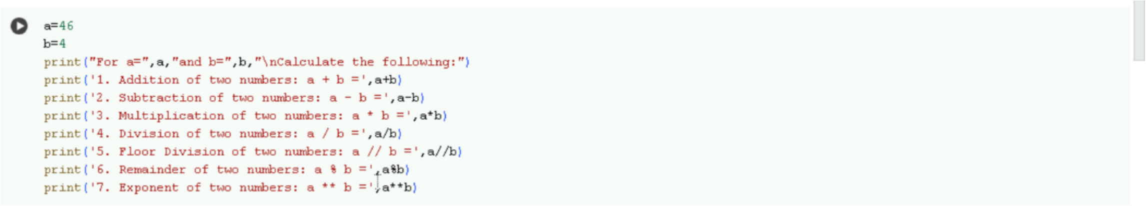
Python Basics

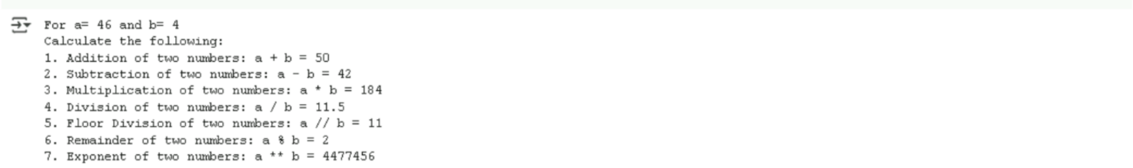
OPERATORS

1.Arithmetic Operators



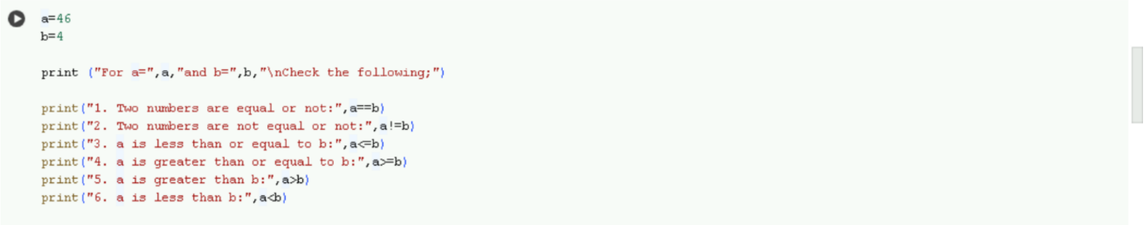
```
a=46
b=4
print("For a=",a,"and b=",b,"\nCalculate the following:")
print('1. Addition of two numbers: a + b =',a+b)
print('2. Subtraction of two numbers: a - b =',a-b)
print('3. Multiplication of two numbers: a * b =',a*b)
print('4. Division of two numbers: a / b =',a/b)
print('5. Floor Division of two numbers: a // b =',a//b)
print('6. Remainder of two numbers: a % b =',a%b)
print('7. Exponent of two numbers: a ** b =',a**b)
```

Output:



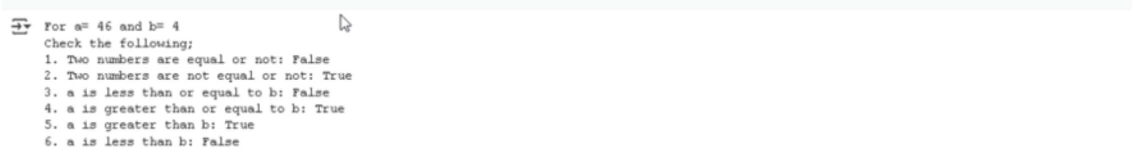
```
For a= 46 and b= 4
Calculate the following:
1. Addition of two numbers: a + b = 50
2. Subtraction of two numbers: a - b = 42
3. Multiplication of two numbers: a * b = 184
4. Division of two numbers: a / b = 11.5
5. Floor Division of two numbers: a // b = 11
6. Remainder of two numbers: a % b = 2
7. Exponent of two numbers: a ** b = 4477456
```

2. Comparison Operators



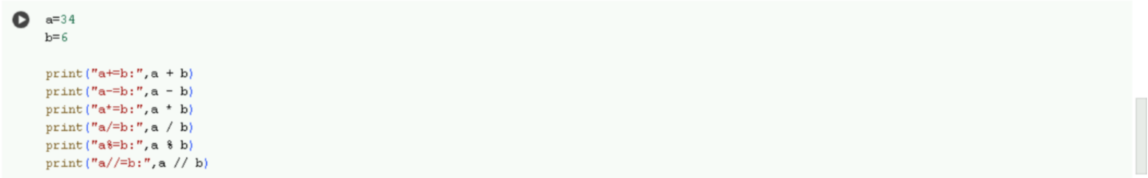
```
a=46
b=4

print ("For a=",a,"and b=",b,"\nCheck the following;")

print("1. Two numbers are equal or not:",a==b)
print("2. Two numbers are not equal or not:",a!=b)
print("3. a is less than or equal to b:",a<=b)
print("4. a is greater than or equal to b:",a>=b)
print("5. a is greater than b:",a>b)
print("6. a is less than b:",a<b)
```

Output:



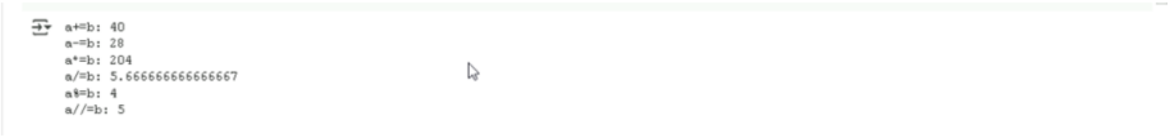
```
For a= 46 and b= 4
Check the following;
1. Two numbers are equal or not: False
2. Two numbers are not equal or not: True
3. a is less than or equal to b: False
4. a is greater than or equal to b: True
5. a is greater than b: True
6. a is less than b: False
```

3.Assignment Operators



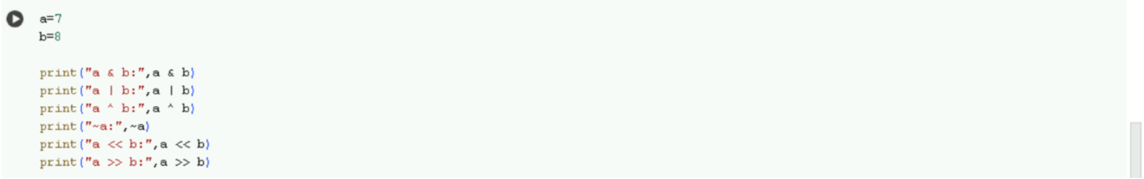
```
a=34
b=6

print("a+=b:",a + b)
print("a-=b:",a - b)
print("a*=b:",a * b)
print("a/=b:",a / b)
print("a%=b:",a % b)
print("a//=b:",a // b)
```

Output:



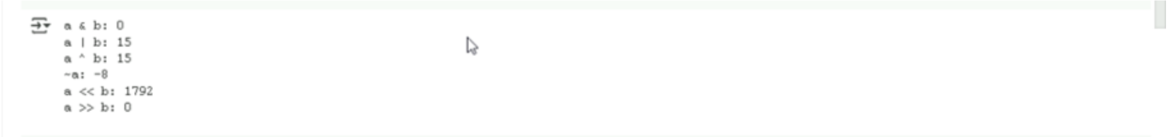
```
a+=b: 40
a-=b: 28
a*=b: 204
a/=b: 5.666666666666667
a%=b: 4
a//=b: 5
```

4.Bitwise Operators



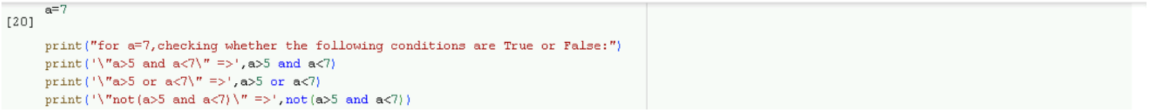
```
a=7
b=8

print("a & b:",a & b)
print("a | b:",a | b)
print("a ^ b:",a ^ b)
print("~a:",~a)
print("a << b:",a << b)
print("a >> b:",a >> b)
```

Output:



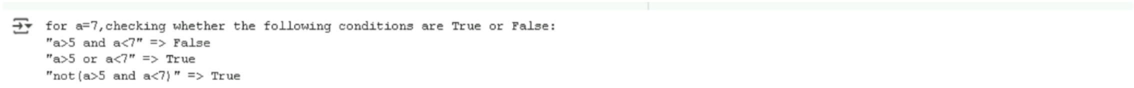
```
a & b: 0
a | b: 15
a ^ b: 15
~a: -8
a << b: 1792
a >> b: 0
```

5.Logical Operators



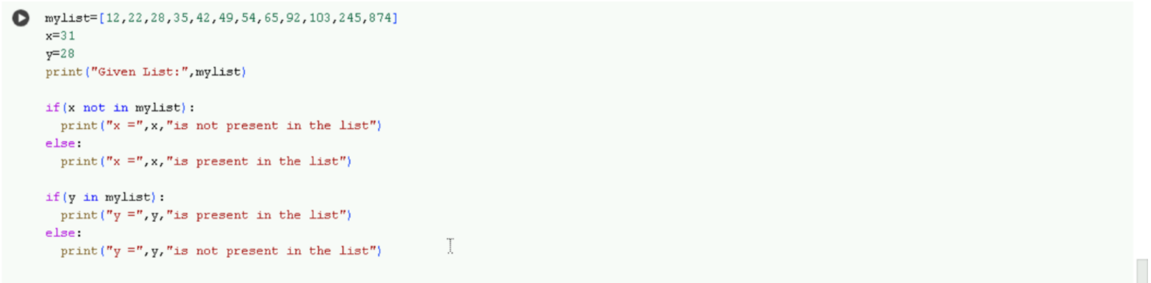
```
[20]
a=7

print("for a=7,checking whether the following conditions are True or False:")
print('\"a>5 and a<7\" =>',a>5 and a<7)
print('\"a>5 or a<7\" =>',a>5 or a<7)
print('\"not(a>5 and a<7)\" =>',not(a>5 and a<7))
```

Output:



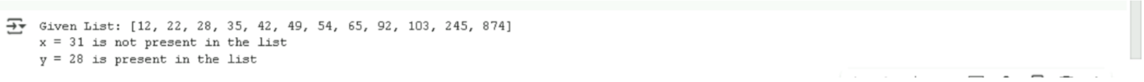
```
for a=7,checking whether the following conditions are True or False:
"a>5 and a<7" => False
"a>5 or a<7" => True
"not(a>5 and a<7)" => True
```

6.Membership Operators



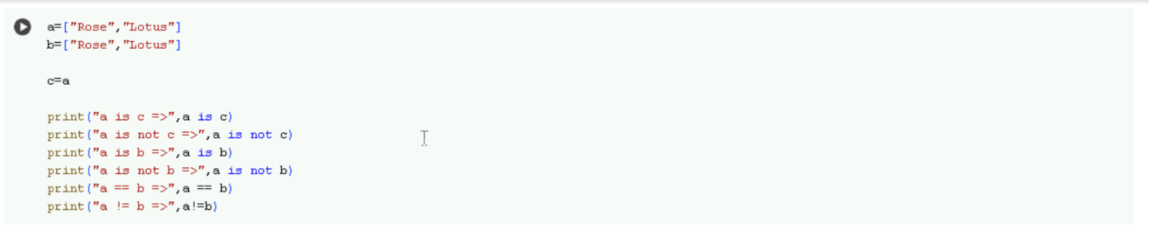
```
mylist=[12,22,28,35,42,49,54,65,92,103,245,874]
x=31
y=28
print("Given List:",mylist)

if(x not in mylist):
  print("x =",x,"is not present in the list")
else:
  print("x =",x,"is present in the list")

if(y in mylist):
  print("y =",y,"is present in the list")
else:
  print("y =",y,"is not present in the list")
```

Output:



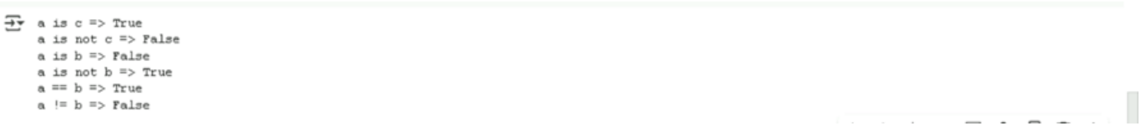
```
Given List: [12, 22, 28, 35, 42, 49, 54, 65, 92, 103, 245, 874]
x = 31 is not present in the list
y = 28 is present in the list
```

