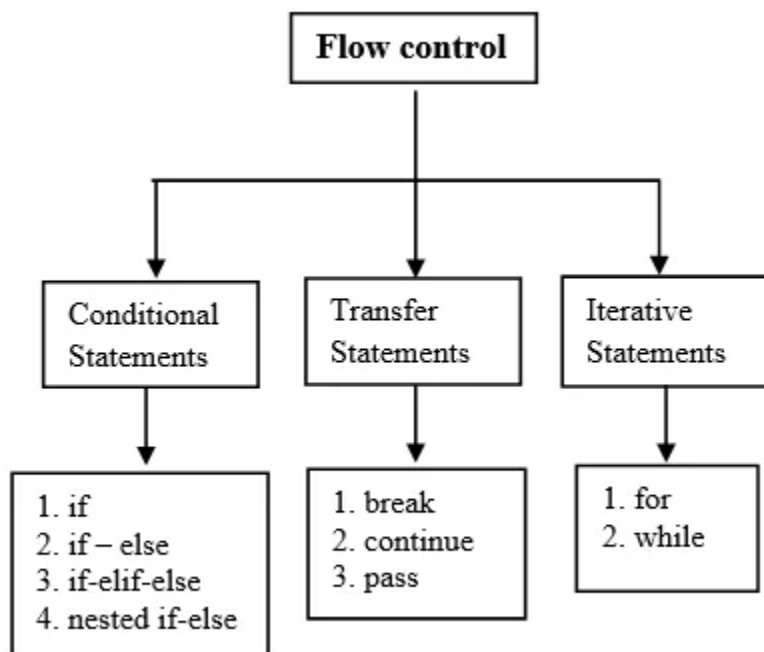
 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering & Technology Department of Information and Communication Technology	
Subject: Programming With Python (01CT1309)	Aim: Develop programs to understand the control structures of python.	
Experiment No: 06	Date:11-8-2025	Enrollment No:92400133108

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.

<div><div>Marwadi University <small>Marwadi Chandarana Group</small></div></div> <div></div>		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:11-8-2025	Enrollment No:92400133108	

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

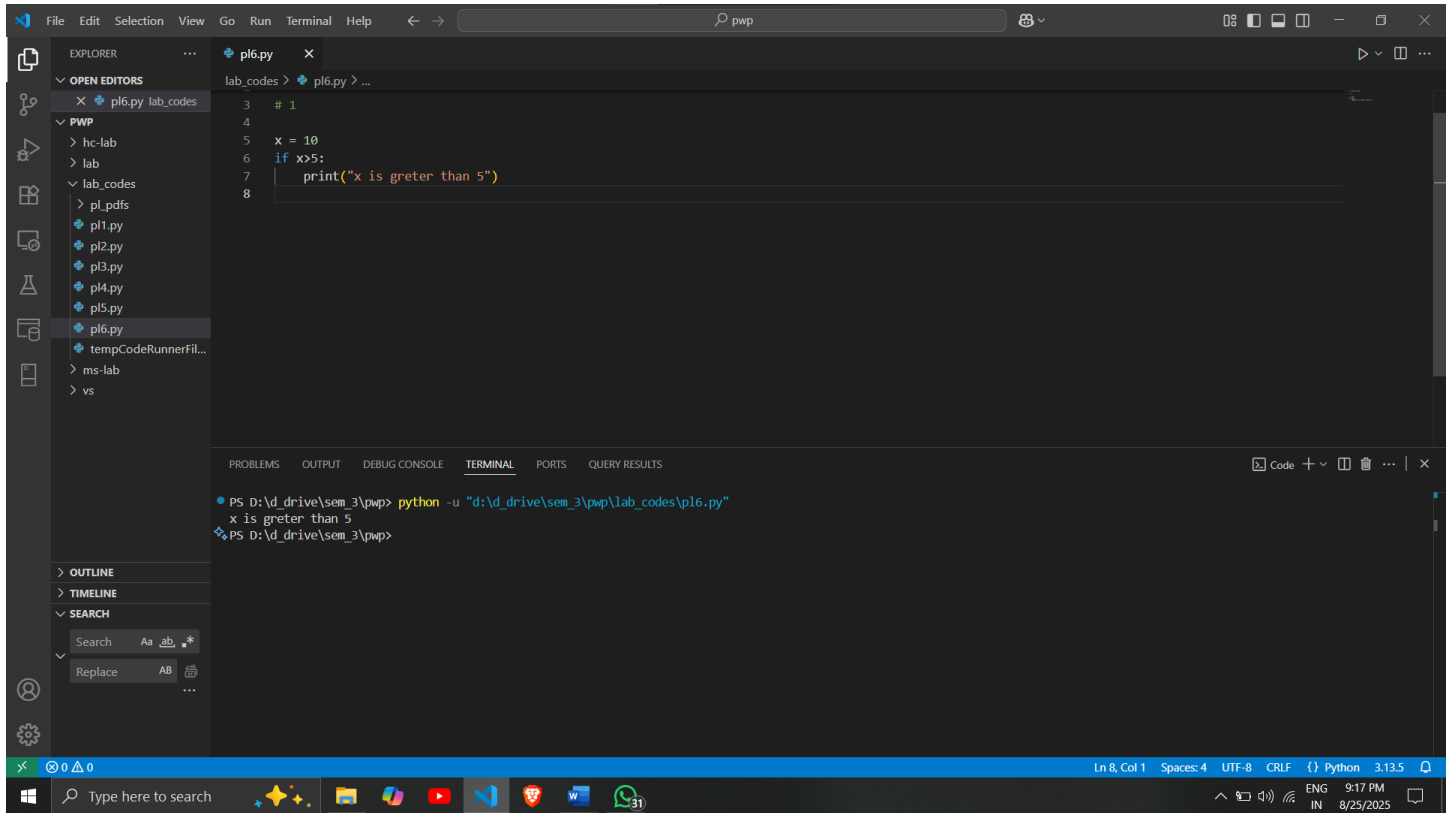
The if statement is used to execute a block of code if a given condition is True. If the condition is False, the code inside the if block is skipped.

