	Marwadi University Faculty of Engineering & Technology Department of Information and Communication Technology
Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.
Experiment No: 06	**Date:5/08/25**	**Enrollment No:92510133011**

Aim: Develop programs to understand the control structures of python.

IDE:

What is a control structure in Python?

A control structure in Python is a block of code that determines the flow of execution of a program. Control structures enable programmers to create programs that can make decisions based on certain conditions, repeat code blocks multiple times, or execute different code paths based on the values of variables.
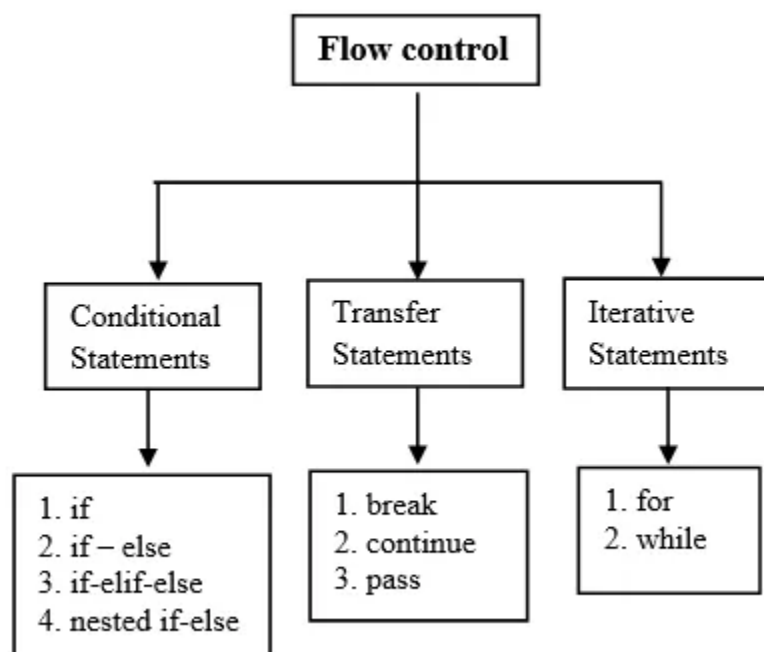
There are several types of control structures in Python:

1. Conditional statements: These structures allow the program to execute different code blocks based on the value of a condition. The if statement is the most common conditional statement in Python, and it can be accompanied by elif and else statements.

2. Loops: These structures allow the program to execute a code block repeatedly based on a condition. Python has two main types of loops: while loops and for loops.

3. Exception handling: These structures allow the program to handle errors and exceptions in a controlled manner, preventing the program from crashing. Python has a try-except structure for handling exceptions.

	Marwadi University
	Faculty of Engineering & Technology
	Department of Information and Communication Technology
Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.
Experiment No: 06	**Date:5/08/25** **Enrollment No:92510133011**

4. Function and method definitions: These structures allow the programmer to define reusable blocks of code that can be called from other parts of the program. Functions and methods can take arguments and return values, and they can also contain other control structures.

By using control structures in Python, programmers can create more complex and powerful programs that can make decisions, repeat actions, and handle errors in a controlled way.

1. Conditional Statements (if, else, elif)

Conditional statements are fundamental to programming and allow us to make decisions based on specific conditions. In Python, we use the keywords if, else, and elif to implement conditional branching. The syntax is clean and straightforward, making Python code highly readable.

1. if Statement

The if statement is used to execute a block of code if a given condition is True. If the condition is False, the code inside the if block is skipped.

Example:

```
x = 10
if x>5:
    print("x is greter than 5")
```

Output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
x is greater than 5
PS C:\Users\trupa\OneDrive\Documents\PWP>
```

2. elif Statement

The elif statement is used when you have multiple conditions to check. It comes after an if statement and before an optional else statement. If the initial if condition is False, Python evaluates the elif condition. You can have multiple elif conditions.

	Marwadi University
	Faculty of Engineering & Technology
	Department of Information and Communication Technology
Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.
Experiment No: 06	**Date:5/08/25**	**Enrollment No:92510133011**

Example:

x = 10
if x>5
   print("x is greater than 5")
elif x ==5:
   print("x is equal to 5")
else:
   print("x is less than 5")

Output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
x is greater than 5
PS C:\Users\trupa\OneDrive\Documents\PWP>
```

3. else Statement
The else statement is used to execute a block of code when the conditions specified in the if and elif statements are not met.

Example

if x>5:
   print("x is greater than 5")
else:
   print("x is not greater than 5")

Output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
x is greater than 5
```

Nested if-else statements
Nested if-else statements in Python allow you to test multiple conditions and execute different code blocks based on the results of these tests.

![Marwadi University logo with NAAC A+]	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**
Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.
Experiment No: 06	**Date:5/08/25**	**Enrollment No:92510133011**

Example
age = 35

if age >= 60:
   print("You are a senior citizen.")
else:
   if age >= 18:
     print("You are an adult.")
   else:
     print("You are a teenager.")

Output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
You are an adult.
```

Example
num = 10

if num > 0:
   if num % 2 == 0:
     print("The number is positive and even.")
   else:
     print("The number is positive but odd.")
else:
   print("The number is not positive.")