7.Identity Operators



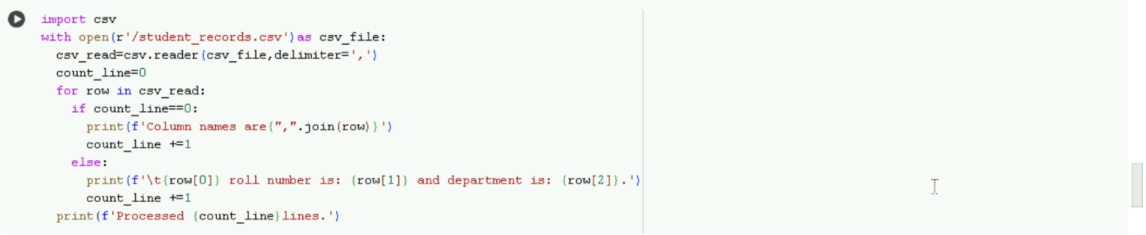
```
a=["Rose","Lotus"]
b=["Rose","Lotus"]

c=a

print("a is c =>",a is c)
print("a is not c =>",a is not c)
print("a is b =>",a is b)
print("a is not b =>",a is not b)
print("a == b =>",a == b)
print("a != b =>",a!=b)
```

Output:



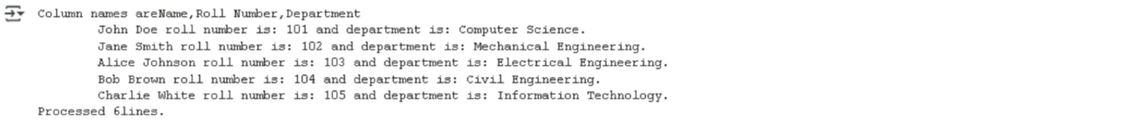
```
a is c => True
a is not c => False
a is b => False
a is not b => True
a == b => True
a != b => False
```

8. How to read csv file in python



```
import csv
with open(r'/student_records.csv')as csv_file:
  csv_read=csv.reader(csv_file,delimiter=',')
  count_line=0
  for row in csv_read:
    if count_line==0:
      print(f'Column names are{",".join(row)}')
      count_line +=1
    else:
      print(f'\t{row[0]} roll number is: {row[1]} and department is: {row[2]}.')
      count_line +=1
  print(f'Processed {count_line}lines.')
```

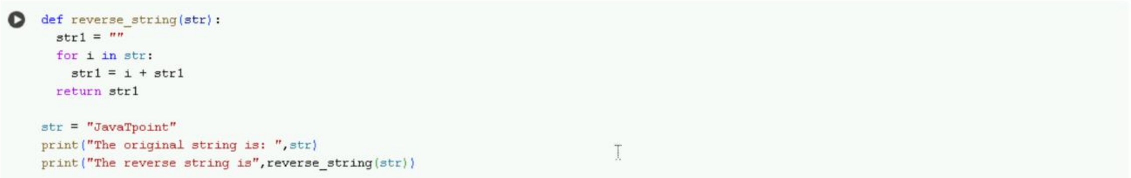
Output:



```
Column names areName,Roll Number,Department
        John Doe roll number is: 101 and department is: Computer Science.
        Jane Smith roll number is: 102 and department is: Mechanical Engineering.
        Alice Johnson roll number is: 103 and department is: Electrical Engineering.
        Bob Brown roll number is: 104 and department is: Civil Engineering.
        Charlie White roll number is: 105 and department is: Information Technology.
Processed 6lines.
```

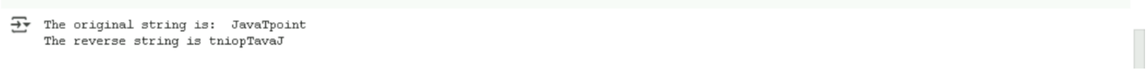
How to reverse a string in python

9.Using for loop



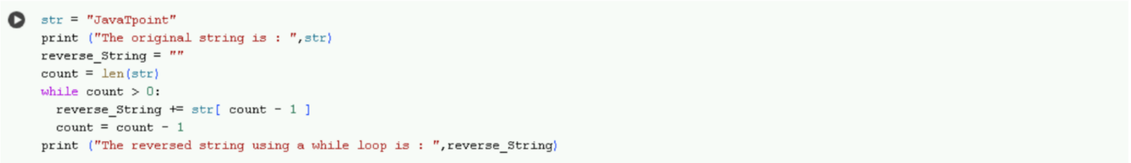
```
def reverse_string(str):
  str1 = ""
  for i in str:
    str1 = i + str1
  return str1

str = "JavaTpoint"
print("The original string is: ",str)
print("The reverse string is",reverse_string(str))
```

Output:



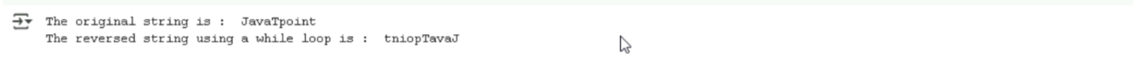
```
The original string is:  JavaTpoint
The reverse string is tniopTavaJ
```

10. Using while loop



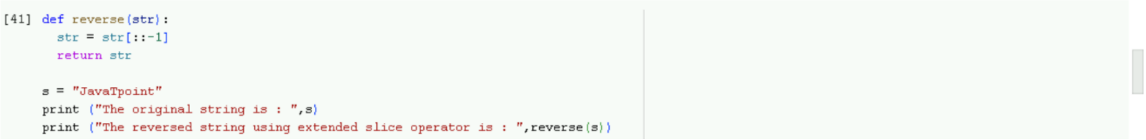
```
str = "JavaTpoint"
print ("The original string is : ",str)
reverse_String = ""
count = len(str)
while count > 0:
  reverse_String += str[ count - 1 ]
  count = count - 1
print ("The reversed string using a while loop is : ",reverse_String)
```

Output:



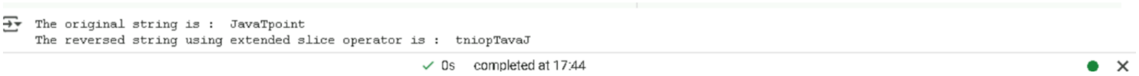
```
The original string is :  JavaTpoint
The reversed string using a while loop is :  tniopTavaJ
```

11. Using the slice([]) Operator



```
[41] def reverse(str):
       str = str[::-1]
       return str

     s = "JavaTpoint"
     print ("The original string is : ",s)
     print ("The reversed string using extended slice operator is : ",reverse(s))
```

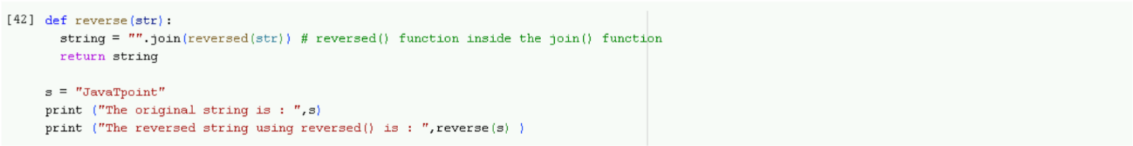
Output:



```
The original string is :  JavaTpoint
The reversed string using extended slice operator is :  tniopTavaJ
```

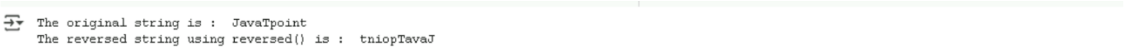
✓ 0s completed at 17:44

12. Using reverse function with join



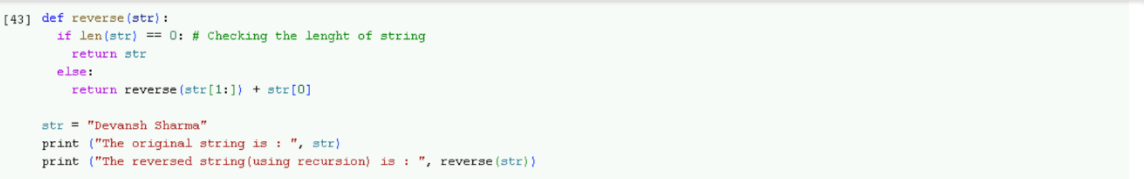
```
[42] def reverse(str):
        string = "".join(reversed(str)) # reversed() function inside the join() function
        return string

     s = "JavaTpoint"
     print ("The original string is : ",s)
     print ("The reversed string using reversed() is : ",reverse(s) )
```

Output:



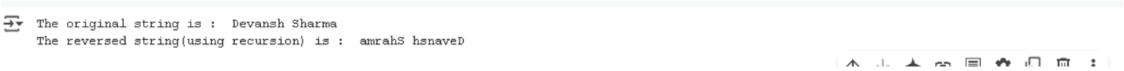
```
The original string is :  JavaTpoint
The reversed string using reversed() is :  tniopTavaJ
```

13. Using recursion ()



```
[43] def reverse(str):
       if len(str) == 0: # Checking the lenght of string
         return str
       else:
         return reverse(str[1:]) + str[0]

     str = "Devansh Sharma"
     print ("The original string is : ", str)
     print ("The reversed string(using recursion) is : ", reverse(str))
```

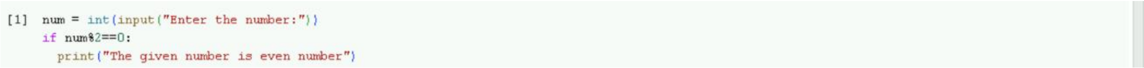
Output:



```
The original string is :  Devansh Sharma
The reversed string(using recursion) is :  amrahS hsnaveD
```

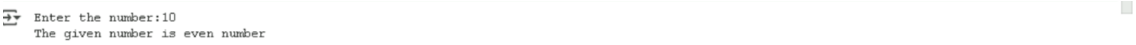
If Statement

14. Program to find even number.



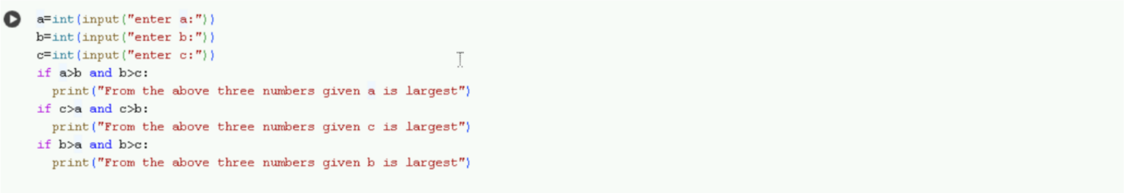
```
[1] num = int(input("Enter the number:"))
    if num%2==0:
      print("The given number is even number")
```

Output:



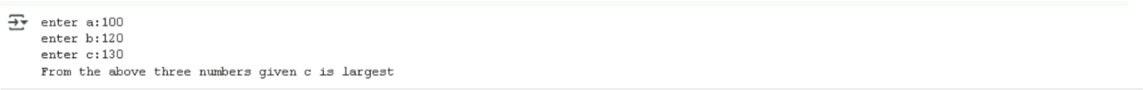
```
Enter the number:10
The given number is even number
```

15. Program to print the largest of the three numbers.



```
a=int(input("enter a:"))
b=int(input("enter b:"))
c=int(input("enter c:"))
if a>b and b>c:
  print("From the above three numbers given a is largest")
if c>a and c>b:
  print("From the above three numbers given c is largest")
if b>a and b>c:
  print("From the above three numbers given b is largest")
```

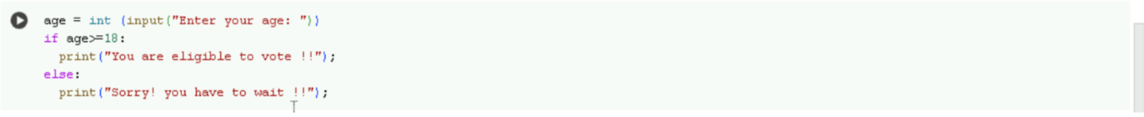
Output:



```
enter a:100
enter b:120
enter c:130
From the above three numbers given c is largest
```

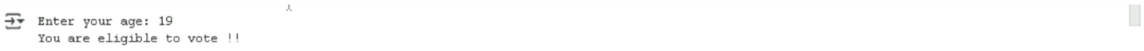
If-else Statement

16.Program to check whether a person is eligible to vote or not.



