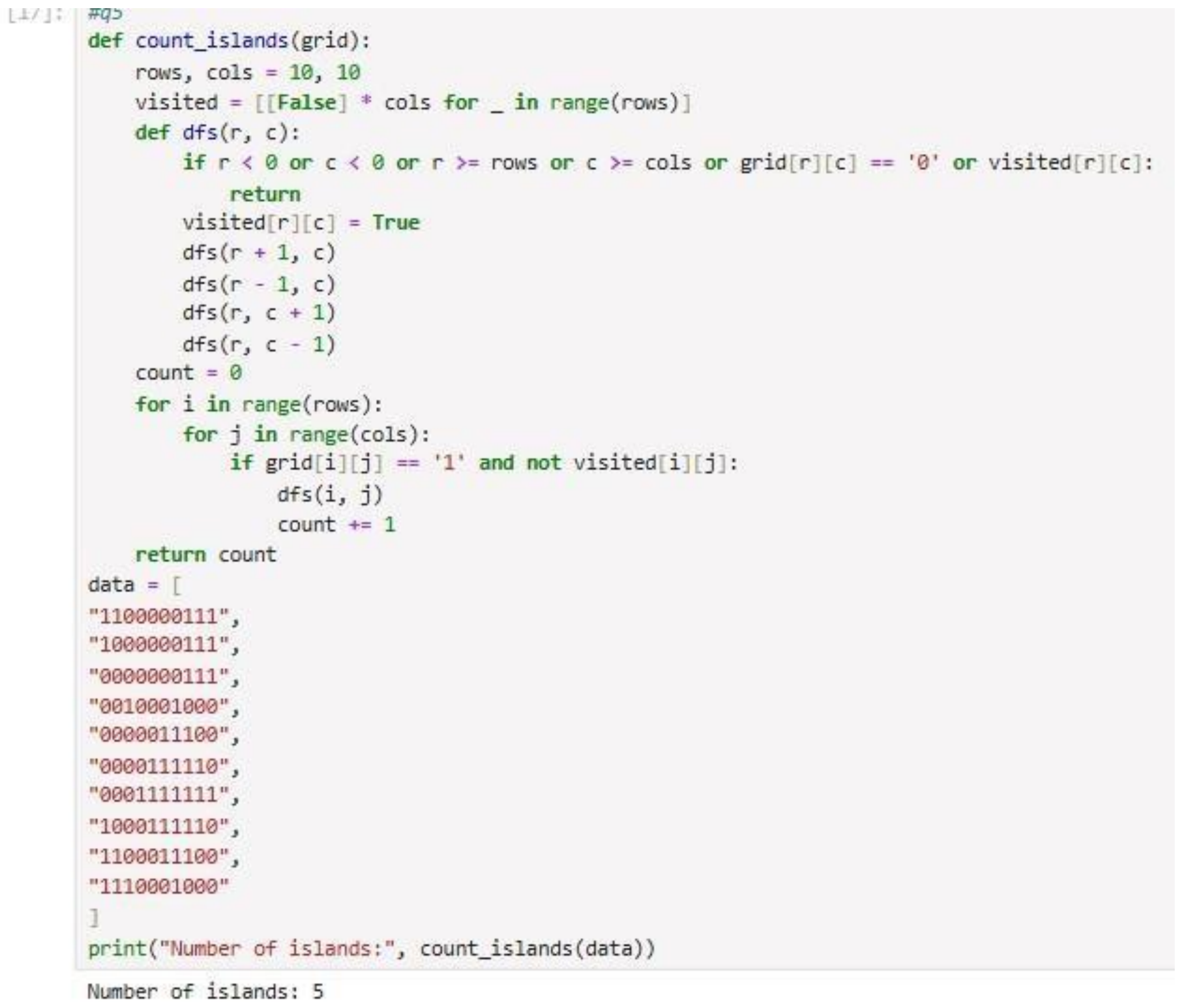
```
[15]: #q4
string_maps = {"1":"abc","2":"def","3":"ghi","4":"jkl","5":"mno","6":"pqrs","7":"tuv","8":"wxy","9":"z"}
def two_digit_combinations(digits):
    if len(digits)<2:
        return []
    combinations = []
    for i in range(len(digits)-1):
        for ch1 in string_maps[digits[i]]:
            for ch2 in string_maps[digits[i+1]]:
                combinations.append(ch1+ch2)
    return combinations
print(two_digit_combinations("12"))
```

```
['ad', 'ae', 'af', 'bd', 'be', 'bf', 'cd', 'ce', 'cf']
```

5. There are 10 vertical and horizontal squares on a plane. Each square is painted blue and green. Blue represents the sea, and green represents the land. When two green squares are in contact with the top and bottom, or right and left, they are said to be ground. The area created by only one green square is called "island". Write a Python program to read the mass data and find the number of islands.

