



K. J. Somaiya Polytechnic, Mumbai-77

Batch No: C1

Enrollment No: FCOW19118

Experiment No.: 04

Experiment Name: Develop programs to understand the control structures of python (Minimum 4 programs on decision making and looping).

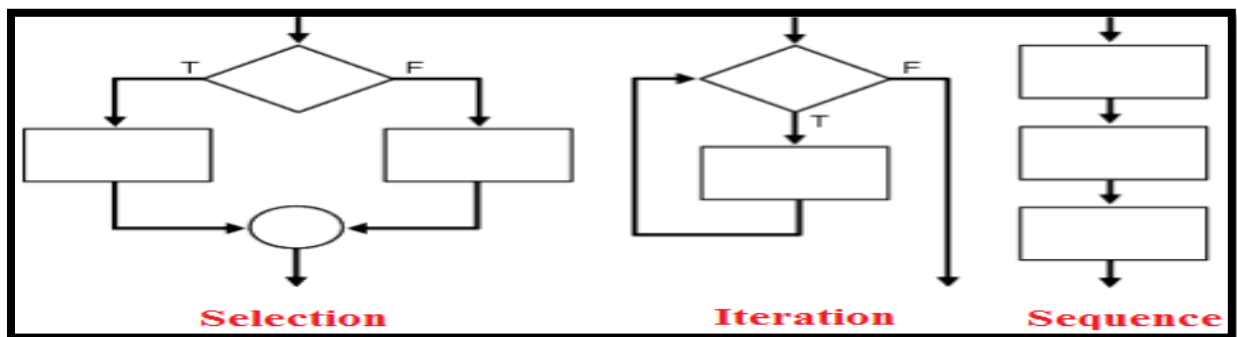
Experiment No: 04

Experiment Name: Develop programs to understand the control structures of python (Minimum 4 programs on decision making and looping).

Course Outcome:**O18RA72.2****Built small programs for decision making statements.****Theory:****Python Control Structures**

According to the structure theorem, any computer program can be written using the basic control structures. A control structure (or flow of control) is a block of programming that analyses variables and chooses a direction in which to go based on given parameters. In simple sentence, a control structure is just a decision that the computer makes. So, it is the basic decision-making process in programming and flow of control determines how a computer program will respond when given certain conditions and parameters.

There are two basic aspects of computer programming: data and instructions. To work with data, you need to understand variables and data types; to work with instructions, you need to understand control structures and statements. Flow of control through any given program is implemented with three basic types of control structures: Sequential, Selection and Repetition.



1. Sequential

Sequential execution is when statements are executed one after another in order. You don't need to do anything more for this to happen.

2. Selection

Selection used for decisions, branching - choosing between 2 or more alternative paths. Conditional Statement in Python perform different computations or actions depending on whether a specific Boolean constraint evaluates to true or false. Conditional statements are handled by IF statements in Python.

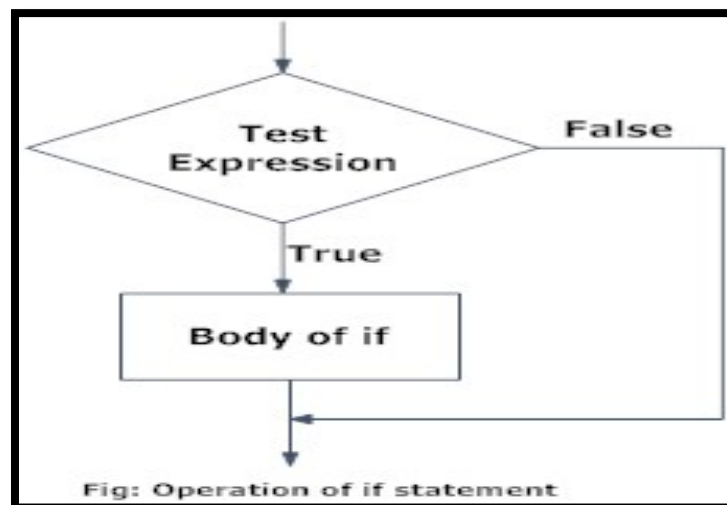
a. if statement

Syntax:

if test expression:

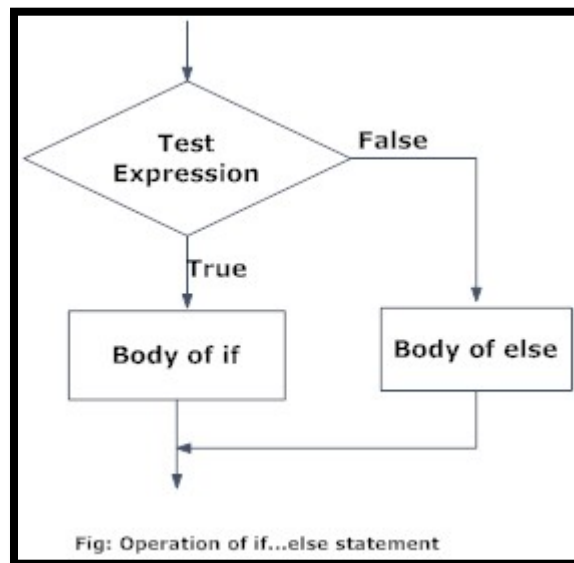
statement(s)

- Here, the program evaluates the test expression and will execute statement(s) only if the test expression is True.
- If the test expression is False, the statement(s) is not executed.
- In Python, the body of the if statement is indicated by the indentation. The body starts with an indentation and the first unindented line marks the end.
- Python interprets non-zero values as True. None and 0 are interpreted as False.

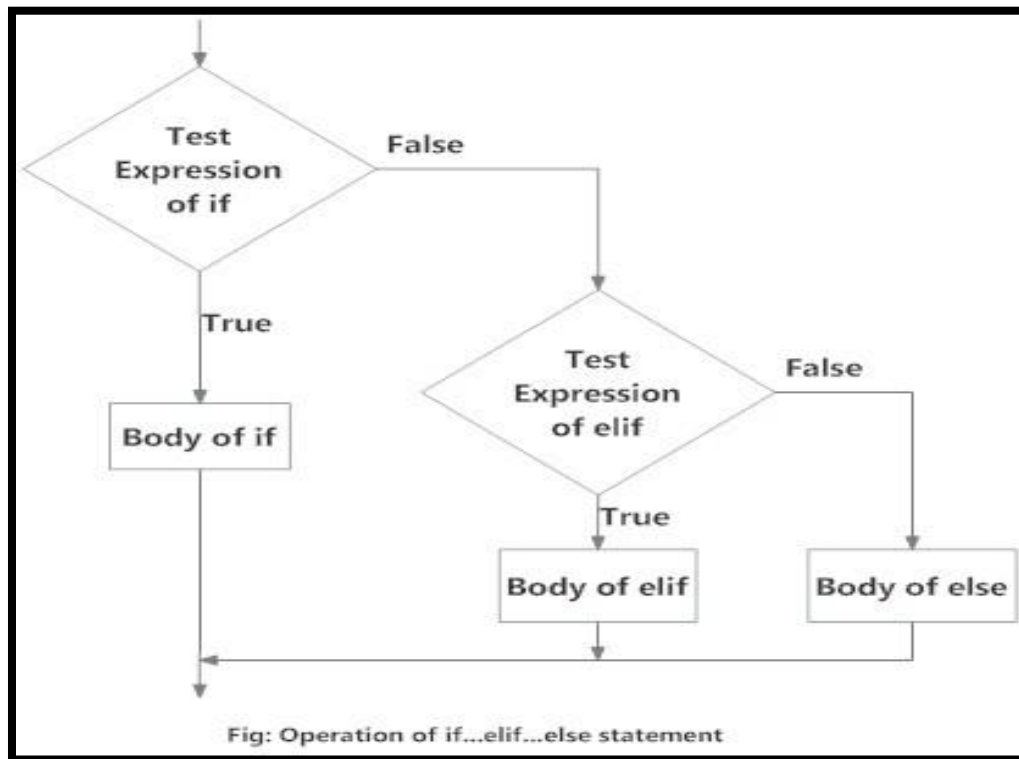


b. if-else statement**Syntax:****if test expression:****Body of if****else:****Body of else**

The if..else statement evaluates test expression and will execute the body of if only when the test condition is True. If the condition is False, the body of else is executed. Indentation is used to separate the blocks.