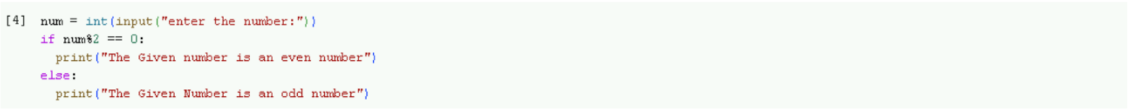
```
age = int (input("Enter your age: "))
if age>=18:
  print("You are eligible to vote !!");
else:
  print("Sorry! you have to wait !!");
```

Output:



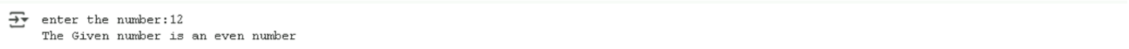
```
Enter your age: 19
You are eligible to vote !!
```

17. Program to check whether a number is even or not.



```
[4] num = int(input("enter the number:"))
    if num%2 == 0:
      print("The Given number is an even number")
    else:
      print("The Given Number is an odd number")
```

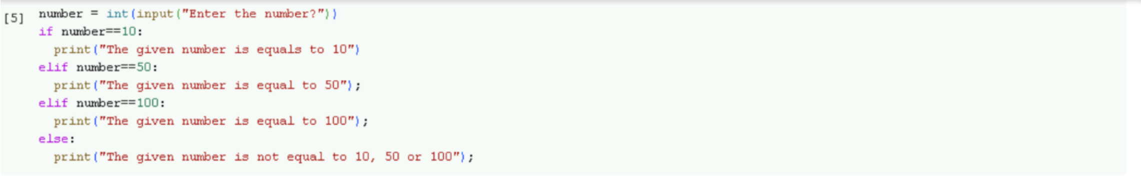
Output:



```
enter the number:12
The Given number is an even number
```

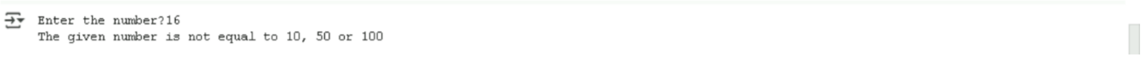
Elif Statement

18. Program to check whether a number is equal to 10, 50 or 100 or not.



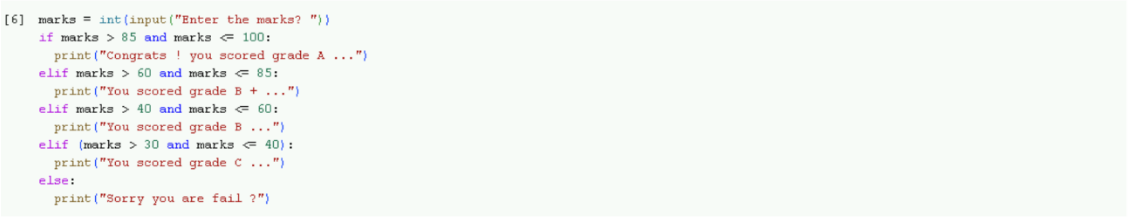
```
[5] number = int(input("Enter the number?"))
    if number==10:
      print("The given number is equals to 10")
    elif number==50:
      print("The given number is equal to 50");
    elif number==100:
      print("The given number is equal to 100");
    else:
      print("The given number is not equal to 10, 50 or 100");
```

Output:



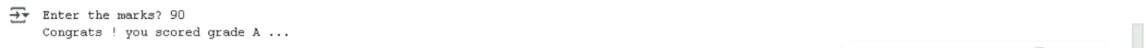
```
Enter the number?16
The given number is not equal to 10, 50 or 100
```

19. Program to check the grade of the student.



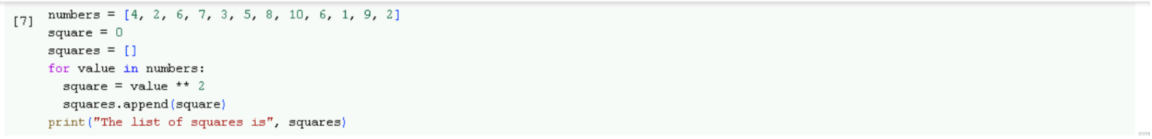
```
[6] marks = int(input("Enter the marks? "))
    if marks > 85 and marks <= 100:
      print("Congrats ! you scored grade A ...")
    elif marks > 60 and marks <= 85:
      print("You scored grade B + ...")
    elif marks > 40 and marks <= 60:
      print("You scored grade B ...")
    elif (marks > 30 and marks <= 40):
      print("You scored grade C ...")
    else:
      print("Sorry you are fail ?")
```

Output:



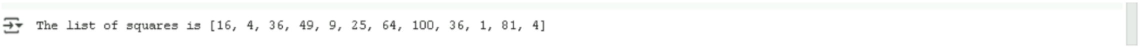
```
Enter the marks? 90
Congrats ! you scored grade A ...
```

20.



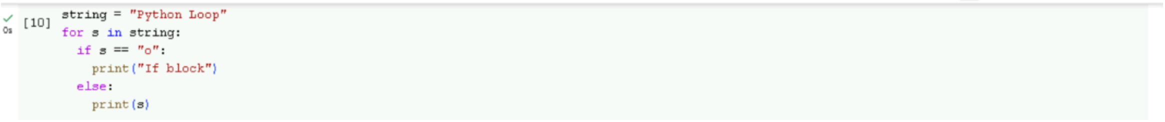
```
[7] numbers = [4, 2, 6, 7, 3, 5, 8, 10, 6, 1, 9, 2]
    square = 0
    squares = []
    for value in numbers:
      square = value ** 2
      squares.append(square)
    print("The list of squares is", squares)
```

Output:



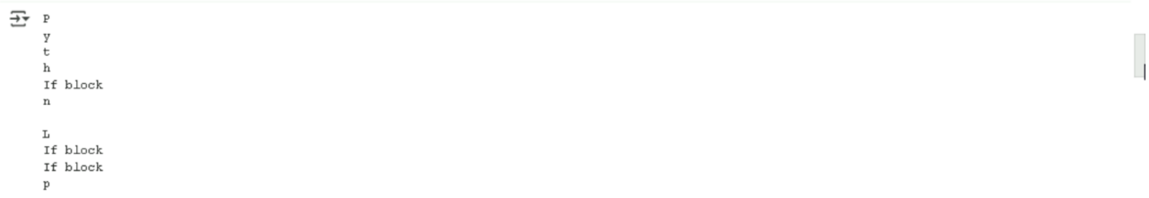
```
The list of squares is [16, 4, 36, 49, 9, 25, 64, 100, 36, 1, 81, 4]
```

21. For Loop



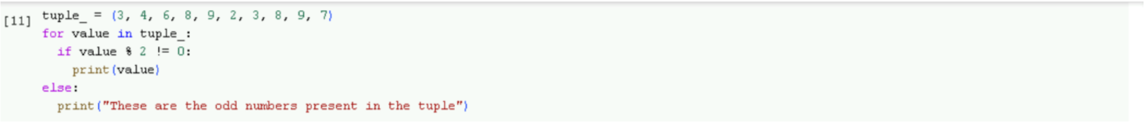
```
[10] string = "Python Loop"
     for s in string:
       if s == "o":
         print("If block")
       else:
         print(s)
```

Output:



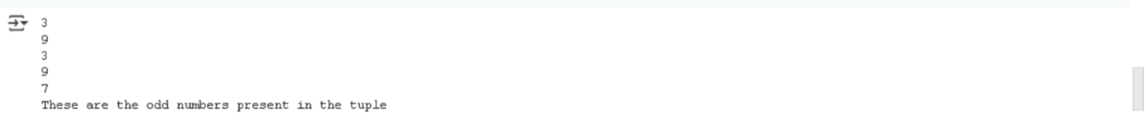
```
P
y
t
h
If block
n

L
If block
If block
p
```

22. Using else Statement with for Loop



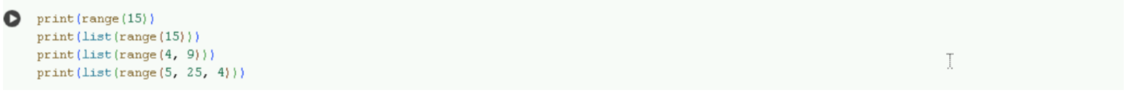
```
[11] tuple_ = (3, 4, 6, 8, 9, 2, 3, 8, 9, 7)
     for value in tuple_:
       if value % 2 != 0:
         print(value)
     else:
       print("These are the odd numbers present in the tuple")
```

Output:



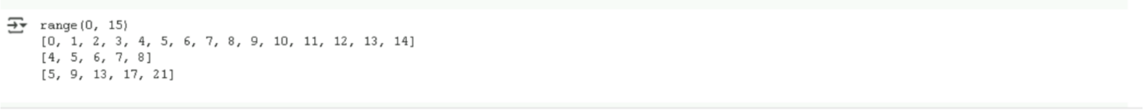
```
3
9
3
9
7
These are the odd numbers present in the tuple
```

23. Program to show the working of range() function



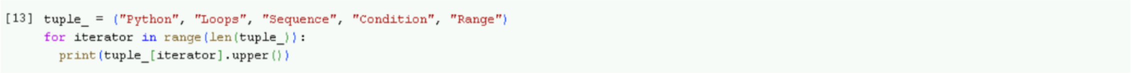
```
print(range(15))
print(list(range(15)))
print(list(range(4, 9)))
print(list(range(5, 25, 4)))
```

Output:



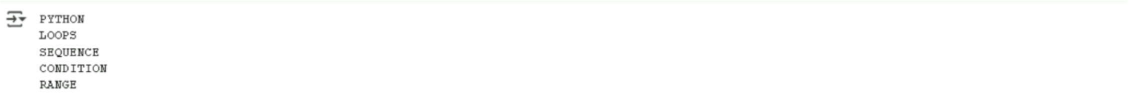
```
range(0, 15)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]
[4, 5, 6, 7, 8]
[5, 9, 13, 17, 21]
```

24. Python program to iterate over a sequence with the help of indexing.



```
[13] tuple_ = ("Python", "Loops", "Sequence", "Condition", "Range")
     for iterator in range(len(tuple_)):
       print(tuple_[iterator].upper())
```

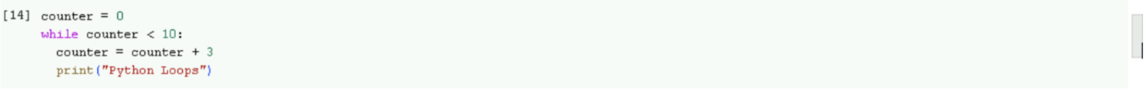
Output:



```
PYTHON
LOOPS
SEQUENCE
CONDITION
RANGE
```

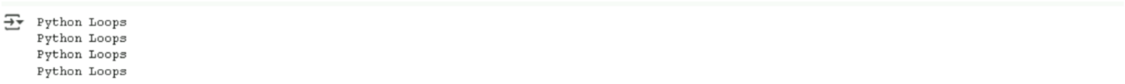
While Loop

25. Python program to show how to use a while loop



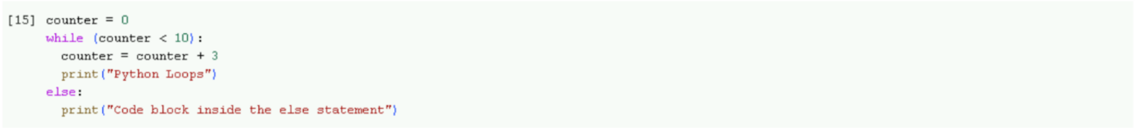
```
[14] counter = 0
     while counter < 10:
       counter = counter + 3
       print("Python Loops")
```

Output:



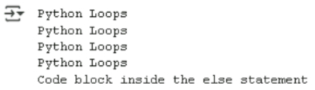
```
Python Loops
Python Loops
Python Loops
Python Loops
```

26. Python program to show how to use else statement with the while loop.



```
[15] counter = 0
     while (counter < 10):
       counter = counter + 3
       print("Python Loops")
     else:
       print("Code block inside the else statement")
```