Output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
The number is positive and even.
```

Looping Statements

1. for Loop
The for loop is used to iterate over sequences like lists, tuples, strings, and dictionaries. It allows you to perform an action for each item in the sequence.

	Marwadi University
![Marwadi University Logo]	**Faculty of Engineering & Technology**
	Department of Information and Communication Technology

Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.	
Experiment No: 06	Date:5/08/25	Enrollment No:92510133011

Example
Fruits = ["apple", "banana", "cherry"]
for fruit in fruits:
    print(fruit)

Output:

```
NameError: name 'fruits' is not defined. Did you mean: 'Fruits'?
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
apple
banana
cherry
```

2. while Loop
The while loop is used to repeatedly execute a block of code as long as a specified condition is True.

Example

x = 1
while x<=5:
    print(x)
    x+=1

Loop Control Statements

Python provides several loop control statements to enhance the functionality of loops.

break: The break statement is used inside a loop (while loop or for loop). When the condition specified in the if statement is true, the break statement is executed, and the control is transferred to the next statement after the loop. This means that the loop is terminated, and the code execution continues from the statement after the loop.

Example

for x in range(1,6):
        if x==3:
        break
        print(x)
Output:

![Marwadi University Logo]	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**
Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.
Experiment No: 06	**Date:5/08/25**	**Enrollment No:92510133011**

```
IndentationError: expected an indented block after 'if' statement on line 2
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
3
```

continue: The continue statement is used to skip a particular iteration of a loop when a specific condition is met. When a continue statement is executed inside a loop, it skips the current iteration of the loop and jumps to the next iteration.

Example
for x in range(1,6):
        if x==3:
        continue
        print(x)

Output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
1
2
4
5
```

pass: The pass statement is a placeholder in Python. It doesn't do anything but is used when a statement is syntactically required. It is often used as a placeholder for functions, loops, or conditional blocks that will be implemented later.

Example
for x in range(1,6):
    if x == 3:
        pass
    print(x)

Output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
1
2
3
4
5
```

Try and Except Statement – Catching Exceptions
- In Python, you may catch and deal with exceptions by using the try and except commands.

NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**
Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.
Experiment No: 06	**Date:5/08/25**	**Enrollment No:92510133011**

- The try and except clauses are used to contain statements that can raise exceptions and statements that handle such exceptions.

Example

try:

  number = int(input("Enter a number: "))

  result = 10 / number

  print("The result is:", result)

except ZeroDivisionError:

  print("Division by zero is not allowed.")

except ValueError:

  print("Invalid input. Please enter a valid number.")

Python Function:

Python functions are reusable code blocks that carry out particular tasks, helping programmers structure their code and make it easier to read. By preventing duplication, functions make the code more modular and manageable. The 'def' keyword, the function name, and any parameters included in parenthesis define a function. The code to be performed is contained in the function body, and the 'return' statement allows the function to produce a result.

Types of Functions in Python

Python supports various types of functions, each serving different purposes in programming. Here are the main types of functions in Python, along with examples:

1. Built-in Functions

These functions are pre-defined in Python and can be used directly without any further declaration.

Example

my_list = [1, 2, 3, 4, 5]

print(len(my_list))

output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
5
```

2. User-defined Functions

	Marwadi University
![Marwadi University logo NAAC A+]	**Faculty of Engineering & Technology**
	Department of Information and Communication Technology

Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.	
Experiment No: 06	**Date:5/08/25**	**Enrollment No:92510133011**

These are functions that users create to perform specific tasks.

Example
def add_numbers(a, b):
    return a + b
result = add_numbers(3, 5)
print(result)
output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
8
```

3. Anonymous Functions (Lambda Functions)
These are small, unnamed functions defined using the lambda keyword. They are typically used for short, simple operations.
Example
add = lambda x, y: x + y
print(add(3, 5))
Output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
8
```

4. Recursive Functions
These are functions that call themselves within their definition. They help solve problems that can be broken down into smaller, similar problems.

Example
def factorial(n):
    if n == 1:
        return 1
    else:
        return n * factorial(n - 1)
print(factorial(5))
Output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
120
```

	Marwadi University
Marwadi University NAAC A+ Marwadi Chandarana Group	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**
Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.
Experiment No: 06	**Date:5/08/25**	**Enrollment No:92510133011**

5. Higher-Order Functions

These functions can take other functions as arguments or return them as results. Examples include map(), filter(), and reduce().

Example
```
def square(x):
    return x * x
numbers = [1, 2, 3, 4, 5]
squared_numbers = list(map(square, numbers))
print(squared_numbers)
```
output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
[1, 4, 9, 16, 25]
```

6. Generator Functions

These functions yield values one at a time and can produce a sequence of values over time, using the yield keyword.

