K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Science and Humanities

Batch: P1-3	**Roll No.:16014022098**
Experiment / assignment / tutorial No.	
Grade: AA / AB / BB / BC / CC / CD /DD	
Signature of the Staff In-charge with date	

TITLE: Decision Making Statements

AIM: 1) Write a program to count the number of prime numbers and composite numbers entered by the user.
2) Write a program to check whether a given number is Armstrong or not.

Expected OUTCOME of Experiment: Use different Decision Making statements in Python.

Resource Needed: Python IDE

Theory:

Decision Control Statements

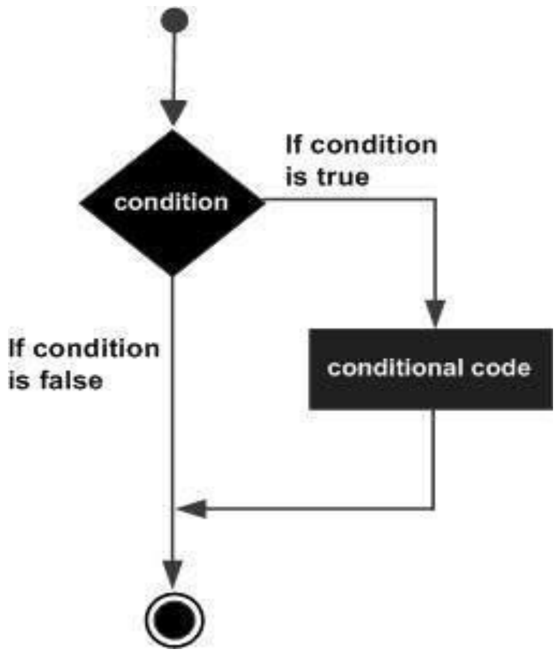
1) Selection/Conditional branching statements
- a) if statement
- b) if-else statement
- c) if-elif-else statement

2)Basic loop Structures/Iterative statement
- a) while loop
- b) for loop

If statement:
In Python **if** statement is used for decision-making operations. It contains a body of code which runs only when the condition given in the **if** statement is true.

Syntax:
```
if condition:
    statement(s)
```

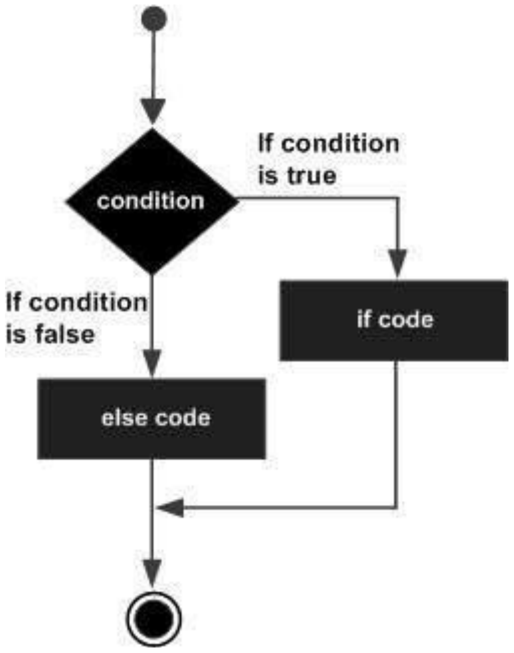
If flowchart:



If-else Statement:
An **else** statement can be combined with an **if** statement. An **else** statement contains the block of code that executes if the conditional expression in the **if** statement resolves to 0 or a FALSE value.
The **else** statement is an optional statement and there could be at most only one **else** statement following **if**.