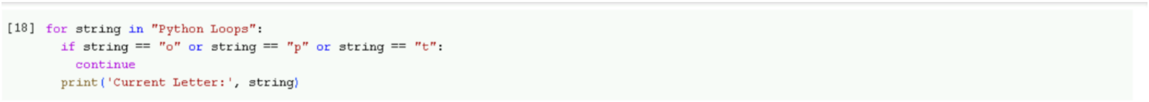
Output:



```
Python Loops
Python Loops
Python Loops
Python Loops
Code block inside the else statement
```

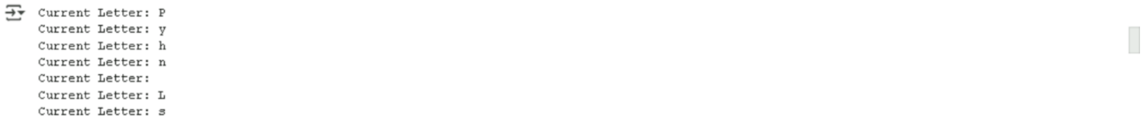
Continue Statement

27. Python program to show how the continue statement works



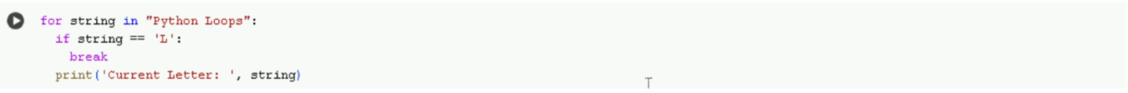
```
[18] for string in "Python Loops":
       if string == "o" or string == "p" or string == "t":
         continue
       print('Current Letter:', string)
```

Output:



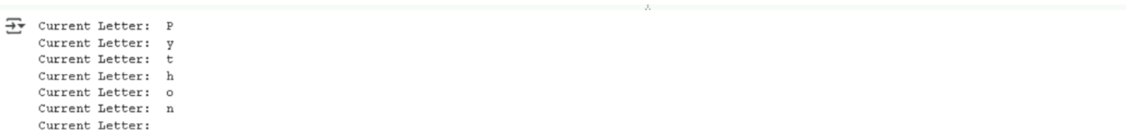
```
Current Letter: P
Current Letter: y
Current Letter: h
Current Letter: n
Current Letter:
Current Letter: L
Current Letter: s
```

28. Break Statement



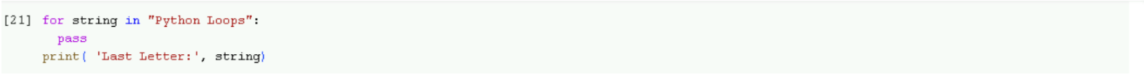
```
for string in "Python Loops":
  if string == 'L':
    break
  print('Current Letter: ', string)
```

Output:



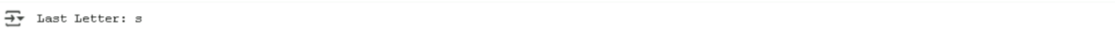
```
Current Letter:  P
Current Letter:  y
Current Letter:  t
Current Letter:  h
Current Letter:  o
Current Letter:  n
Current Letter:
```

29. Pass Statement



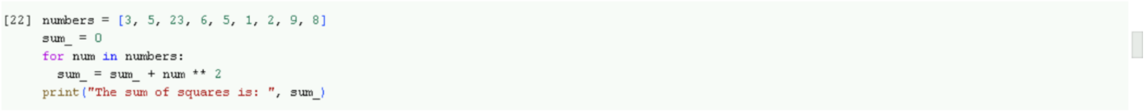
```
[21] for string in "Python Loops":
       pass
     print( 'Last Letter:', string)
```

Output:



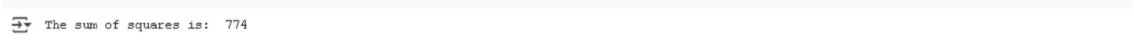
```
Last Letter: s
```

30. Code to find the sum of squares of each element of the list using for loop



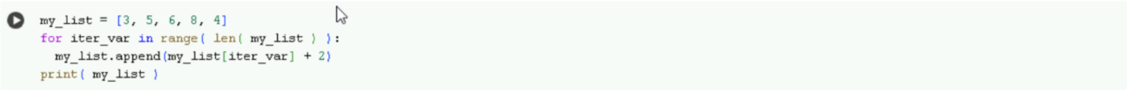
```
[22] numbers = [3, 5, 23, 6, 5, 1, 2, 9, 8]
     sum_ = 0
     for num in numbers:
       sum_ = sum_ + num ** 2
     print("The sum of squares is: ", sum_)
```

Output:



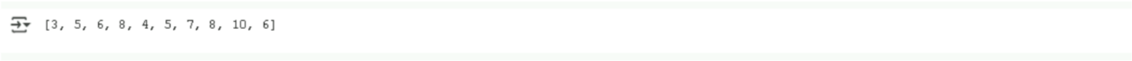
```
The sum of squares is:  774
```

31.range() function



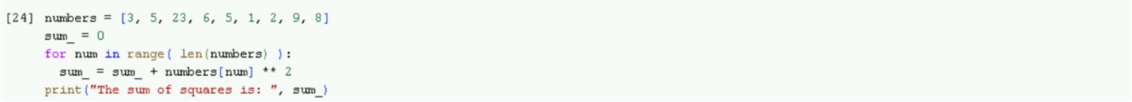
```
my_list = [3, 5, 6, 8, 4]
for iter_var in range( len( my_list ) ):
  my_list.append(my_list[iter_var] + 2)
print( my_list )
```

Output:



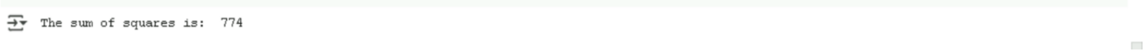
```
[3, 5, 6, 8, 4, 5, 7, 8, 10, 6]
```

32. Code to find the sum of squares of each element of the list using for loop.



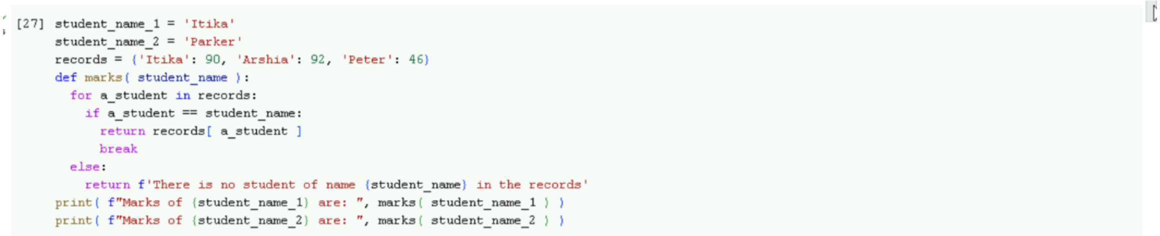
```
[24] numbers = [3, 5, 23, 6, 5, 1, 2, 9, 8]
     sum_ = 0
     for num in range( len(numbers) ):
       sum_ = sum_ + numbers[num] ** 2
     print("The sum of squares is: ", sum_)
```

Output:



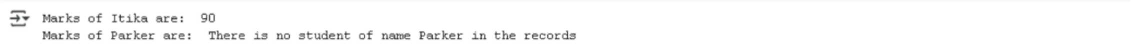
```
The sum of squares is:  774
```

33. Code to print marks of a student from the record



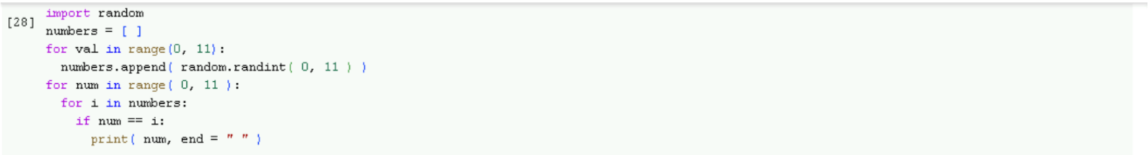
```
[27] student_name_1 = 'Itika'
     student_name_2 = 'Parker'
     records = {'Itika': 90, 'Arshia': 92, 'Peter': 46}
     def marks( student_name ):
       for a_student in records:
         if a_student == student_name:
           return records[ a_student ]
           break
       else:
         return f'There is no student of name {student_name} in the records'
     print( f"Marks of {student_name_1} are: ", marks( student_name_1 ) )
     print( f"Marks of {student_name_2} are: ", marks( student_name_2 ) )
```

Output:



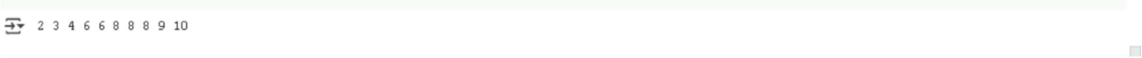
```
Marks of Itika are:  90
Marks of Parker are:  There is no student of name Parker in the records
```

34. Nested Loops



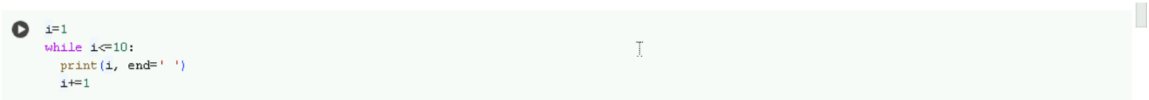
```
[28] import random
     numbers = [ ]
     for val in range(0, 11):
       numbers.append( random.randint( 0, 11 ) )
     for num in range( 0, 11 ):
       for i in numbers:
         if num == i:
           print( num, end = " " )
```

Output:



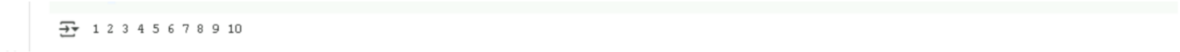
```
2 3 4 6 6 8 8 8 9 10
```

35. code examples of while loops in Python for printing numbers from 1 to 10.



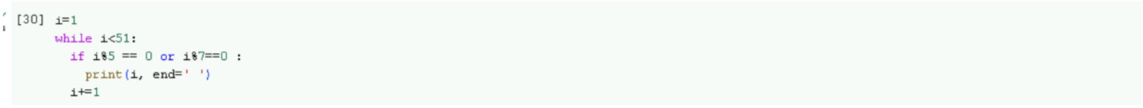
```
i=1
while i<=10:
  print(i, end=' ')
  i+=1
```

Output:



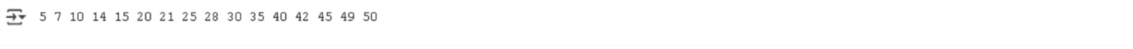
```
1 2 3 4 5 6 7 8 9 10
```

36. code examples of while loops in Python for Printing those numbers divisible by either 5 or 7 within 1 to 50 using a while loop.