Example:

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

Output

 Marwadi University 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:11-8-2025	Enrollment No:92400133108



The screenshot shows a Visual Studio Code editor window with a Python file named `pl6.py` open. The file contains the following code:

```

3  # 1
4
5  x = 10
6  if x>5:
7      print("x is greter than 5")
8

```



The terminal at the bottom shows the command `python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"` being executed, resulting in the output `x is greter than 5`. The status bar at the bottom indicates the file is at line 8, column 1, using UTF-8 encoding and CRLF line endings.

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.

Example:

```
x = 10
```

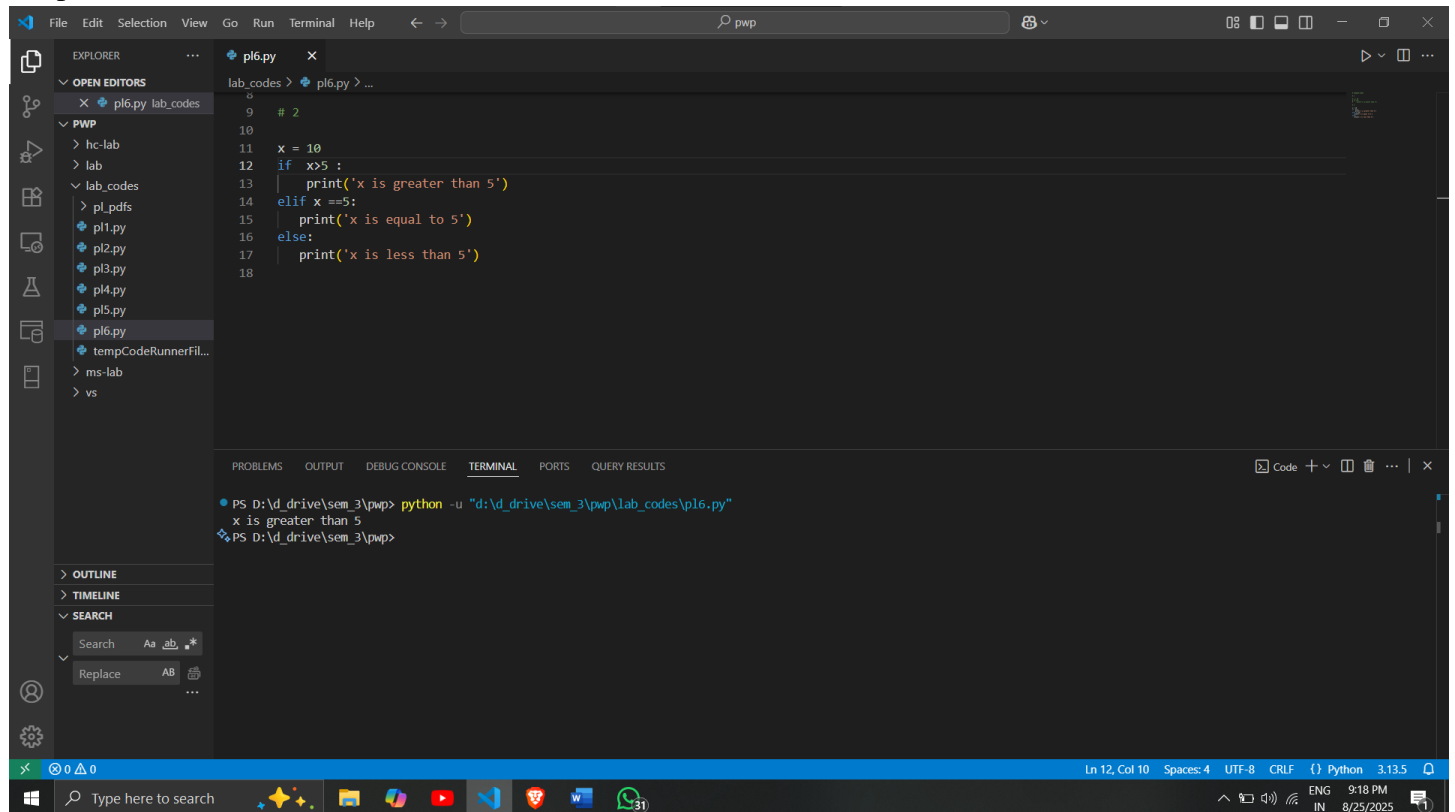
 Marwadi University Marwadi Chandarana Group 	Marwadi University Faculty of Engineering & Technology Department of Information and Communication Technology	
Subject: Programming With Python (01CT1309)	Aim: Develop programs to understand the control structures of python.	
Experiment No: 06	Date:11-8-2025	Enrollment No:92400133108

```

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



The screenshot shows a code editor with a file explorer on the left. The file explorer shows a directory structure with files like pl1.py, pl2.py, pl3.py, pl4.py, pl5.py, and pl6.py. The main editor window displays the Python code from the previous block. Below the code editor, the terminal window shows the command `python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"` and the output `x is greater than 5`.

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:

```

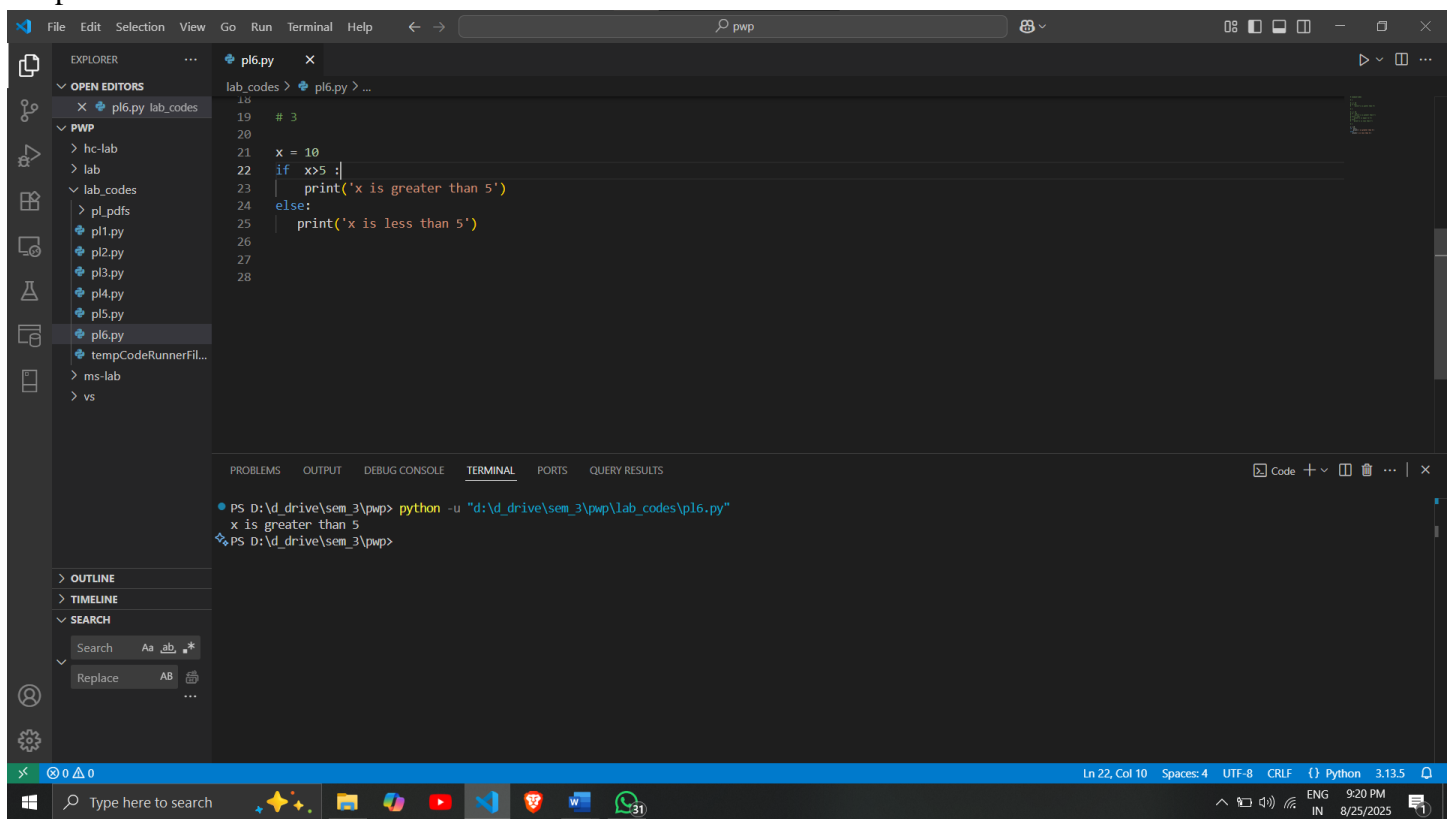
 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering & Technology Department of Information and Communication Technology
Subject: Programming With Python (01CT1309)	Aim: Develop programs to understand the control structures of python.
Experiment No: 06	Date:11-8-2025 Enrollment No:92400133108

```

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

```

Output



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.

Example



age = 35

if age >= 60:

 print("You are a senior citizen.")

else:

 if age >= 18:

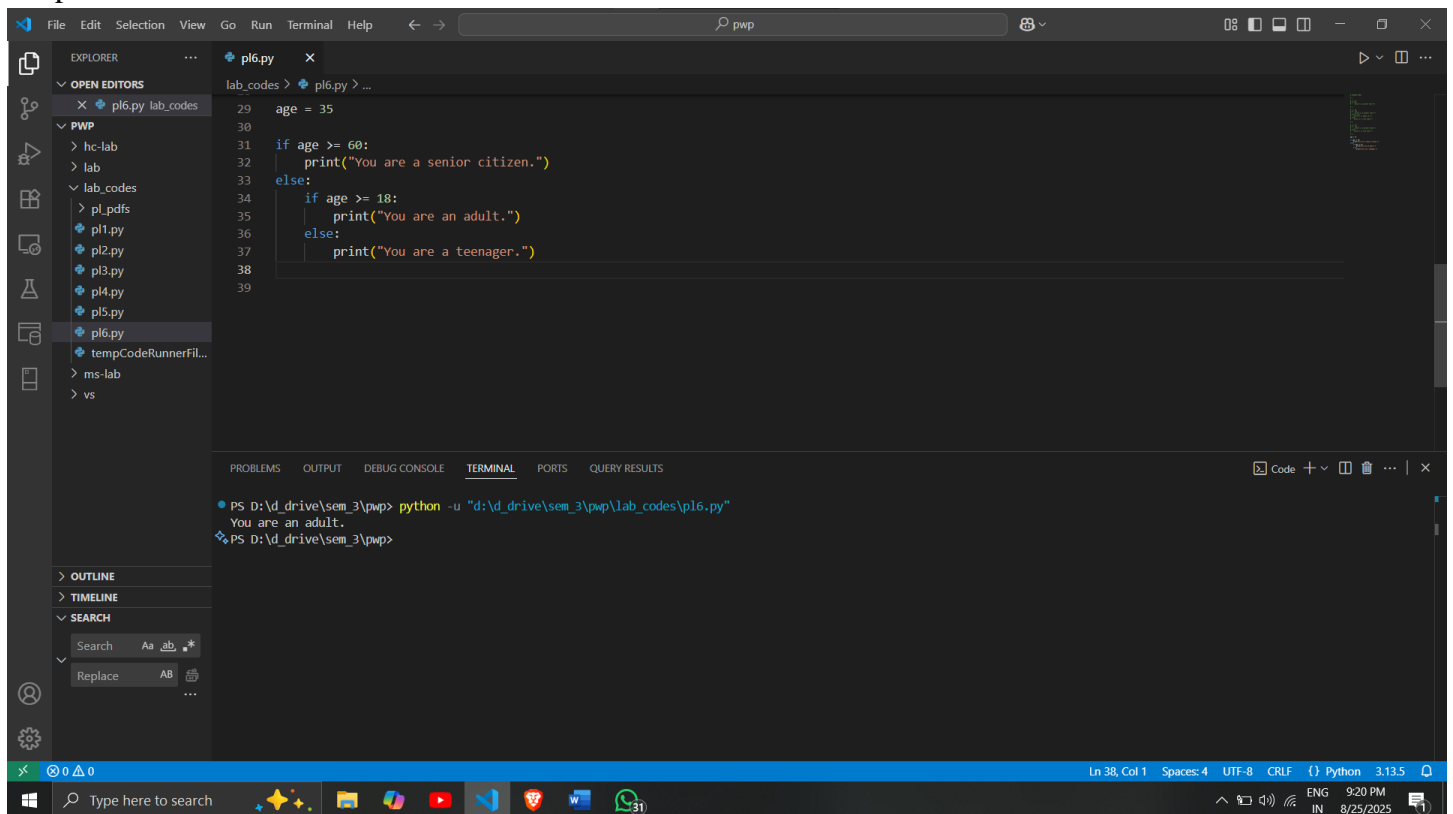
 Marwadi University Marwadi Chandarana Group 	Marwadi University Faculty of Engineering & Technology Department of Information and Communication Technology	
Subject: Programming With Python (01CT1309)	Aim: Develop programs to understand the control structures of python.	
Experiment No: 06	Date:11-8-2025	Enrollment No:92400133108

```

print("You are an adult.")
else:
    print("You are a teenager.")