Flowchart:**c. if-elif-else statement****Syntax:****if test expression:****Body of if****elif test expression:****Body of elif****else:****Body of else**

- The elif is short for else if. It allows us to check for multiple expressions.
- If the condition for if is False, it checks the condition of the next elif block and so on.
- If all the conditions are False, the body of else is executed.
- Only one block among the several if...elif...else blocks is executed according to the condition.
- The if block can have only one else block. But it can have multiple elif blocks.

Flowchart:**d. Python Nested if statements**

- We can have a if...elif...else statement inside another if...elif...else statement. This is called nesting in computer programming.
- Any number of these statements can be nested inside one another. Indentation is the only way to figure out the level of nesting. They can get confusing, so they must be avoided unless necessary.

e. Switch statement

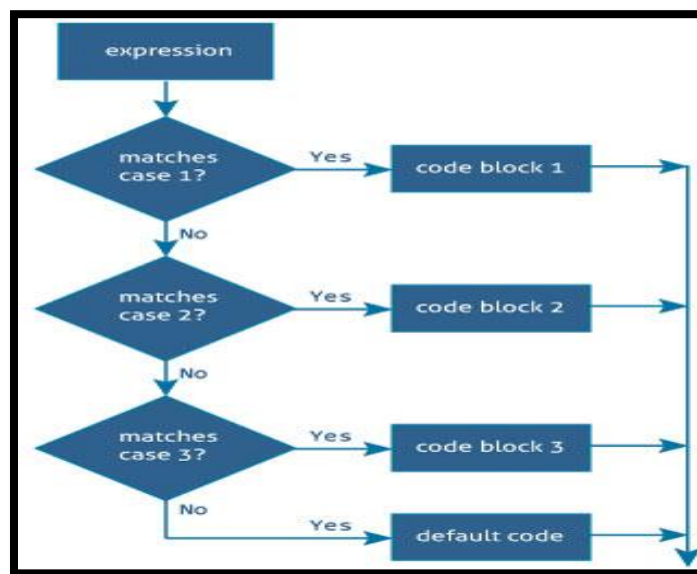
- The switch statement is also a variety of Swift control statement e.g.if-else, guard etc., that performs different actions based on different conditions.
- The beauty of switch statement is, it can compare a value with several possible matching patterns. Therefore, it can be used as a substitute for long if..else..if ladders while matching complex pattern.

Syntax:

```
switch variable/expression {  
  case value1:  
    // statements  
  case value2:  
    // statements  
  default:  
    // statements  
}
```

How Switch Statement in Swift works?

- The switch expression is evaluated once.
- It takes the expression and compares with each case value in the order (Top -> Bottom).
- If there is a match, the statement inside the case are executed and the entire switch statement finishes its execution as soon as the first matching switch case is completed.
- If there is no match for the case, it falls to the next case.
- The default keyword specifies the code to run if there is no case match.

Flowchart:

3. Repetition

Repetition used for looping, i.e. repeating a piece of code multiple times in a row.

a. while loop

In python, while loop is used to execute a block of statements repeatedly until a given a condition is satisfied. And when the condition becomes false, the line immediately after the loop in program is executed. With the while loop we can execute a set of statements as long as a condition is true.