Input:
A single data set is represented by 10 rows of 10 numbers representing green squares as 1 and blue squares as zeros.
1100000111

1000000111
0000000111
0010001000
0000011100
0000111110
0001111111
1000111110
1100011100
1110001000
Number of islands:
5



```
[17]: #q5
def count_islands(grid):
    rows, cols = 10, 10
    visited = [[False] * cols for _ in range(rows)]
    def dfs(r, c):
        if r < 0 or c < 0 or r >= rows or c >= cols or grid[r][c] == '0' or visited[r][c]:
            return
        visited[r][c] = True
        dfs(r + 1, c)
        dfs(r - 1, c)
        dfs(r, c + 1)
        dfs(r, c - 1)
    count = 0
    for i in range(rows):
        for j in range(cols):
            if grid[i][j] == '1' and not visited[i][j]:
                dfs(i, j)
                count += 1
    return count
data = [
"1100000111",
"1000000111",
"0000000111",
"0010001000",
"0000011100",
"0000111110",
"0001111111",
"1000111110",
"1100011100",
"1110001000"
]
print("Number of islands:", count_islands(data))
```

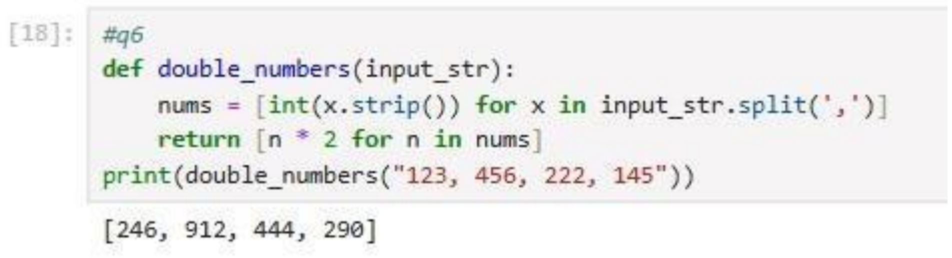
```
Number of islands: 5
```

6. Write a python program to read from input a string. The input contains

COMMA separated integers. The string can also contain spaces before and after COMMA. The output of the program is a list of integers that contains the DOUBLE of all numbers in the input. NOTE:

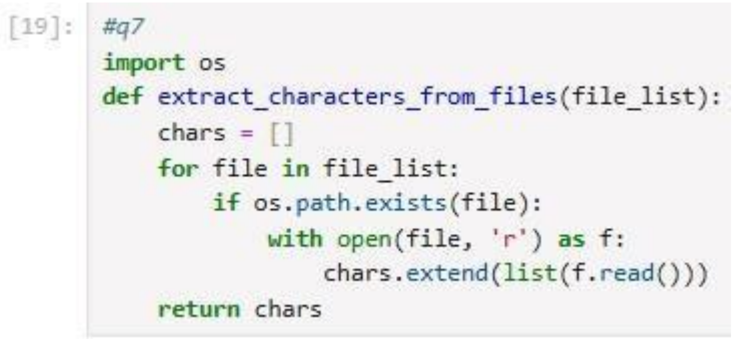
- The output value is computed as a list of integers.
- Use Python's print(...) to print it directly.
- Negative integers and ZERO are allowed in the input.
- Use python's split(...) function to process the input string.

Input:
123, 456, 222, 145 Output:
[246, 912, 444, 290] Input:
-1, 0, -2, 2, 0, 1 Output:
[-2, 0, -4, 4, 0, 2]



```
[18]: #q6
      def double_numbers(input_str):
          nums = [int(x.strip()) for x in input_str.split(',')]
          return [n * 2 for n in nums]
      print(double_numbers("123, 456, 222, 145"))

      [246, 912, 444, 290]
```