```

Output



The screenshot shows a Visual Studio Code editor with a Python file named `pl6.py` open. The code in the file is as follows:

```

29 age = 35
30
31 if age >= 60:
32     print("You are a senior citizen.")
33 else:
34     if age >= 18:
35         print("You are an adult.")
36     else:
37         print("You are a teenager.")
38
39

```

The terminal at the bottom shows the command `python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"` being executed, resulting in the output:

```

PS D:\d_drive\sem_3\pwp> python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"
You are an adult.
PS D:\d_drive\sem_3\pwp>

```

Example



num = 10

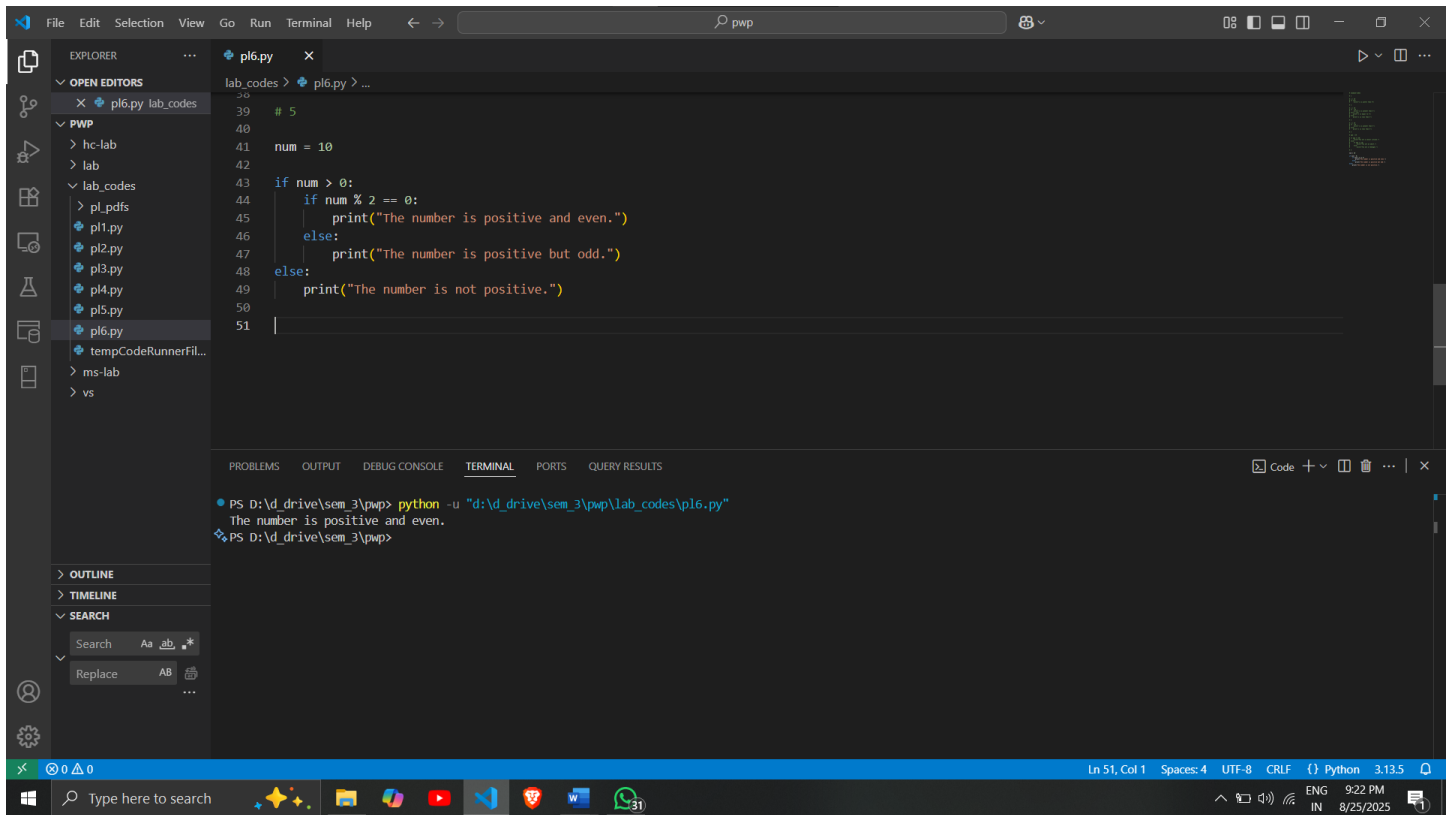
```

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

 Marwadi University 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:11-8-2025	Enrollment No:92400133108



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left displays a file tree with folders like 'lab_codes' and 'PWP'. The main editor window shows a Python file named 'pl6.py' with the following code:

```

39 # 5
40
41 num = 10
42
43 if num > 0:
44     if num % 2 == 0:
45         print("The number is positive and even.")
46     else:
47         print("The number is positive but odd.")
48 else:
49     print("The number is not positive.")
50
51

```

The TERMINAL pane at the bottom shows the command prompt output:

```

PS D:\drive\sem_3\pwp> python -u "d:\drive\sem_3\pwp\lab_codes\pl6.py"
The number is positive and even.
PS D:\drive\sem_3\pwp>

```

The status bar at the bottom indicates the current line and column (Ln 51, Col 1), encoding (UTF-8), and the active language (Python 3.13.5).

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.



Example

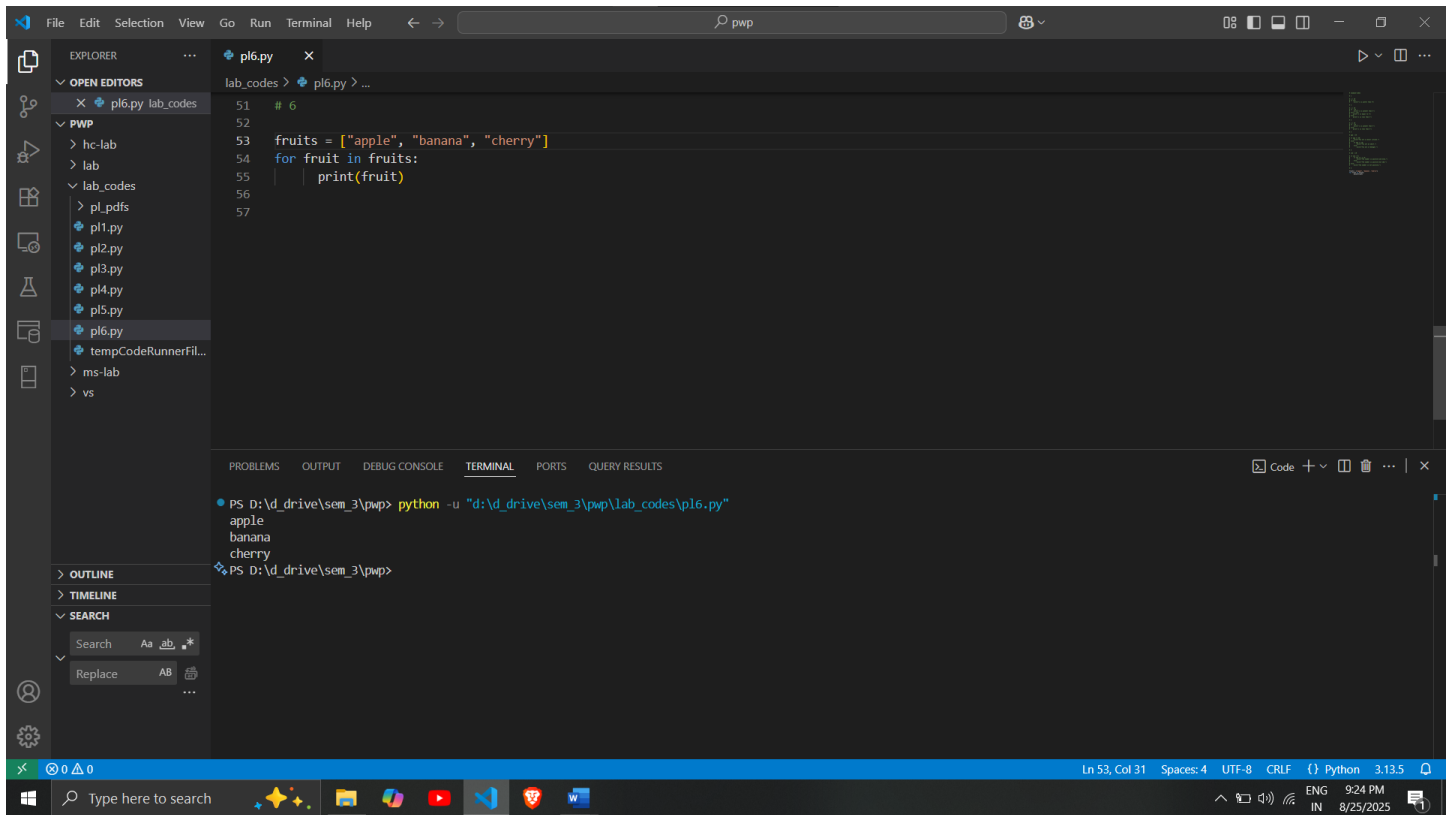
```

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

```

Output

 Marwadi University 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:11-8-2025	Enrollment No:92400133108



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left displays a file tree with folders like 'lab_codes' and 'pl6.py'. The main editor window shows a Python script named 'pl6.py' with the following code:

```

51 # 6
52
53 fruits = ["apple", "banana", "cherry"]
54 for fruit in fruits:
55     print(fruit)
56
57

```

The TERMINAL pane at the bottom shows the command prompt output:

```

PS D:\drive\sem_3\pwp> python -u "d:\drive\sem_3\pwp\lab_codes\pl6.py"
apple
banana
cherry
PS D:\drive\sem_3\pwp>

```

2. while Loop



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

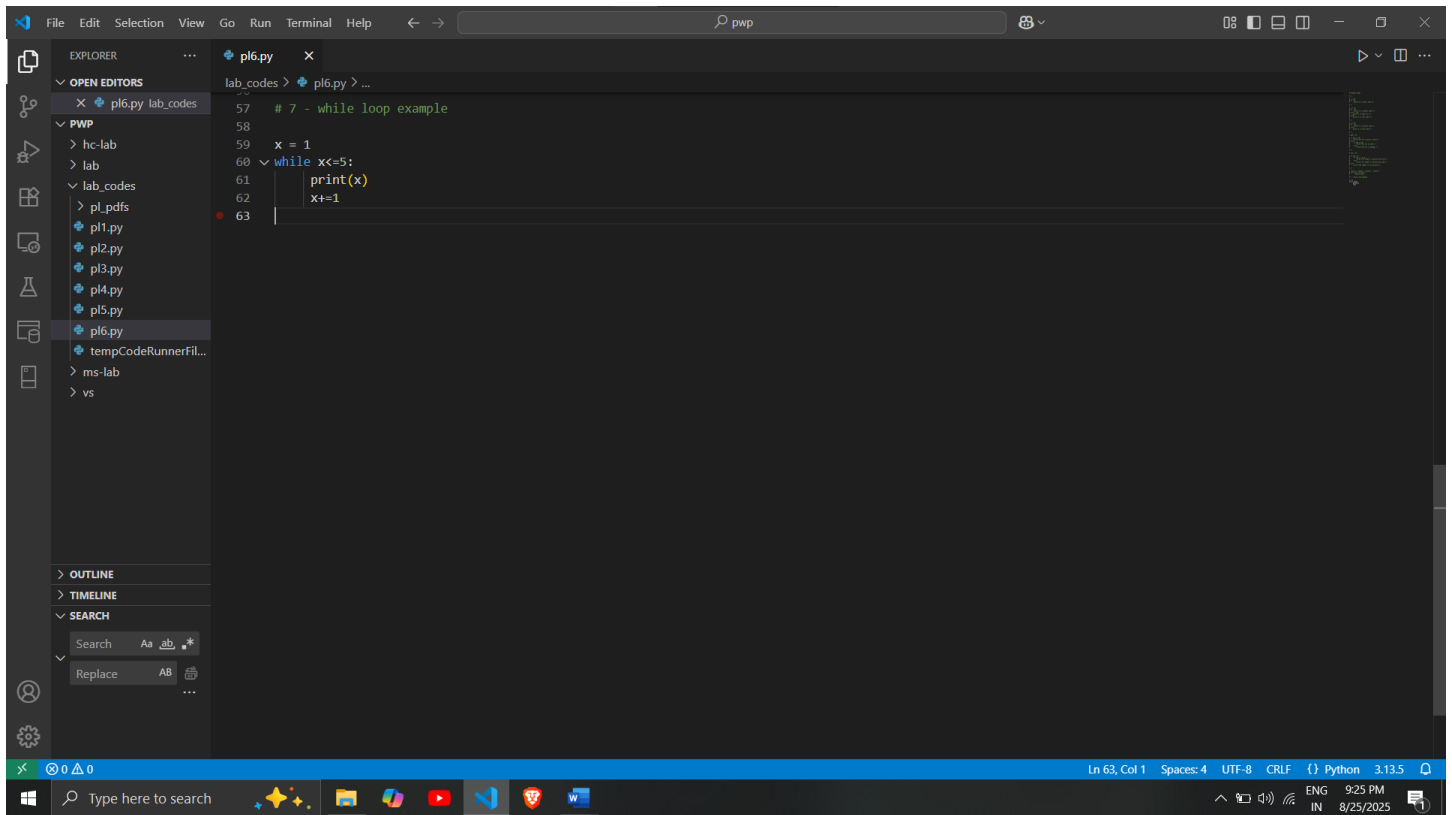
Example

```

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

```


 Marwadi University 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:11-8-2025	Enrollment No:92400133108