Syntax:
```
if expression:
   statement(s)
else:
   statement(s)
```

If-else flowchart:



If-elif-else Statement:

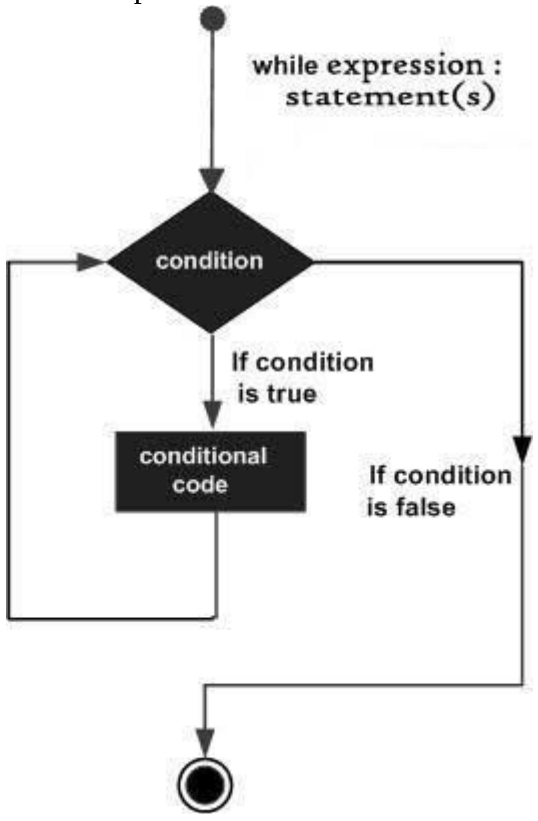
The **elif** statement allows you to check multiple expressions for TRUE and execute a block of code as soon as one of the conditions evaluates to TRUE.
Similar to the else, the **elif** statement is optional. However, unlike **else**, for which there can be at most one statement, there can be an arbitrary number of **elif** statements following an **if.**

Syntax:
```
if expression1:
   statement(s)
elif expression2:
   statement(s)
elif expression3:
   statement(s)
else:
   statement(s)
```

While loop:
A **while** loop statement in Python programming language repeatedly executes a target statement as long as a given condition is true.

Syntax:
```
while expression:
   statement(s)
```

While loop flowchart:

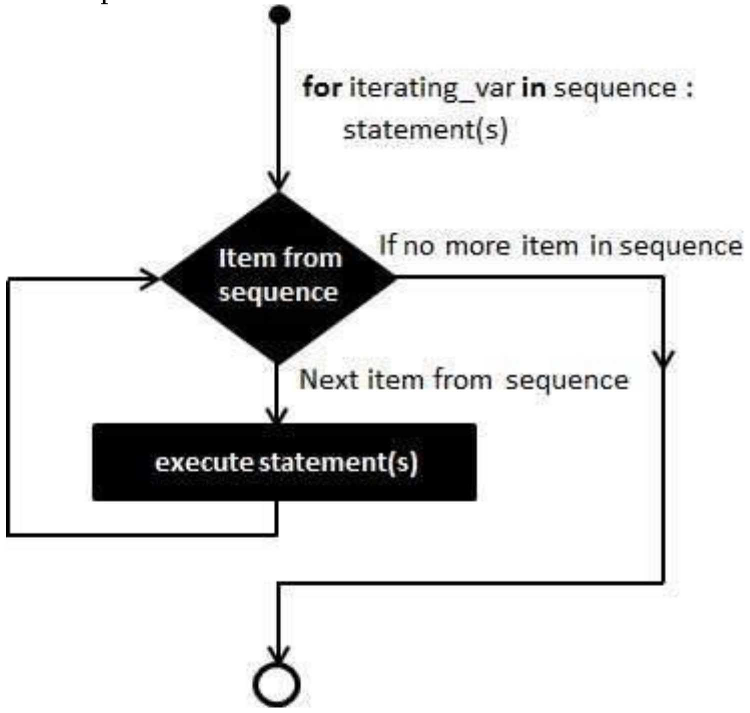


For Loop:
The **for** statement in Python differs a bit from what you may be used to in C. Rather than giving the user the ability to define both the iteration step and halting condition (as C), Python's **for** statement iterates over the items of any sequence (a list or a string), in the order that they appear in the sequence.

Syntax:

```
for iterating_var in sequence:
   statements(s)
```

For loop flowchart:



Problem Definition:

1) Write a program to read the numbers until -1 is encountered. Also, count the number of prime numbers and composite numbers entered by the user

2) Write a program to check whether a number is Armstrong or not.
(Armstrong number is a number that is equal to the sum of cubes of its digits for example: 153 = 1^3 + 5^3 + 3^3.)