Syntax:

while expression:

statement(s)

In Python, all the statements indented by the same number of character spaces after a programming construct are considered to be part of a single block of code. Python uses indentation as its method of grouping statements.

b. for loop

- A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).
- This is less like the for keyword in other programming languages, and works more like an iterator method as found in other object-orientated programming languages.
- With the for loop we can execute a set of statements, once for each item in a list, tuple, set etc.
- In Python, there is no C style for loop, i.e., for (i=0; i<n; i++). There is “for in” loop which is similar to for each loop in other languages.

Syntax:

for iterator_var in sequence:

statements(s)

Note: It can be used to iterate over iterators and a range.

c. Nested Loops

Python programming language allows to use one loop inside another loop. A nested loop is a loop inside a loop. The "inner loop" will be executed one time for each iteration of the "outer loop":

Syntax:

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

The syntax for a nested while loop statement in Python programming language is as follows :

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

A final note on loop nesting is that we can put any type of loop inside of any other type of loop. For example, a for loop can be inside a while loop or vice versa.

Loop Control Statements

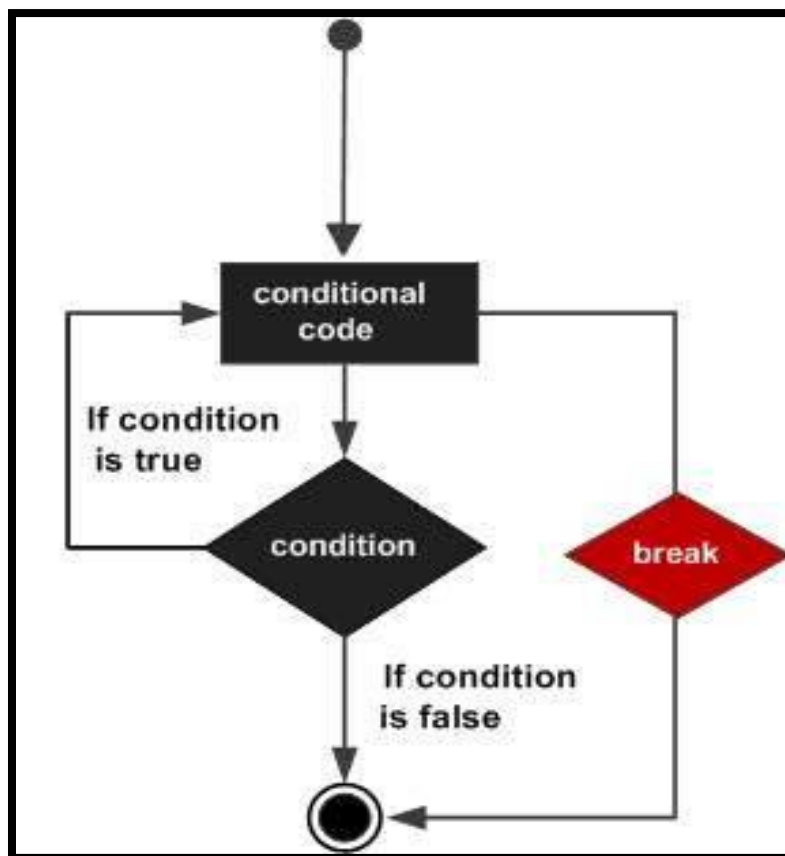
Loop control statements change execution from its normal sequence. When execution leaves a scope, all automatic objects that were created in that scope are destroyed. Python supports the following control statements.

a. continue statement

- It returns the control to the beginning of the loop. With the continue statement we can stop the current iteration of the loop, and continue with the next:
- The continue statement in Python returns the control to the beginning of the while loop. The continue statement rejects all the remaining statements in the current iteration of the loop and moves the control back to the top of the loop.
- The continue statement can be used in both while and for loops.

Syntax: continue**b. break statement**

- It brings control out of the loop.
- It terminates the current loop and resumes execution at the next statement, just like the traditional break statement in C.
- The most common use for break is when some external condition is triggered requiring a hasty exit from a loop. The break statement can be used in both while and for loops.
- If you are using nested loops, the break statement stops the execution of the innermost loop and start executing the next line of code after the block.