```

57  # 7 - while loop example
58
59  x = 1
60  while x<=5:
61      print(x)
62      x+=1
63

```

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

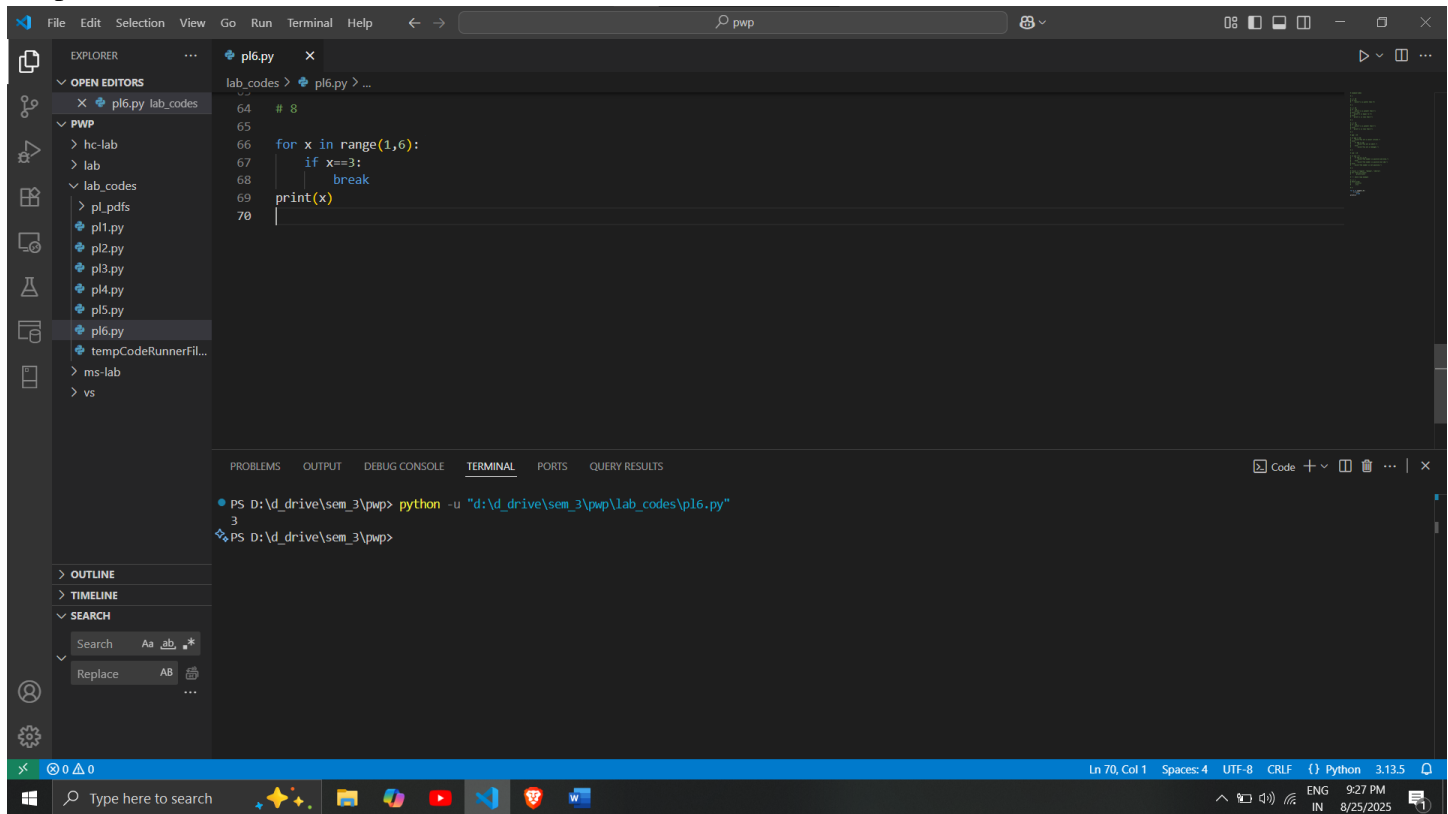
 Marwadi University Marwadi Chandarana Group 	Marwadi University Faculty of Engineering & Technology Department of Information and Communication Technology	
Subject: Programming With Python (01CT1309)	Aim: Develop programs to understand the control structures of python.	
Experiment No: 06	Date:11-8-2025	Enrollment No:92400133108

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



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):
```

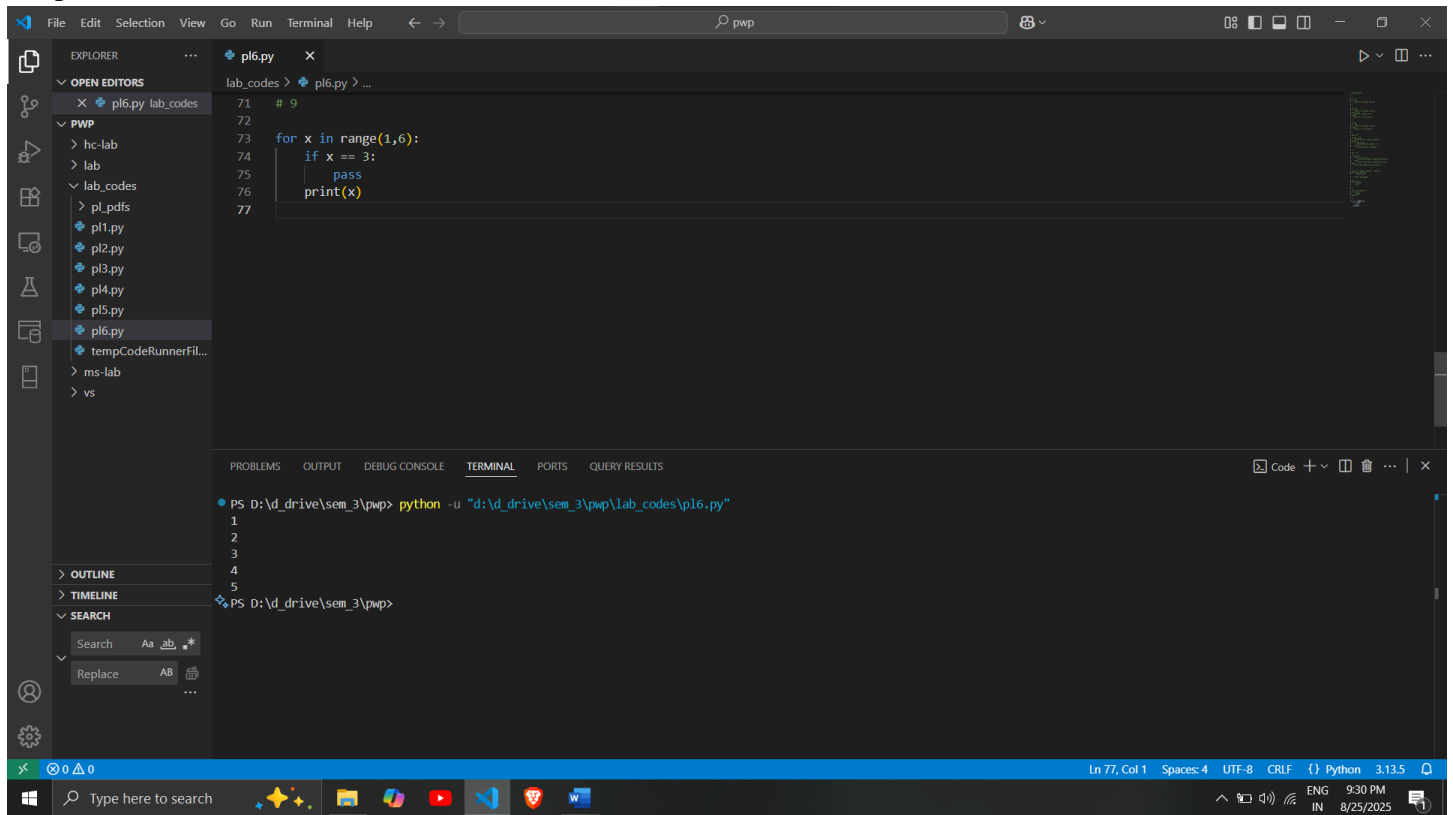
 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering & Technology Department of Information and Communication Technology	
Subject: Programming With Python (01CT1309)	Aim: Develop programs to understand the control structures of python.	
Experiment No: 06	Date:11-8-2025	Enrollment No:92400133108

```

if x == 3:
    pass
print(x)