Books/ Journals/ Websites referred:

1. Reema Thareja, *Python Programming: Using Problem Solving Approach*, Oxford University Press, First Edition 2017, India
2. Sheetal Taneja and Naveen Kumar, *Python Programming: A modular Approach*, Pearson India, Second Edition 2018,India
3. https://docs.python.org/3/tutorial/controlflow.html#for-statements

---

Implementation details:

Q1

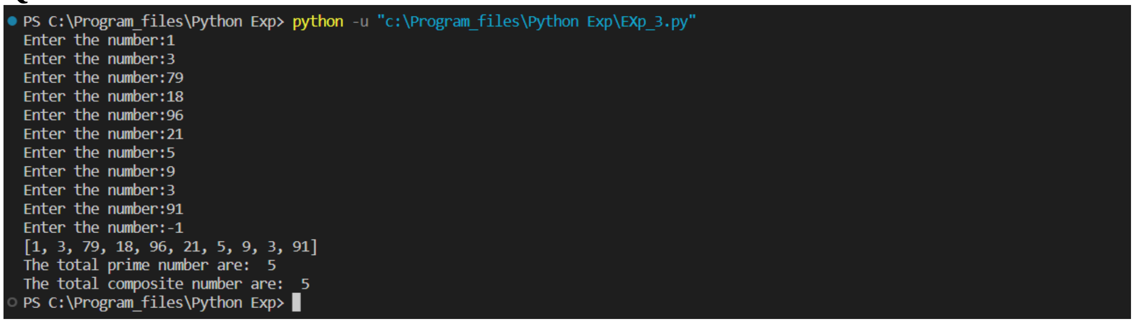
```python
lst = []
composite = 0
prime = 0
while True:
    n = int(input("Enter the number:"))
    if n == -1:
        break
    lst.append(n)
length = len(lst)
for i in range(length):
    if lst[i] == 2:
        prime += 1
    elif lst[i] == 3:
        prime += 1
    elif lst[i] == 5:
        prime += 1
    elif lst[i] == 7:
        prime += 1
    elif lst[i] % 2 == 0:
        composite += 1
    elif lst[i] % 3 == 0:
        composite += 1
    elif lst[i] % 5 == 0:
        composite += 1
    elif lst[i] % 7 == 0:
        composite += 1
    elif lst[i] % 11 == 0:
        composite += 1
    else:
        prime += 1
print(lst)
print("The total prime number are: ", prime)
print("The total composite number are: ", composite)
```

Q2

```
l=(input("Enter number:"))
lenght=len(l)
sum=0
for i in l:
    sum=sum+int(i)**3
if(int(l)==sum):
    print("number is armstrong")
else:
    print("numbaer is not armstrong")
```

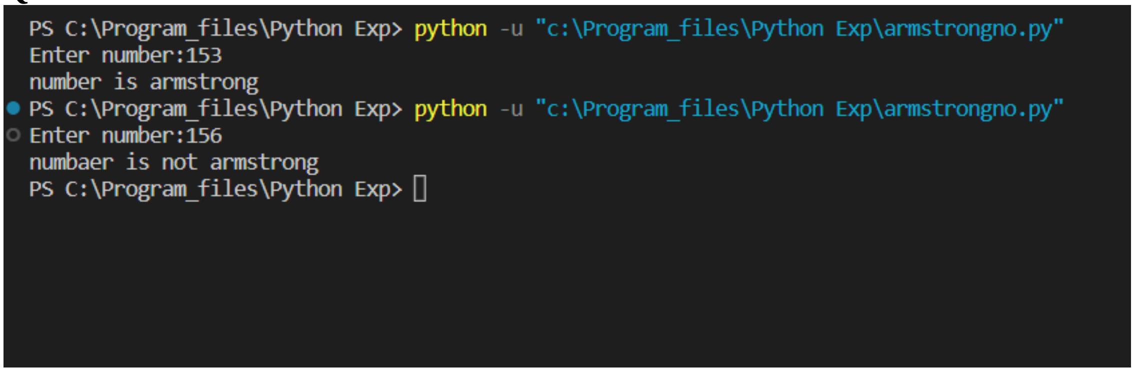
Output(s):