7. Write a Python program to extract characters from various text files and put them into a list.



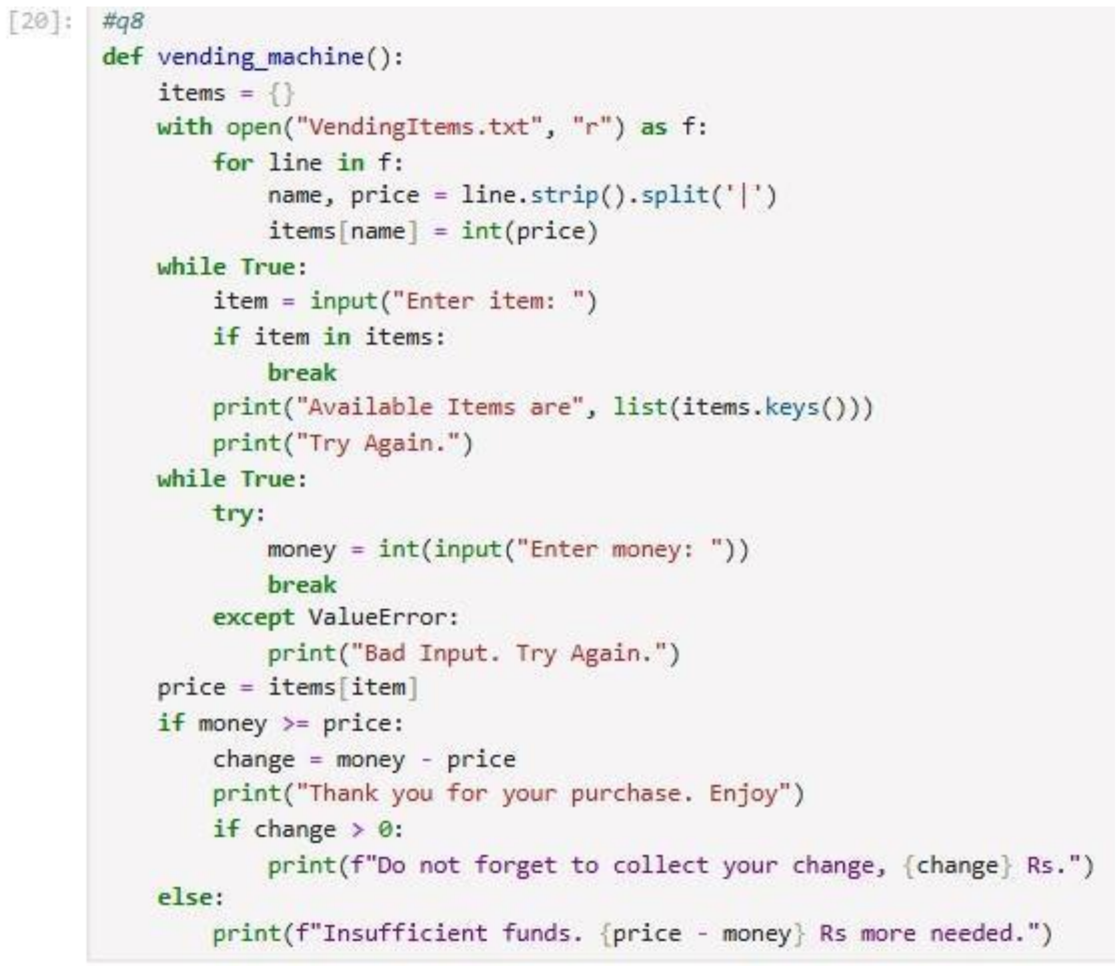
```
[19]: #q7
      import os
      def extract_characters_from_files(file_list):
          chars = []
          for file in file_list:
              if os.path.exists(file):
                  with open(file, 'r') as f:
                      chars.extend(list(f.read()))
          return chars
```

8. Vending Machine

In this program, you have to simulate a simple Vending Machine. You need to read a file,VendingItems.txt. It should contain Item-names and their Prices (integers) separated by a PIPE (|). Here is a sample:

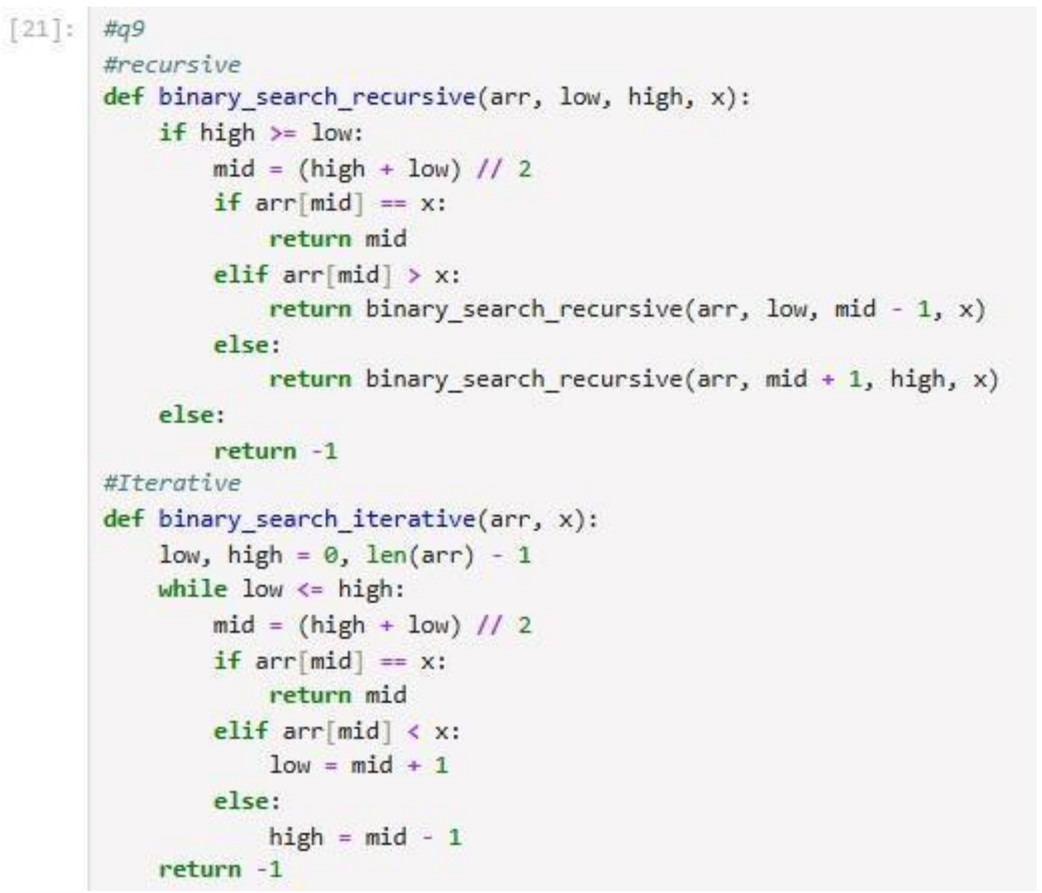
Potato Chips|20
Popcorn|30
Chocolate|15
Biscuit|10
Soft Drink|12

Store the lines read in a dictionary, mapping items to their respective prices. The input to the program will consist of an item name. If the item is not a valid item (i.e., an item not available in the vending machine), you need to ask the user to try again. Use exceptions to handle the missing items.
After a proper item is read, the program expects the user to provide the money to deposit in the vending machine (integer). Again, the program will ask the user to try again till a proper integer is read (using exception handling).
Finally, depending upon the money deposited by the user, the Machine will display the appropriate message(s).
EXAMPLE INTERACTION (INPUT and OUTPUT are intermixed. INPUTs are shaded): batata vada
Available Items are ['Potato Chips', 'Popcorn', 'Chocolate', 'Biscuit', 'Soft Drink'].
Try Again. pani-puri
Available Items are ['Potato Chips', 'Popcorn', 'Chocolate', 'Biscuit', 'Soft Drink']. Try Again. Potato Chips credit-card Bad Input credit-card.
Try Again. debit-card Bad
Input debit-card.
Try Again. 25
Thank you for your purchase. Enjoy Do not forget to collect your change, 5 Rs.



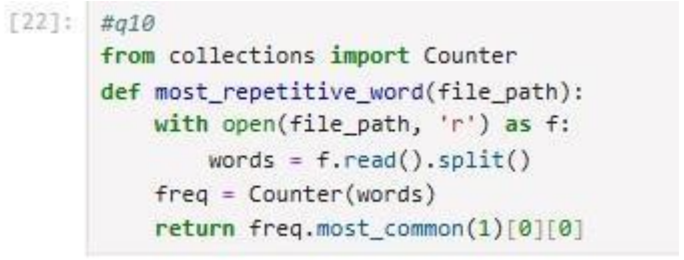
```
[20]: #q8
def vending_machine():
    items = {}
    with open("VendingItems.txt", "r") as f:
        for line in f:
            name, price = line.strip().split('|')
            items[name] = int(price)
    while True:
        item = input("Enter item: ")
        if item in items:
            break
        print("Available Items are", list(items.keys()))
        print("Try Again.")
    while True:
        try:
            money = int(input("Enter money: "))
            break
        except ValueError:
            print("Bad Input. Try Again.")
    price = items[item]
    if money >= price:
        change = money - price
        print("Thank you for your purchase. Enjoy")
        if change > 0:
            print(f"Do not forget to collect your change, {change} Rs.")
    else:
        print(f"Insufficient funds. {price - money} Rs more needed.")
```