Syntax:**break****Flowchart:**

c. pass statement

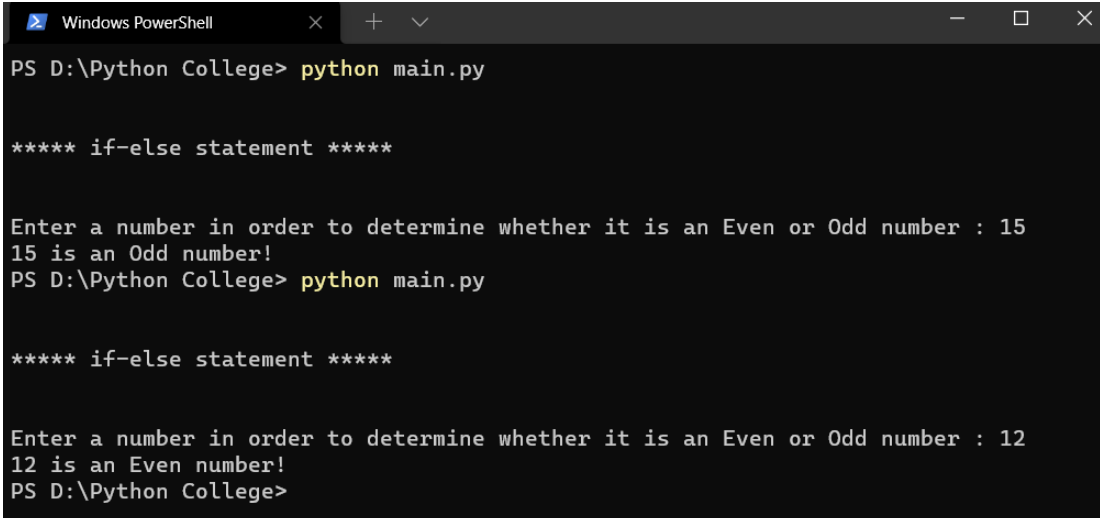
- We use pass statement to write empty loops. Pass is also used for empty control statement, function and classes.
- The pass statement in Python is used when a statement is required syntactically but you do not want any command or code to execute.
- The pass statement is a null operation; nothing happens when it executes. The pass is also useful in places where your code will eventually go, but has not been written yet (e.g., in stubs for example):

Syntax:

Pass

Implementation and Output**1. if-else statement****Program:**

```
print("\n")
print("***** if-else statement *****")
print("\n")
number=int(input("Enter a number in order to determine whether it is an Even or
Odd number : "))
if (number%2)==0:
    print("%d is an Even number!"%(number,))
else:
    print("%d is an Odd number!"%(number,))
```

Output:

```
Windows PowerShell
PS D:\Python College> python main.py

***** if-else statement *****

Enter a number in order to determine whether it is an Even or Odd number : 15
15 is an Odd number!
PS D:\Python College> python main.py

***** if-else statement *****

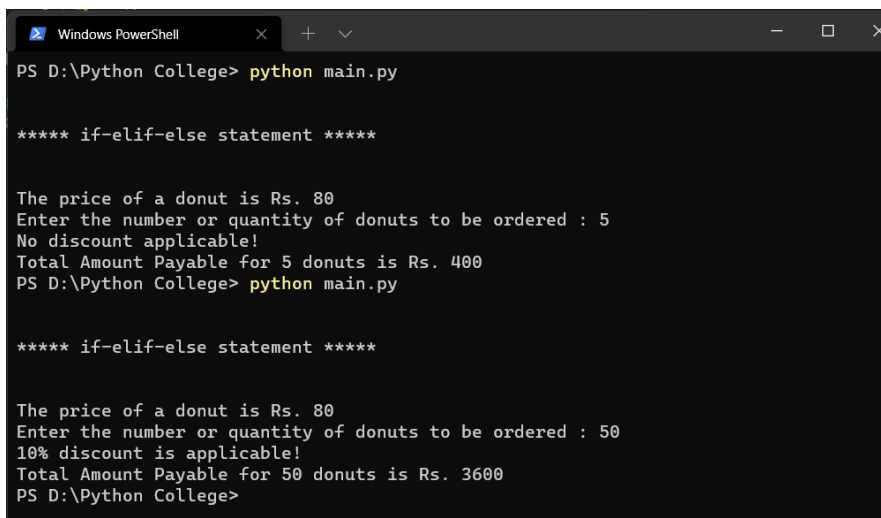
Enter a number in order to determine whether it is an Even or Odd number : 12
12 is an Even number!
PS D:\Python College>
```