Example
```
def generate_numbers():
    for i in range(1, 6):
        yield i
for number in generate_numbers():
    print(number)
```
output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
1
2
3
4
5
```

Post Lab Exercise:

a. Write a Python program to print all odd numbers between 1 to 100 using a while loop.
```
num = 1
while num <= 100:
    print(num)
```

	Marwadi University
	Faculty of Engineering & Technology
	Department of Information and Communication Technology
Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.
Experiment No: 06	**Date:5/08/25**	**Enrollment No:92510133011**

num += 2

output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
1
3
5
7
9
11
13
15
17
19
21
23
25
27
29
31
33
35
37
39
41
43
45
47
49
51
53
55
57
59
61
63
65
67
69
71
73
75
77
```

```
79
81
83
85
87
89
91
93
95
97
99
```

b. Write a Python function program to count a number of digits in a number.

	Marwadi University
Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**
Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.
Experiment No: 06	**Date:5/08/25**	**Enrollment No:92510133011**

```python
def count_digits(number):
    if number == 0:
        return 1
    count = 0
    while number != 0:
        number = number // 10
        count += 1
    return count
num = int(input("Enter a number: "))
print("Number of digits:", count_digits(num))
```

output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
Enter a number: 92510133011
Number of digits: 11
PS C:\Users\trupa\OneDrive\Documents\PWP>
```

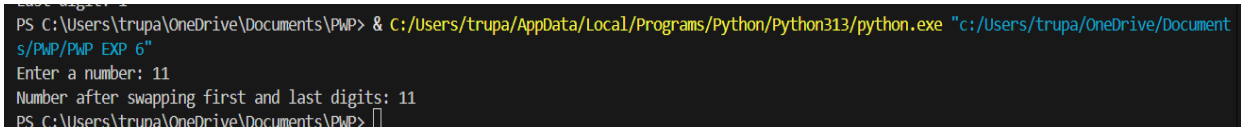c. Write a Python program to find the first and last digits of a number.

```python
def first_last_digits(number):
    last_digit = number % 10
    while number >= 10:
        number = number // 10
    first_digit = number
    return first_digit, last_digit
num = int(input("Enter a number: "))
first, last = first_last_digits(num)
print("First digit:", first)
print("Last digit:", last)
```

output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
Enter a number: 11
First digit: 1
Last digit: 1
```

d. Write a Python program to swap the first and last digits of a number.

```python
def swap_first_last(number):
    num_str = str(number)
    if len(num_str) == 1:
        return number
```

	Marwadi University
	Faculty of Engineering & Technology
	Department of Information and Communication Technology

Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.	
Experiment No: 06	**Date:5/08/25**	**Enrollment No:92510133011**

```
    swapped = num_str[-1] + num_str[1:-1] + num_str[0]
    return int(swapped)
num = int(input("Enter a number: "))
result = swap_first_last(num)
print("Number after swapping first and last digits:", result)
```
output:


```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
Enter a number: 11
Number after swapping first and last digits: 11
PS C:\Users\trupa\OneDrive\Documents\PWP>
```
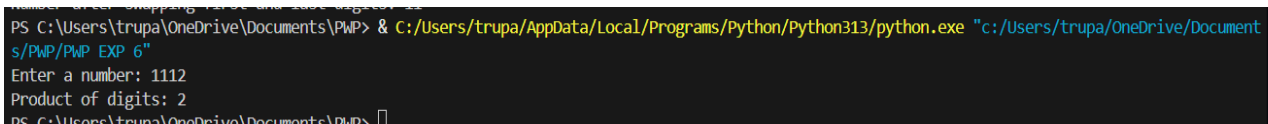
e. Write a Python program to calculate the product of digits of a number.

```
def product_of_digits(number):
    product = 1
    if number == 0:
        return 0
    while number > 0:
        digit = number % 10
        product *= digit
        number = number // 10
    return product
num = int(input("Enter a number: "))
result = product_of_digits(num)
print("Product of digits:", result)
```
output:


```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Document
s/PWP/PWP EXP 6"
Enter a number: 1112
Product of digits: 2
PS C:\Users\trupa\OneDrive\Documents\PWP>
```

f. Write a Python program to enter a number and print its reverse.

```
def reverse_number(number):
    reverse = 0
    while number > 0:
        digit = number % 10
        reverse = reverse * 10 + digit
        number = number // 10
    return reverse
```

	Marwadi University
	Marwadi University
	Faculty of Engineering & Technology
	Department of Information and Communication Technology
Subject: Programming With Python (01CT1309)	**Aim:** Develop programs to understand the control structures of python.
Experiment No: 06	**Date:5/08/25**	**Enrollment No:92510133011**

```
num = int(input("Enter a number: "))
reversed_num = reverse_number(num)
print("Reversed number:", reversed_num)
```

output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> & C:/Users/trupa/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/trupa/OneDrive/Documents/PWP/PWP EXP 6"
Enter a number: 1234
Reversed number: 4321
```