9. Write a Python program for binary search using recursive as well as iterative method.



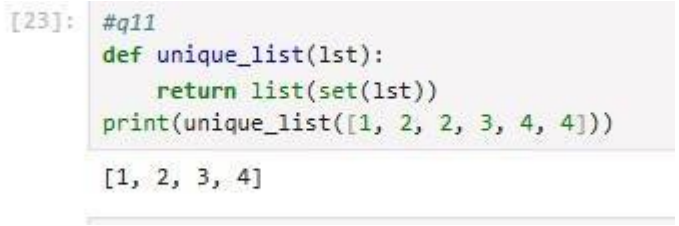
[21]:
```python
#q9
#recursive
def binary_search_recursive(arr, low, high, x):
    if high >= low:
        mid = (high + low) // 2
        if arr[mid] == x:
            return mid
        elif arr[mid] > x:
            return binary_search_recursive(arr, low, mid - 1, x)
        else:
            return binary_search_recursive(arr, mid + 1, high, x)
    else:
        return -1
#Iterative
def binary_search_iterative(arr, x):
    low, high = 0, len(arr) - 1
    while low <= high:
        mid = (high + low) // 2
        if arr[mid] == x:
            return mid
        elif arr[mid] < x:
            low = mid + 1
        else:
            high = mid - 1
    return -1
```

10.Write a Python program to find the most repetitive word in a text file.



[22]:
```python
#q10
from collections import Counter
def most_repetitive_word(file_path):
    with open(file_path, 'r') as f:
        words = f.read().split()
    freq = Counter(words)
    return freq.most_common(1)[0][0]
```

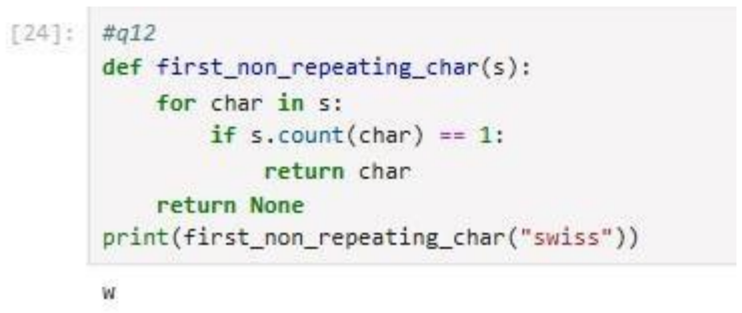
11. Write a Python function that takes a list and returns a new list with unique elements of the first list.



[23]:
```python
#q11
def unique_list(lst):
    return list(set(lst))
print(unique_list([1, 2, 2, 3, 4, 4]))
```

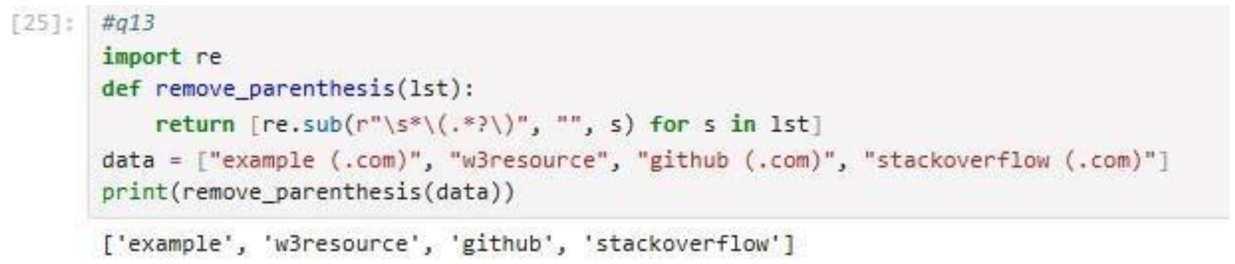
```
[1, 2, 3, 4]
```

12. Write a Python program to find the first non-repeating character in given string.



```
[24]: #q12
      def first_non_repeating_char(s):
          for char in s:
              if s.count(char) == 1:
                  return char
          return None
      print(first_non_repeating_char("swiss"))

      w
```

13. Write a Python program to remove the parenthesis area in a string. Sample data:
["example (.com)", "w3resource", "github (.com)", "stackoverflow (.com)"] Expected Output: example
w3resource github stackoverflow



```
[25]: #q13
      import re
      def remove_parenthesis(lst):
          return [re.sub(r"\s*\(.*?\)", "", s) for s in lst]
      data = ["example (.com)", "w3resource", "github (.com)", "stackoverflow (.com)"]
      print(remove_parenthesis(data))

      ['example', 'w3resource', 'github', 'stackoverflow']
```