Q1



```
PS C:\Program_files\Python Exp> python -u "c:\Program_files\Python Exp\EXp_3.py"
Enter the number:1
Enter the number:3
Enter the number:79
Enter the number:18
Enter the number:96
Enter the number:21
Enter the number:5
Enter the number:9
Enter the number:3
Enter the number:91
Enter the number:-1
[1, 3, 79, 18, 96, 21, 5, 9, 3, 91]
The total prime number are:  5
The total composite number are:  5
PS C:\Program_files\Python Exp>
```

Q2



```
PS C:\Program_files\Python Exp> python -u "c:\Program_files\Python Exp\armstrongno.py"
Enter number:153
number is armstrong
PS C:\Program_files\Python Exp> python -u "c:\Program_files\Python Exp\armstrongno.py"
Enter number:156
numbaer is not armstrong
PS C:\Program_files\Python Exp>
```

Conclusion:
Nested if else were executed and use of conditional statement wit loops were understood

Post Lab Questions:

1) When should we use nested if statements? Illustrate your answer with the help of an example.

   Ans. When we are given if more than one condition that needs to be checked and all conditions returning different values. In a nested if statement, we can have an if…elif…else statement inside another if…elif…else statement

```
num=15
if(num>0):
    if(num==0)
          print(“zero”)
     else:
          print(“positive number”)
else:
print(“negative number”)

output:

positive number
```

2) Explain the utility of break and continue statements with the help of an example.

Ans: Break statement in Python is used to bring the control out of the loop when some external condition is triggered. Break statement is put inside the loop body (generally after if condition).

<u>Syntax:</u>

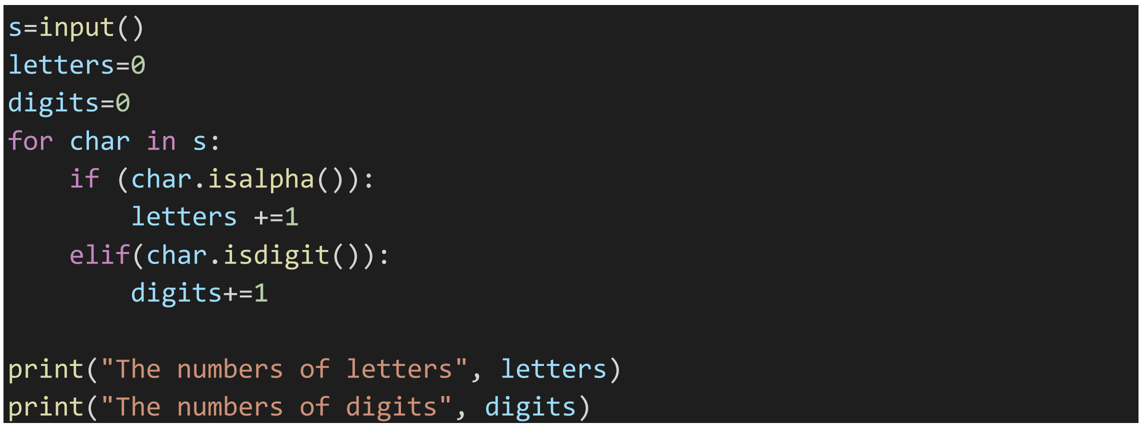
```
Break
```

Example:

```
for i in range(10)
        if(i==10):
           break
```

3) Write a program that accepts a string from user and calculate the number of digits and letters in string.

ans



```
s=input()
letters=0
digits=0
for char in s:
    if (char.isalpha()):
        letters +=1
    elif(char.isdigit()):
        digits+=1

print("The numbers of letters", letters)
print("The numbers of digits", digits)
```

Date: _____________ **Signature of faculty in-charge**