2. if-elif-else statement

Program:

```
print("\n")
print("***** if-elif-else statement *****")
print("\n")
price=80
print("The price of a donut is Rs. 80")
quantity=int(input("Enter the number or quantity of donuts to be ordered : "))
amount=price*quantity
if (amount>3000):
    print("10% discount is applicable!")
    discount=amount*10/100
    amount=amount-discount
elif (amount>2000):
    print("5% discount is applicable!")
    discount=amount*5/100
    amount=amount-discount
elif (amount>1000):
    print("2% discount is applicable!")
    discount=amount*2/100
    amount=amount-discount
elif (amount>500):
    print("1% discount is applicable!")
    discount=amount*1/100
    amount=amount-discount
else:
    print("No discount applicable!")
print("Total Amount Payable for %d donuts is Rs. %d"%(quantity,amount))
```

Output:



```
Windows PowerShell
PS D:\Python College> python main.py

***** if-elif-else statement *****

The price of a donut is Rs. 80
Enter the number or quantity of donuts to be ordered : 5
No discount applicable!
Total Amount Payable for 5 donuts is Rs. 400
PS D:\Python College> python main.py

***** if-elif-else statement *****

The price of a donut is Rs. 80
Enter the number or quantity of donuts to be ordered : 50
10% discount is applicable!
Total Amount Payable for 50 donuts is Rs. 3600
PS D:\Python College>
```

```
Windows PowerShell
PS D:\Python College> python main.py
PS D:\Python College> python main.py

***** if-elif-else statement *****

The price of a donut is Rs. 80
Enter the number or quantity of donuts to be ordered : 8
1% discount is applicable!
Total Amount Payable for 8 donuts is Rs. 633
PS D:\Python College> python main.py

***** if-elif-else statement *****

The price of a donut is Rs. 80
Enter the number or quantity of donuts to be ordered : 15
2% discount is applicable!
Total Amount Payable for 15 donuts is Rs. 1176
PS D:\Python College> python main.py

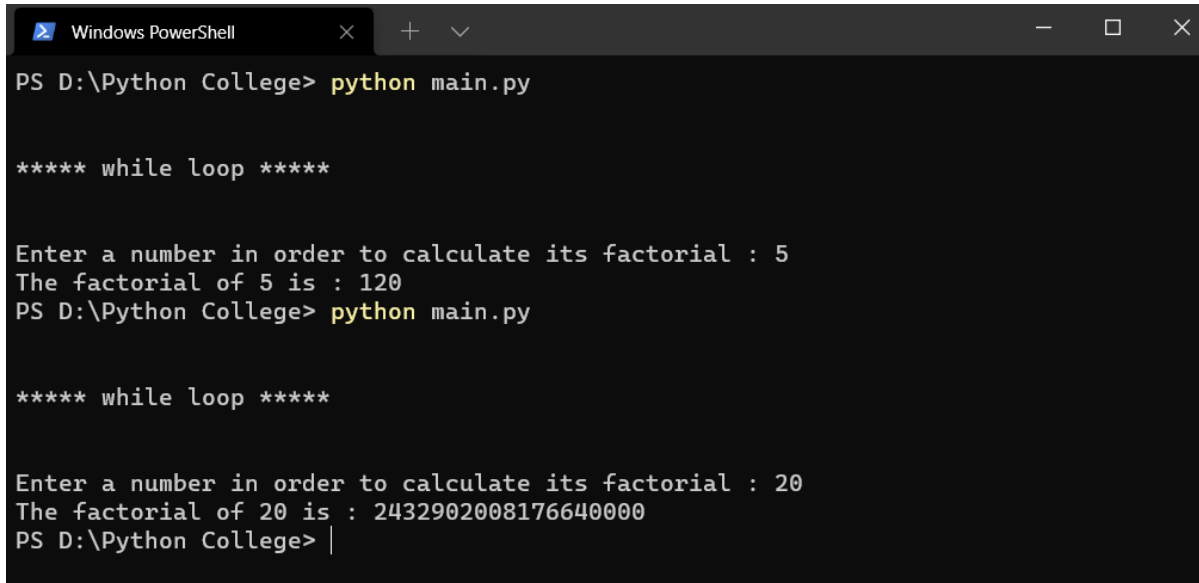
***** if-elif-else statement *****

The price of a donut is Rs. 80
Enter the number or quantity of donuts to be ordered : 30
5% discount is applicable!
Total Amount Payable for 30 donuts is Rs. 2280
PS D:\Python College> |
```