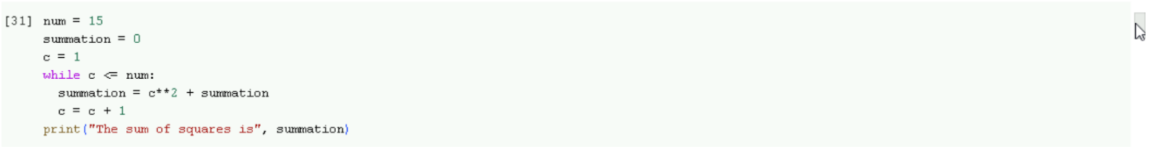
```
[30] i=1
     while i<51:
       if i%5 == 0 or i%7==0 :
         print(i, end=' ')
       i+=1
```

Output:



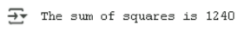
```
5 7 10 14 15 20 21 25 28 30 35 40 42 45 49 50
```

37. Code examples of while loops in Python, the sum of squares of the first 15 natural numbers using a while loop.



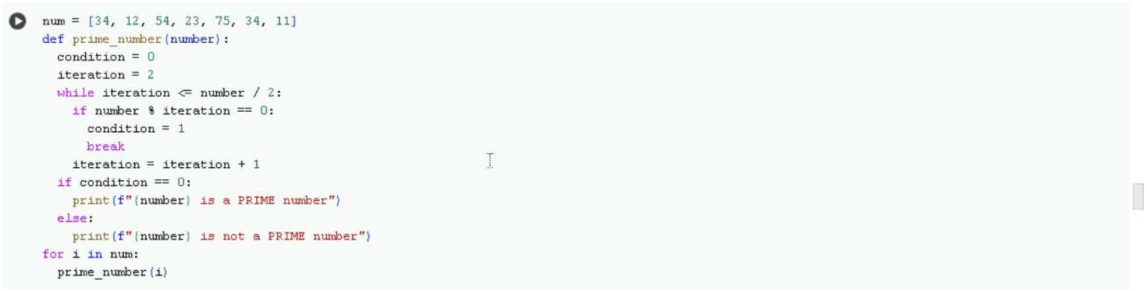
```
[31] num = 15
     summation = 0
     c = 1
     while c <= num:
       summation = c**2 + summation
       c = c + 1
     print("The sum of squares is", summation)
```

Output:



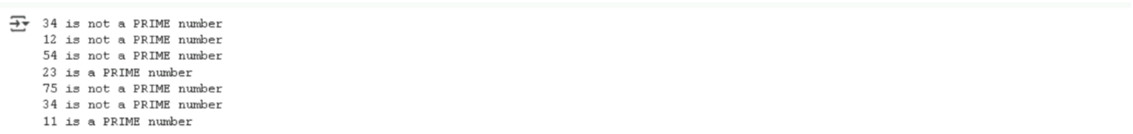
```
The sum of squares is 1240
```

38. Code examples of while loops in Python for a number is Prime number or not.



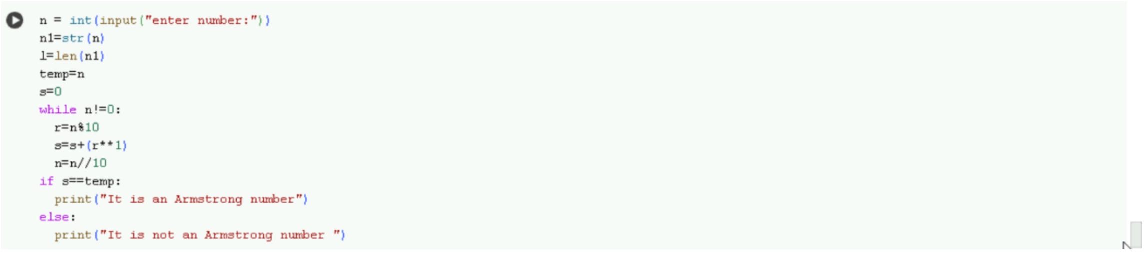
```
num = [34, 12, 54, 23, 75, 34, 11]
def prime_number(number):
  condition = 0
  iteration = 2
  while iteration <= number / 2:
    if number % iteration == 0:
      condition = 1
      break
    iteration = iteration + 1
  if condition == 0:
    print(f"{number} is a PRIME number")
  else:
    print(f"{number} is not a PRIME number")
for i in num:
  prime_number(i)
```

Output:



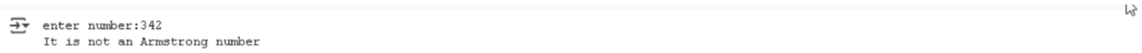
```
34 is not a PRIME number
12 is not a PRIME number
54 is not a PRIME number
23 is a PRIME number
75 is not a PRIME number
34 is not a PRIME number
11 is a PRIME number
```

39. Program using a while loop to verify whether the given integer is an Armstrong number.



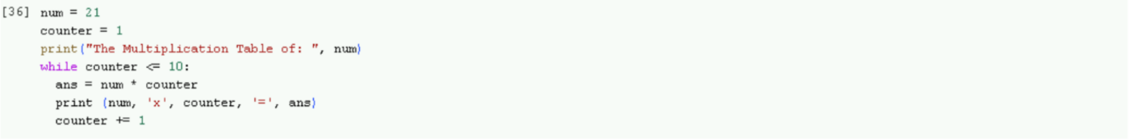
```
n = int(input("enter number:"))
n1=str(n)
l=len(n1)
temp=n
s=0
while n!=0:
  r=n%10
  s=s+(r**1)
  n=n//10
if s==temp:
  print("It is an Armstrong number")
else:
  print("It is not an Armstrong number ")
```

Output:



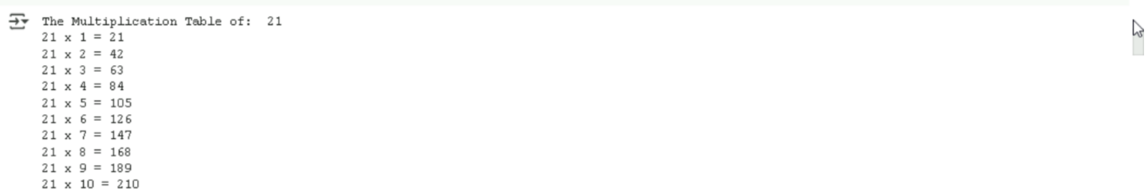
```
enter number:342
It is not an Armstrong number
```

40. U se the while loop for printing the multiplication table of a given number.



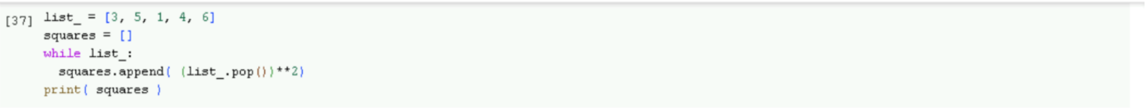
```
[36] num = 21
     counter = 1
     print("The Multiplication Table of: ", num)
     while counter <= 10:
       ans = num * counter
       print (num, 'x', counter, '=', ans)
       counter += 1
```

Output:



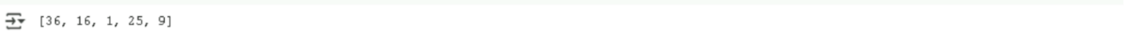
```
The Multiplication Table of:  21
21 x 1 = 21
21 x 2 = 42
21 x 3 = 63
21 x 4 = 84
21 x 5 = 105
21 x 6 = 126
21 x 7 = 147
21 x 8 = 168
21 x 9 = 189
21 x 10 = 210
```

41.Code examples of while loops in Python for square every number of a list.



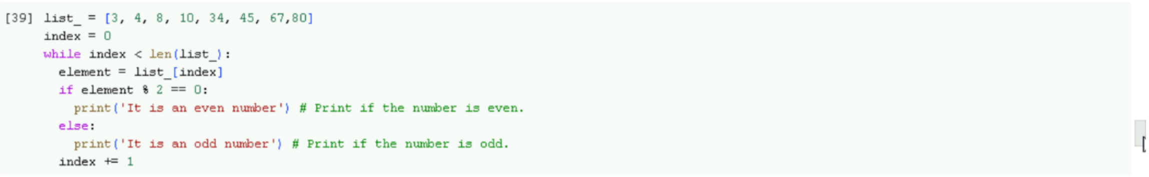
```
[37] list_ = [3, 5, 1, 4, 6]
     squares = []
     while list_:
       squares.append( (list_.pop())**2)
     print( squares )
```

Output:



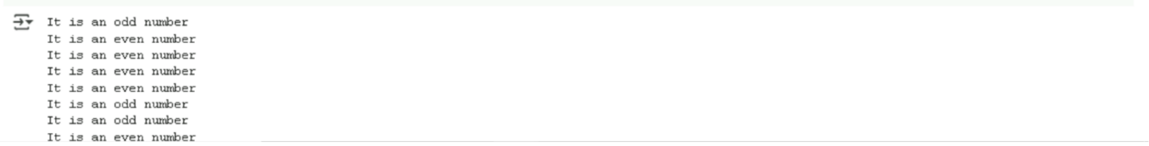
```
[36, 16, 1, 25, 9]
```

42. code examples of while loops in Python for determine odd and even number from every number of a list.



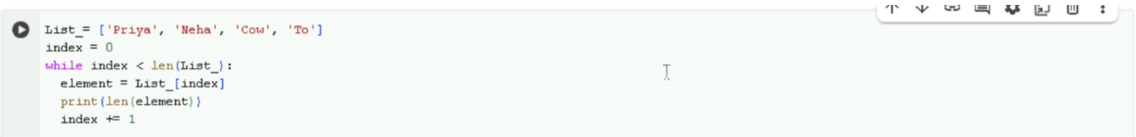
```
[39] list_ = [3, 4, 8, 10, 34, 45, 67,80]
     index = 0
     while index < len(list_):
       element = list_[index]
       if element % 2 == 0:
         print('It is an even number') # Print if the number is even.
       else:
         print('It is an odd number') # Print if the number is odd.
       index += 1
```

Output:



```
It is an odd number
It is an even number
It is an even number
It is an even number
It is an even number
It is an odd number
It is an odd number
It is an even number
```

43. code examples of while loops in Python for determine the number letters of every word from the given list.

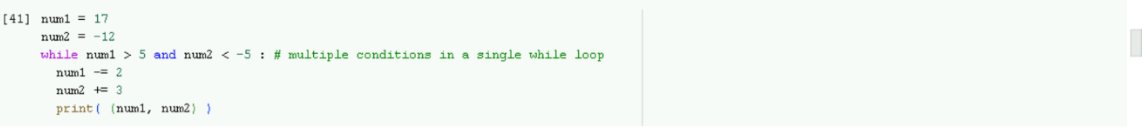


```
List_= ['Priya', 'Neha', 'Cow', 'To']
index = 0
while index < len(List_):
  element = List_[index]
  print(len(element))
  index += 1
```

Output:

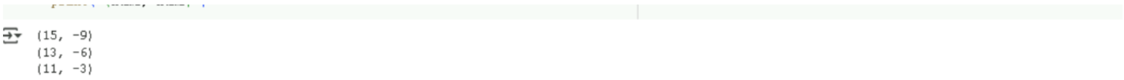
```
5
4
3
2
```

44. code examples of while loops in Python for multiple condition.



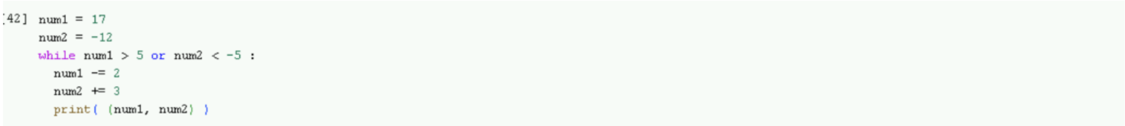
```
[41] num1 = 17
     num2 = -12
     while num1 > 5 and num2 < -5 : # multiple conditions in a single while loop
       num1 -= 2
       num2 += 3
       print( (num1, num2) )
```

Output:



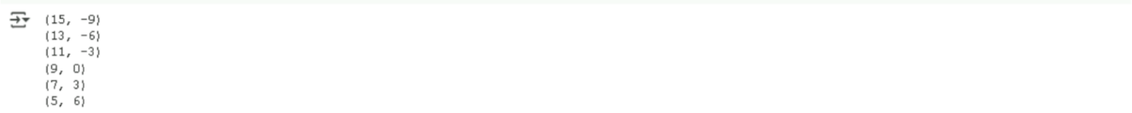
```
(15, -9)
(13, -6)
(11, -3)
```

45. another example of multiple conditions with an OR operator.



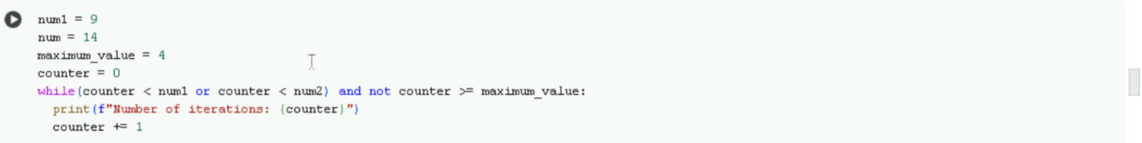
```
[42] num1 = 17
     num2 = -12
     while num1 > 5 or num2 < -5 :
       num1 -= 2
       num2 += 3
       print( (num1, num2) )
```

Output:



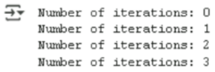
```
(15, -9)
(13, -6)
(11, -3)
(9, 0)
(7, 3)
(5, 6)
```

46.