3. while loop

Program:

```
print("\n")
print("***** while loop *****")
print("\n")
i=1
factorial=1
number=int(input("Enter a number in order to calculate its factorial : "))
while i<=number:
    factorial=factorial*i
    i=i+1
print("The factorial of %d is : %d"%(number,factorial))
```

Output:

```
Windows PowerShell
PS D:\Python College> python main.py

***** while loop *****

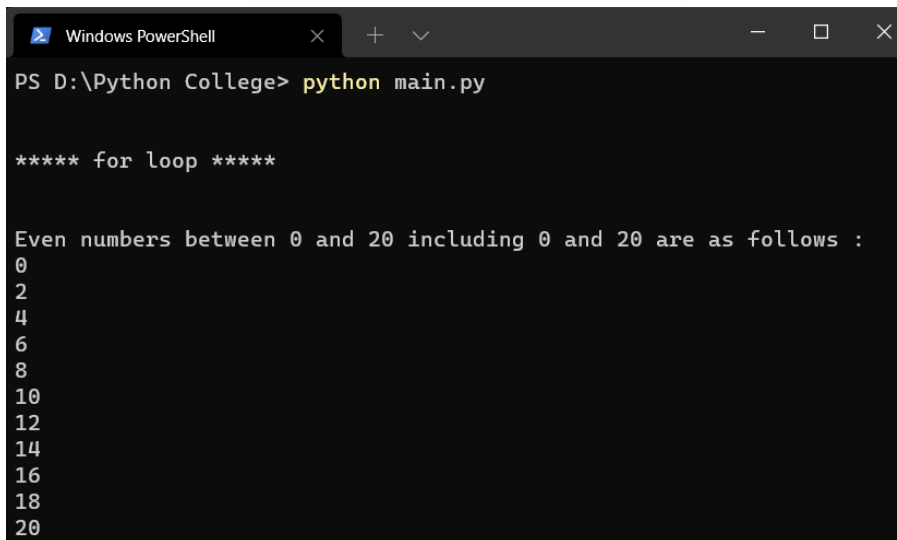
Enter a number in order to calculate its factorial : 5
The factorial of 5 is : 120
PS D:\Python College> python main.py

***** while loop *****

Enter a number in order to calculate its factorial : 20
The factorial of 20 is : 2432902008176640000
PS D:\Python College> |
```

4. for loop**Program:**

```
print("\n")
print("***** for loop *****")
print("\n")
print("Even numbers between 0 and 20 including 0 and 20 are as follows : ")
for even_number in range(0,21,2):
    print(even_number)
```

Output:

```
Windows PowerShell
PS D:\Python College> python main.py

***** for loop *****

Even numbers between 0 and 20 including 0 and 20 are as follows :
0
2
4
6
8
10
12
14
16
18
20
```

Conclusion: Thus, we have developed programs to understand the control structures of python.