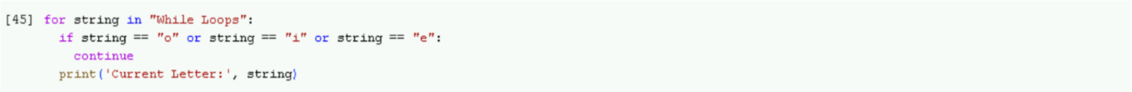
```
num1 = 9
num = 14
maximum_value = 4
counter = 0
while(counter < num1 or counter < num2) and not counter >= maximum_value:
  print(f"Number of iterations: {counter}")
  counter += 1
```

Output:



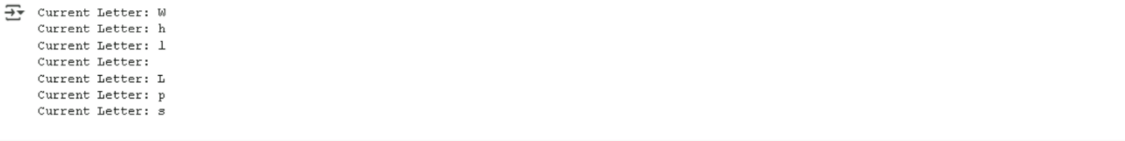
```
Number of iterations: 0
Number of iterations: 1
Number of iterations: 2
Number of iterations: 3
```

47. Continue Statement



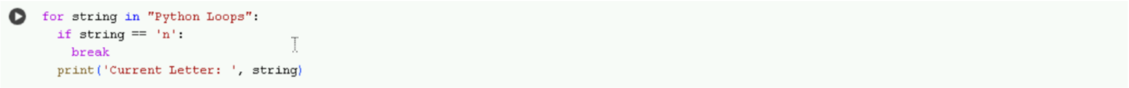
```
[45] for string in "While Loops":
       if string == "o" or string == "i" or string == "e":
         continue
       print('Current Letter:', string)
```

Output:



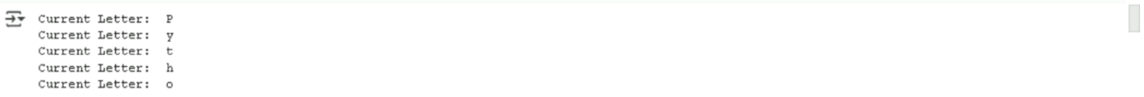
```
Current Letter: W
Current Letter: h
Current Letter: l
Current Letter: 
Current Letter: L
Current Letter: p
Current Letter: s
```

48. Break Statement



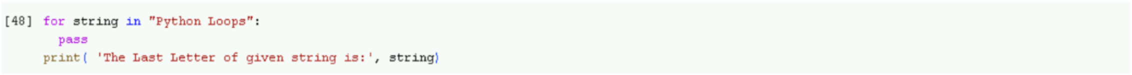
```
for string in "Python Loops":
  if string == 'n':
    break
  print('Current Letter: ', string)
```

Output:



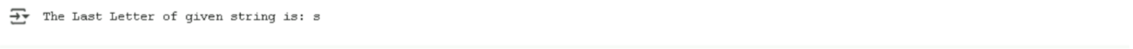
```
Current Letter:  P
Current Letter:  y
Current Letter:  t
Current Letter:  h
Current Letter:  o
```

49. Pass Statement



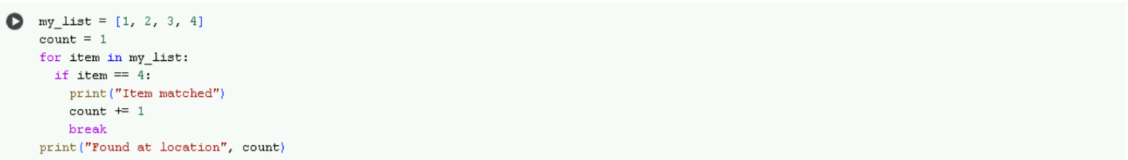
```
[48] for string in "Python Loops":
       pass
     print( 'The Last Letter of given string is:', string)
```

Output:



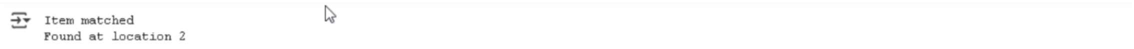
```
The Last Letter of given string is: s
```

50. break statement with for loop



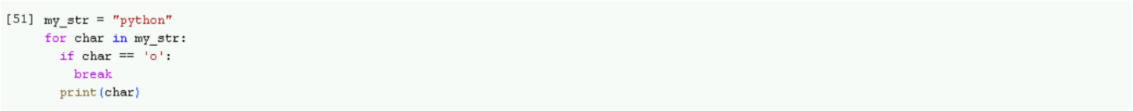
```
my_list = [1, 2, 3, 4]
count = 1
for item in my_list:
  if item == 4:
    print("Item matched")
    count += 1
    break
print("Found at location", count)
```

Output:



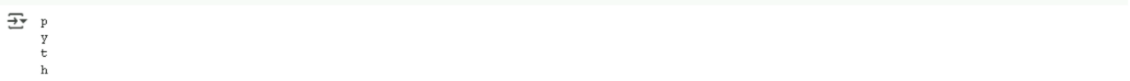
```
Item matched
Found at location 2
```

51. Breaking out of a loop early



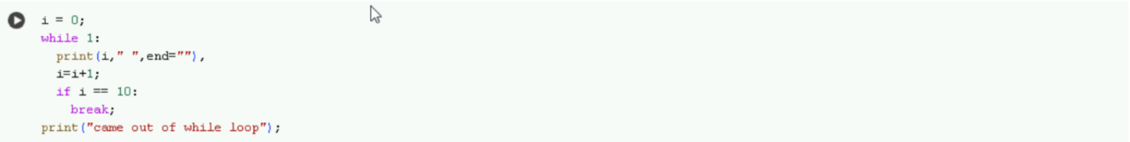
```
[51] my_str = "python"
     for char in my_str:
       if char == 'o':
         break
       print(char)
```

Output:



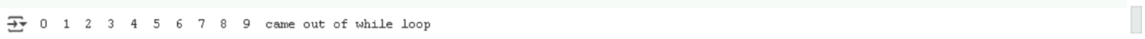
```
p
y
t
h
```

52. break statement with while loop



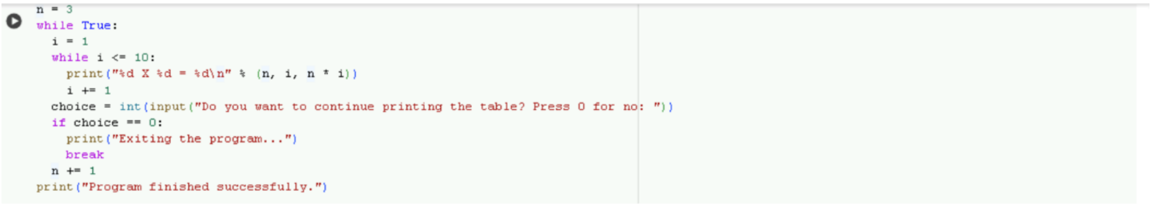
```
i = 0;
while 1:
  print(i," ",end=""),
  i=i+1;
  if i == 10:
    break;
print("came out of while loop");
```

Output:



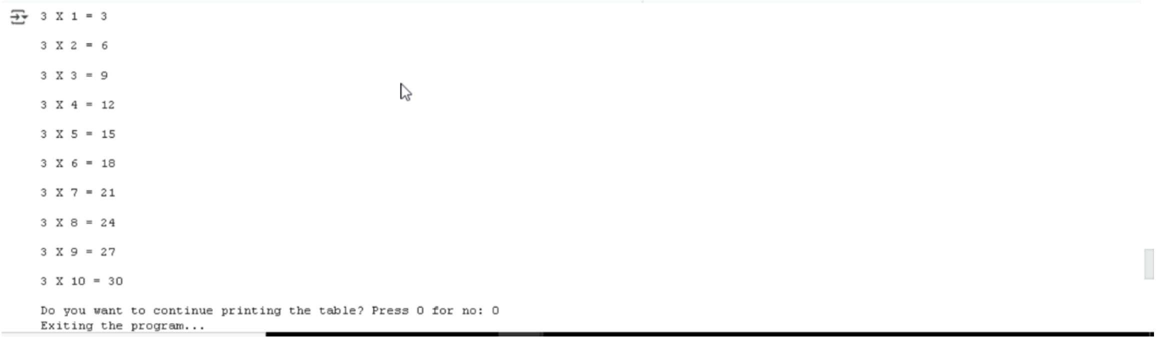
```
0  1  2  3  4  5  6  7  8  9  came out of while loop
```

53. break statement with nested loop



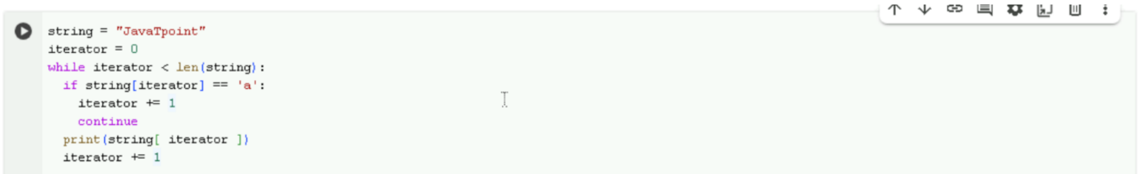
```
n = 3
while True:
  i = 1
  while i <= 10:
    print("%d X %d = %d\n" % (n, i, n * i))
    i += 1
  choice = int(input("Do you want to continue printing the table? Press 0 for no: "))
  if choice == 0:
    print("Exiting the program...")
    break
  n += 1
print("Program finished successfully.")
```

Output:



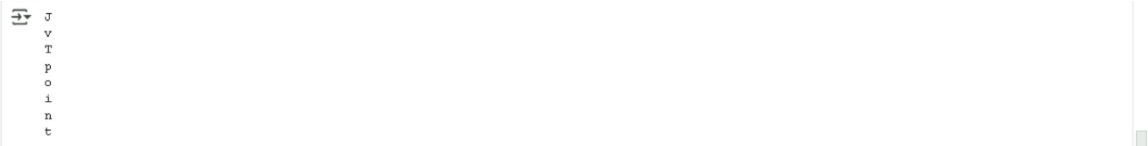
```
3 X 1 = 3
3 X 2 = 6
3 X 3 = 9
3 X 4 = 12
3 X 5 = 15
3 X 6 = 18
3 X 7 = 21
3 X 8 = 24
3 X 9 = 27
3 X 10 = 30
Do you want to continue printing the table? Press 0 for no: 0
Exiting the program...
```

54. Python Continue Statements in while Loop



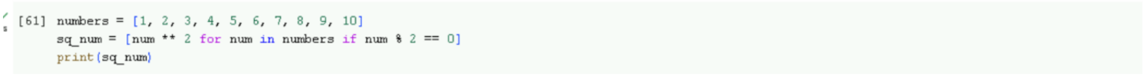
```
string = "JavaTpoint"
iterator = 0
while iterator < len(string):
  if string[iterator] == 'a':
    iterator += 1
    continue
  print(string[ iterator ])
  iterator += 1
```

Output:



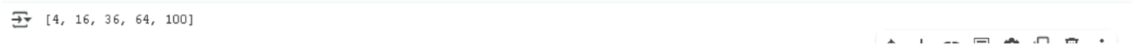
```
J
v
T
p
o
i
n
t
```

55. Python Continue statement in list comprehension



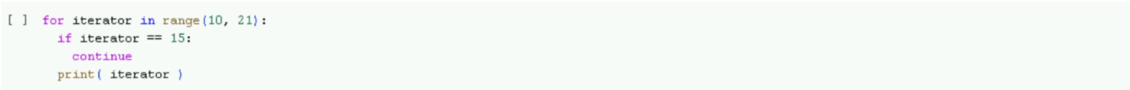
```
[61] numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
     sq_num = [num ** 2 for num in numbers if num % 2 == 0]
     print(sq_num)
```

Output:



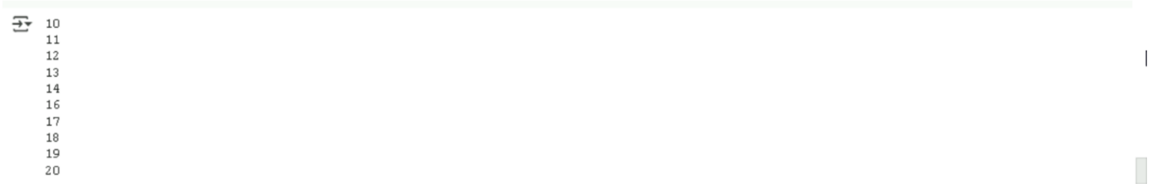
```
[4, 16, 36, 64, 100]
```

56. Printing numbers from 10 to 20 except 15 can be done using continue statement and for loop.



```
[ ]  for iterator in range(10, 21):
       if iterator == 15:
         continue
       print( iterator )
```

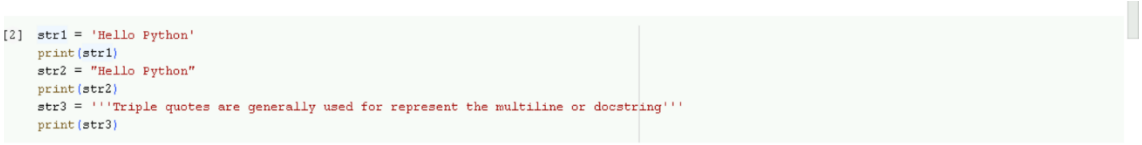
Output:



```
10
11
12
13
14
16
17
18
19
20
```

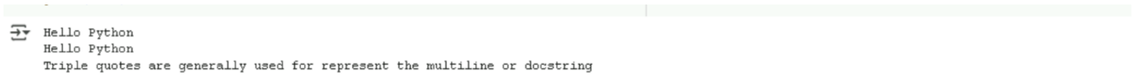
Python String

57. Creating String in Python



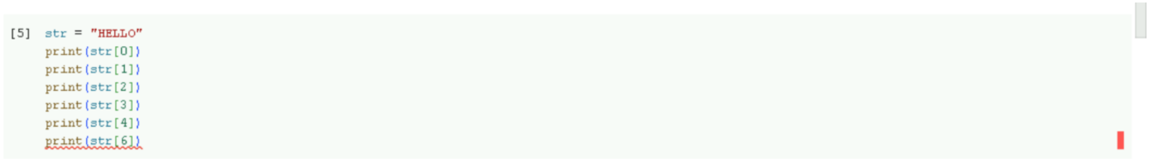
```
[2]  str1 = 'Hello Python'
     print(str1)
     str2 = "Hello Python"
     print(str2)
     str3 = '''Triple quotes are generally used for represent the multiline or docstring'''
     print(str3)
```

Output:



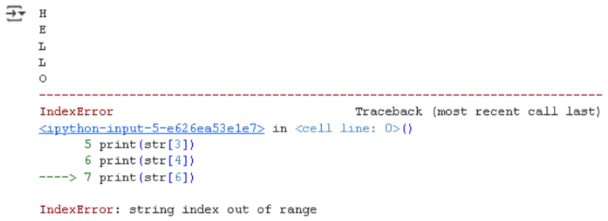
```
Hello Python
Hello Python
Triple quotes are generally used for represent the multiline or docstring
```

58. Strings indexing and splitting



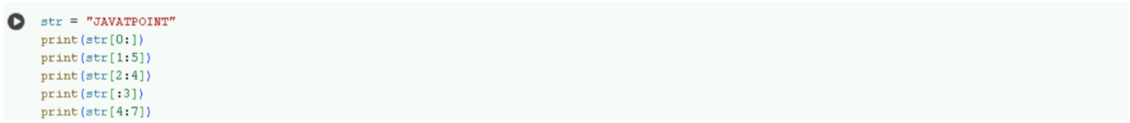
```
[5]  str = "HELLO"
     print(str[0])
     print(str[1])
     print(str[2])
     print(str[3])
     print(str[4])
     print(str[6])
```

Output:



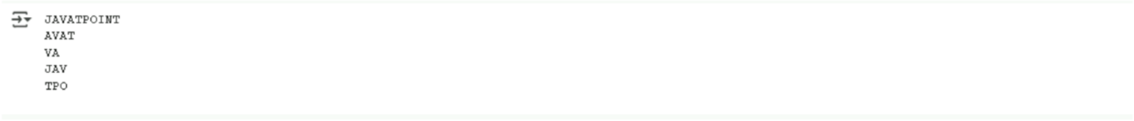
```
H
E
L
L
O
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-5-e626ea53e1e7> in <cell line: 0>()
      5 print(str[3])
      6 print(str[4])
----> 7 print(str[6])

IndexError: string index out of range
```

59. Strings indexing and splitting



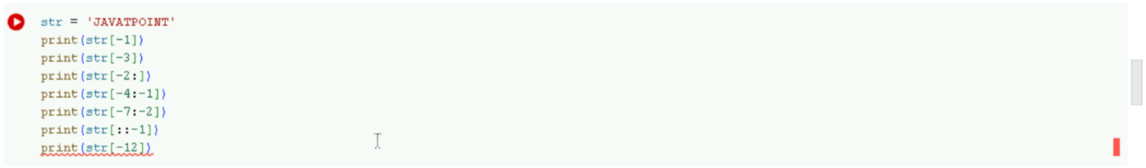
```
str = "JAVATPOINT"
print(str[0:])
print(str[1:5])
print(str[2:4])
print(str[:3])
print(str[4:7])
```

Output:



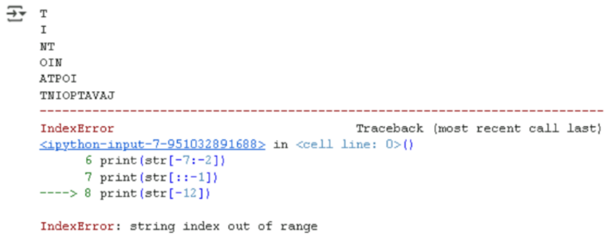
```
JAVATPOINT
AVAT
VA
JAV
TPO
```

60. Strings indexing and splitting



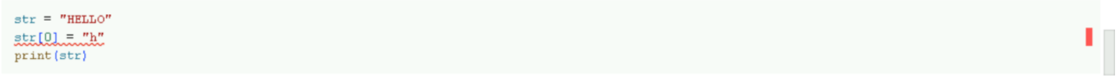
```
str = 'JAVATPOINT'
print(str[-1])
print(str[-3])
print(str[-2:])
print(str[-4:-1])
print(str[-7:-2])
print(str[::-1])
print(str[-12])
```

Output:



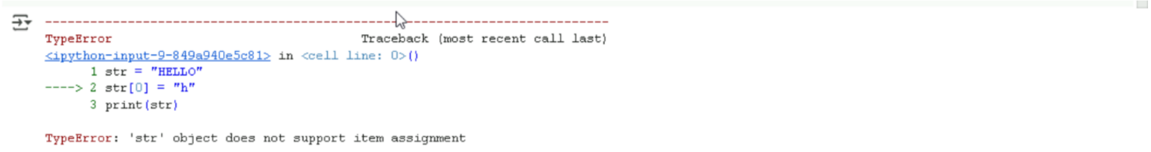
```
T
I
NT
OIN
ATPOI
TNIOPTAVAJ
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-7-951032891688> in <cell line: 0>()
      6 print(str[-7:-2])
      7 print(str[::-1])
----> 8 print(str[-12])

IndexError: string index out of range
```

61. Reassigning Strings



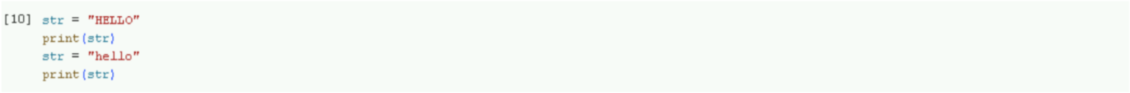
```
str = "HELLO"
str[0] = "h"
print(str)
```

Output:



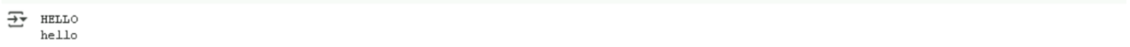
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-9-849a940e5c81> in <cell line: 0>()
      1 str = "HELLO"
----> 2 str[0] = "h"
      3 print(str)

TypeError: 'str' object does not support item assignment
```

62. Reassigning Strings



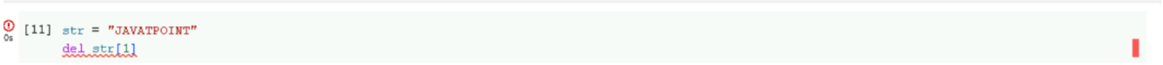
```
[10] str = "HELLO"
     print(str)
     str = "hello"
     print(str)
```

Output:



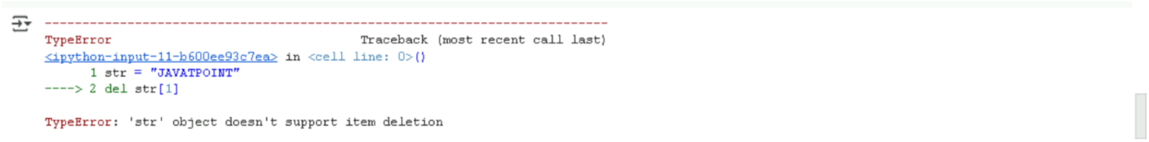
```
HELLO
hello
```

63. Deleting the String



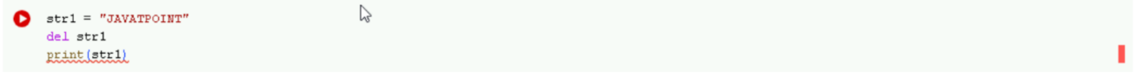
```
[11] str = "JAVATPOINT"
     del str[1]
```

Output:



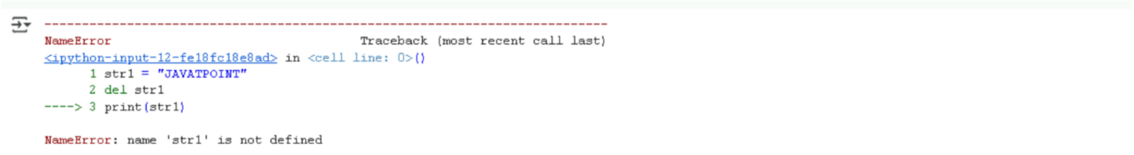
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-11-b600ee93c7ea> in <cell line: 0>()
      1 str = "JAVATPOINT"
----> 2 del str[1]

TypeError: 'str' object doesn't support item deletion
```

64. Deleting the String



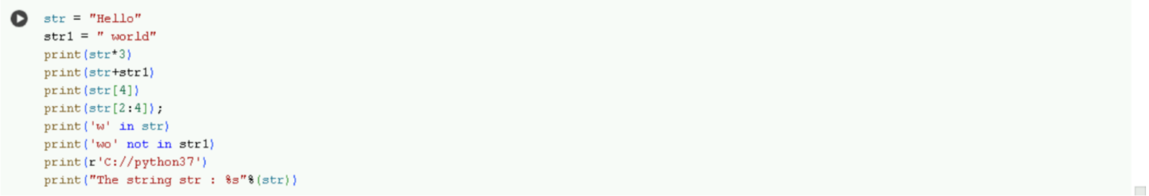
```
str1 = "JAVATPOINT"
del str1
print(str1)
```

Output:



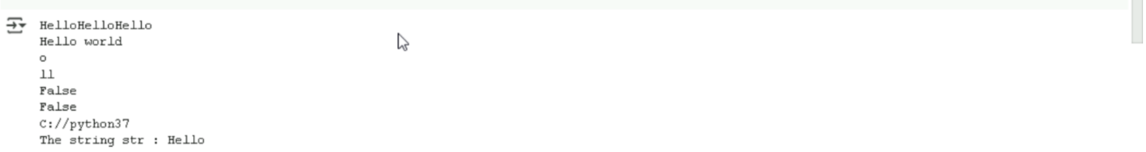
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-12-fe18fc18e8ad> in <cell line: 0>()
      1 str1 = "JAVATPOINT"
      2 del str1
----> 3 print(str1)

NameError: name 'str1' is not defined
```

65. String Operators



```
str = "Hello"
str1 = " world"
print(str*3)
print(str+str1)
print(str[4])
print(str[2:4]);
print('w' in str)
print('wo' not in str1)
print(r'C://python37')
print("The string str : %s"%(str))
```

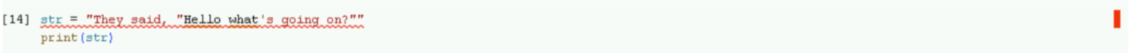
Output:



```
HelloHelloHello
Hello world
o
ll
False
False
C://python37
The string str : Hello
```

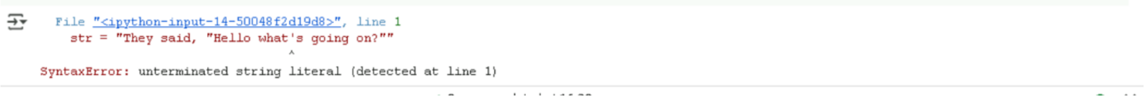
Python String Formatting

66. Escape Sequence



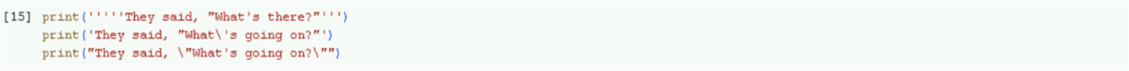
```
[14] str = "They said, "Hello what's going on?""
     print(str)
```

Output:



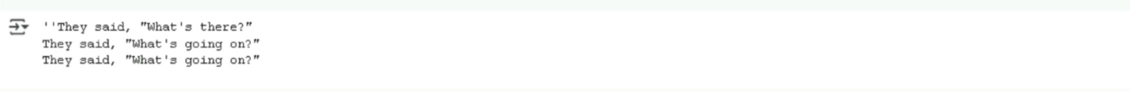
```
  File "<ipython-input-14-50048f2d19d8>", line 1
    str = "They said, "Hello what's going on?""
                                 ^
SyntaxError: unterminated string literal (detected at line 1)
```

67. Escape Sequence



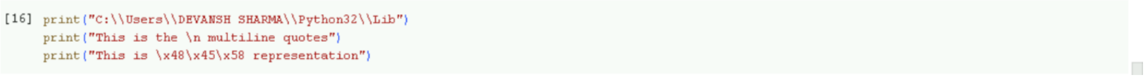
```
[15] print('''''They said, "What's there?"''')
     print('They said, "What\'s going on?"')
     print("They said, \"What's going on?\"")
```

Output:



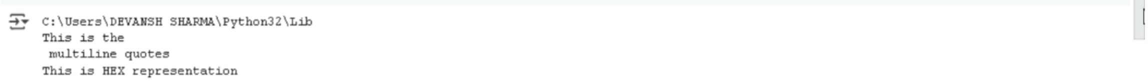
```
''They said, "What's there?"
They said, "What's going on?"
They said, "What's going on?"
```

68. Escape Sequence



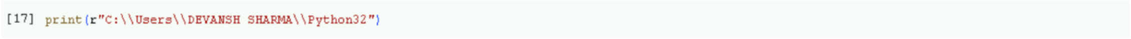
```
[16] print("C:\\Users\\DEVANSH SHARMA\\Python32\\Lib")
     print("This is the \n multiline quotes")
     print("This is \x48\x45\x58 representation")
```

Output:



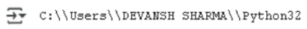
```
C:\Users\DEVANSH SHARMA\Python32\Lib
This is the
 multiline quotes
This is HEX representation
```

69. Escape Sequence



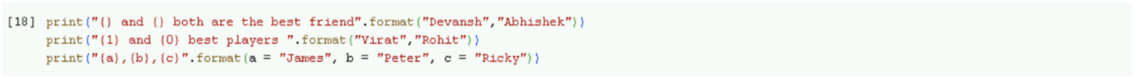
```
[17] print(r"C:\\Users\\DEVANSH SHARMA\\Python32")
```

Output:



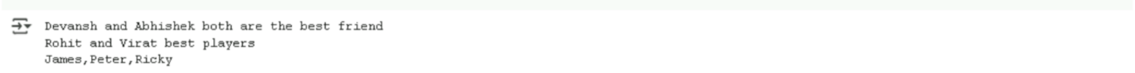
```
C:\\Users\\DEVANSH SHARMA\\Python32
```

70. format() method



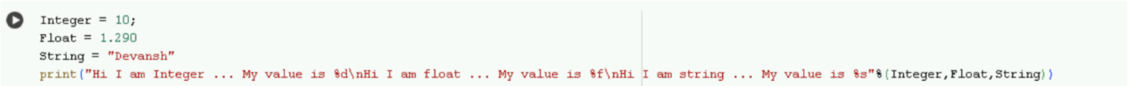
```
[18] print("{} and {} both are the best friend".format("Devansh","Abhishek"))
     print("{1} and {0} best players ".format("Virat","Rohit"))
     print("{a},{b},{c}".format(a = "James", b = "Peter", c = "Ricky"))
```

Output:



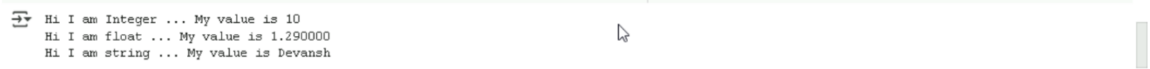
```
Devansh and Abhishek both are the best friend
Rohit and Virat best players
James,Peter,Ricky
```

71. String Formatting Using % Operator



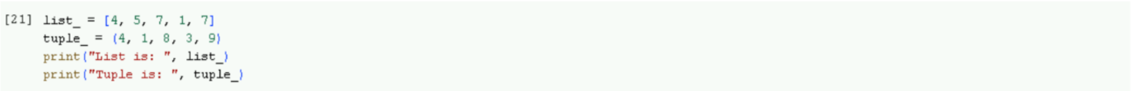
```
Integer = 10;
Float = 1.290
String = "Devansh"
print("Hi I am Integer ... My value is %d\nHi I am float ... My value is %f\nHi I am string ... My value is %s"%(Integer,Float,String))
```

Output:



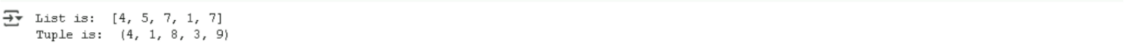
```
Hi I am Integer ... My value is 10
Hi I am float ... My value is 1.290000
Hi I am string ... My value is Devansh
```

72. Python code to show the difference between creating a list and a tuple.



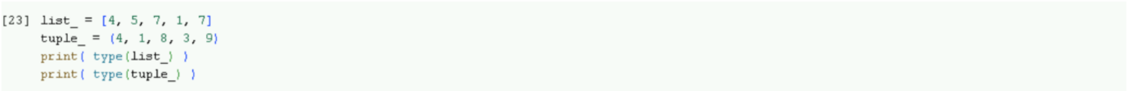
```
[21] list_ = [4, 5, 7, 1, 7]
     tuple_ = (4, 1, 8, 3, 9)
     print("List is: ", list_)
     print("Tuple is: ", tuple_)
```

Output:



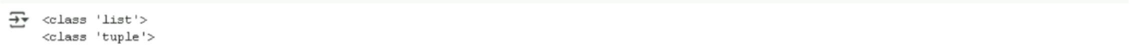
```
List is:  [4, 5, 7, 1, 7]
Tuple is:  (4, 1, 8, 3, 9)
```

73. Code to print the data type of the data structure using the type() function.



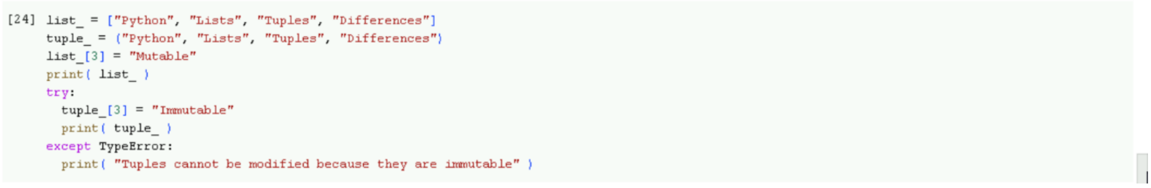
```
[23] list_ = [4, 5, 7, 1, 7]
     tuple_ = (4, 1, 8, 3, 9)
     print( type(list_) )
     print( type(tuple_) )
```

Output:



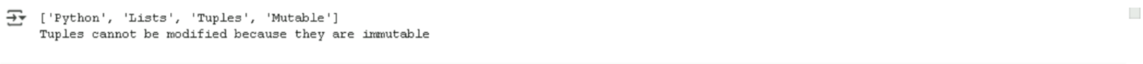
```
<class 'list'>
<class 'tuple'>
```

74. Updating the element of list and tuple at a particular index.



```
[24] list_ = ["Python", "Lists", "Tuples", "Differences"]
     tuple_ = ("Python", "Lists", "Tuples", "Differences")
     list_[3] = "Mutable"
     print( list_ )
     try:
       tuple_[3] = "Immutable"
       print( tuple_ )
     except TypeError:
       print( "Tuples cannot be modified because they are immutable" )
```

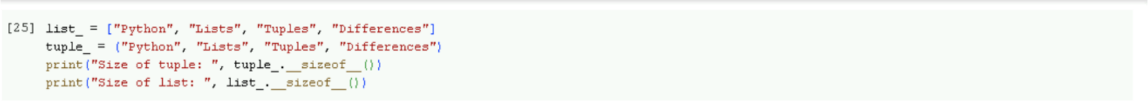
Output:



```
['Python', 'Lists', 'Tuples', 'Mutable']
Tuples cannot be modified because they are immutable
```

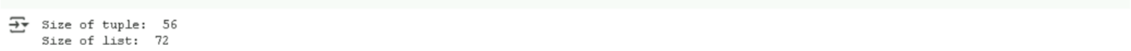
75.

75. Code to show the difference in the size of a list and a tuple



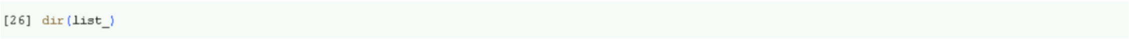
```
[25] list_ = ["Python", "Lists", "Tuples", "Differences"]
     tuple_ = ("Python", "Lists", "Tuples", "Differences")
     print("Size of tuple: ", tuple_.__sizeof__())
     print("Size of list: ", list_.__sizeof__())
```

Output:



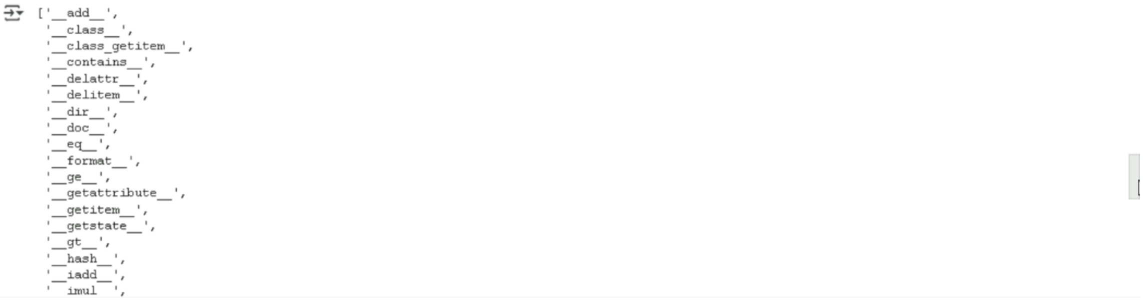
```
Size of tuple:  56
Size of list:  72
```

76. Printing directory of list.



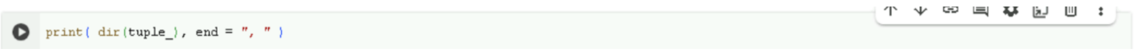
```
[26] dir(list_)
```

Output:



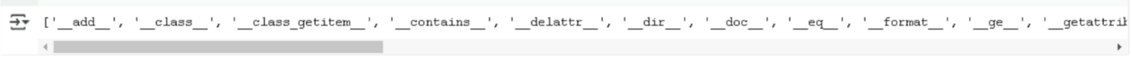
```
['__add__',
 '__class__',
 '__class_getitem__',
 '__contains__',
 '__delattr__',
 '__delitem__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__getstate__',
 '__gt__',
 '__hash__',
 '__iadd__',
 '__imul__',
```

77. Printing directory of a tuple.



```
print( dir(tuple_), end = ", " )
```

Output:



```
['__add__', '__class__', '__class_getitem__', '__contains__', '__delattr__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattrik
```