```

Output



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows a project structure with folders like 'lab_codes' and 'pl6.py'. The main editor window displays a Python script with the following code:

```

71 # 9
72
73 for x in range(1,6):
74     if x == 3:
75         pass
76     print(x)
77

```

The TERMINAL pane at the bottom shows the command prompt output:

```

PS D:\d_drive\sem_3\pwp> python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"
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.
- 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.")

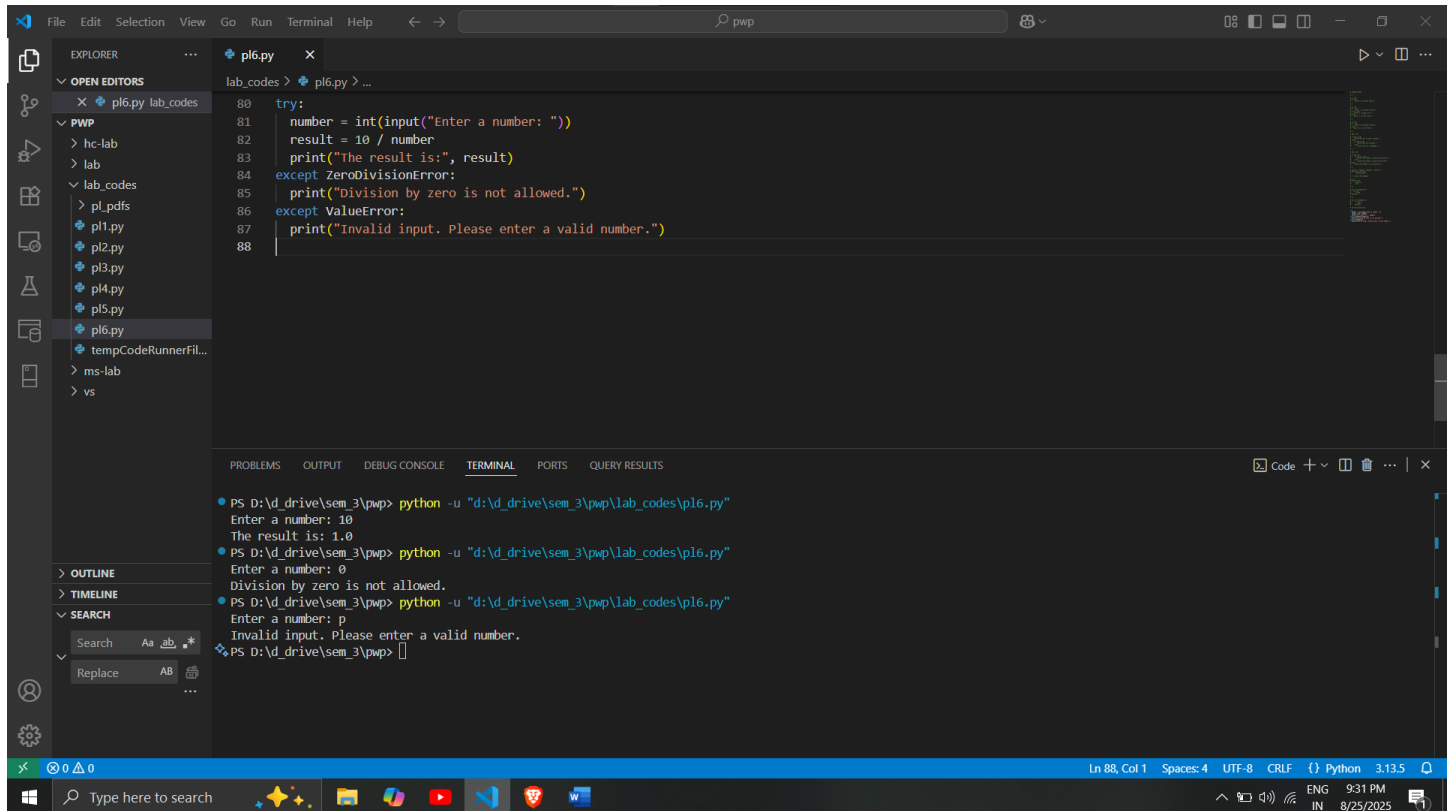
```

except ValueError:

 Marwadi University 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:11-8-2025	Enrollment No:92400133108

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

Output



The screenshot shows a VS Code editor with a file named `pl6.py` open. The code in the editor is as follows:

```
80 try:
81     number = int(input("Enter a number: "))
82     result = 10 / number
83     print("The result is:", result)
84 except ZeroDivisionError:
85     print("Division by zero is not allowed.")
86 except ValueError:
87     print("Invalid input. Please enter a valid number.")
88
```

The terminal at the bottom shows the execution of the program with three different inputs:

```
PS D:\d_drive\sem_3\pwp> python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"
Enter a number: 10
The result is: 1.0

PS D:\d_drive\sem_3\pwp> python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"
Enter a number: 0
Division by zero is not allowed.



PS D:\d_drive\sem_3\pwp> python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"
Enter a number: p
Invalid input. Please enter a valid number.
PS D:\d_drive\sem_3\pwp>
```

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:

 Marwadi University 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:11-8-2025	Enrollment No:92400133108

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))
```

2. User-defined Functions

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)
```



3. Anonymous Functions (Lambda Functions)

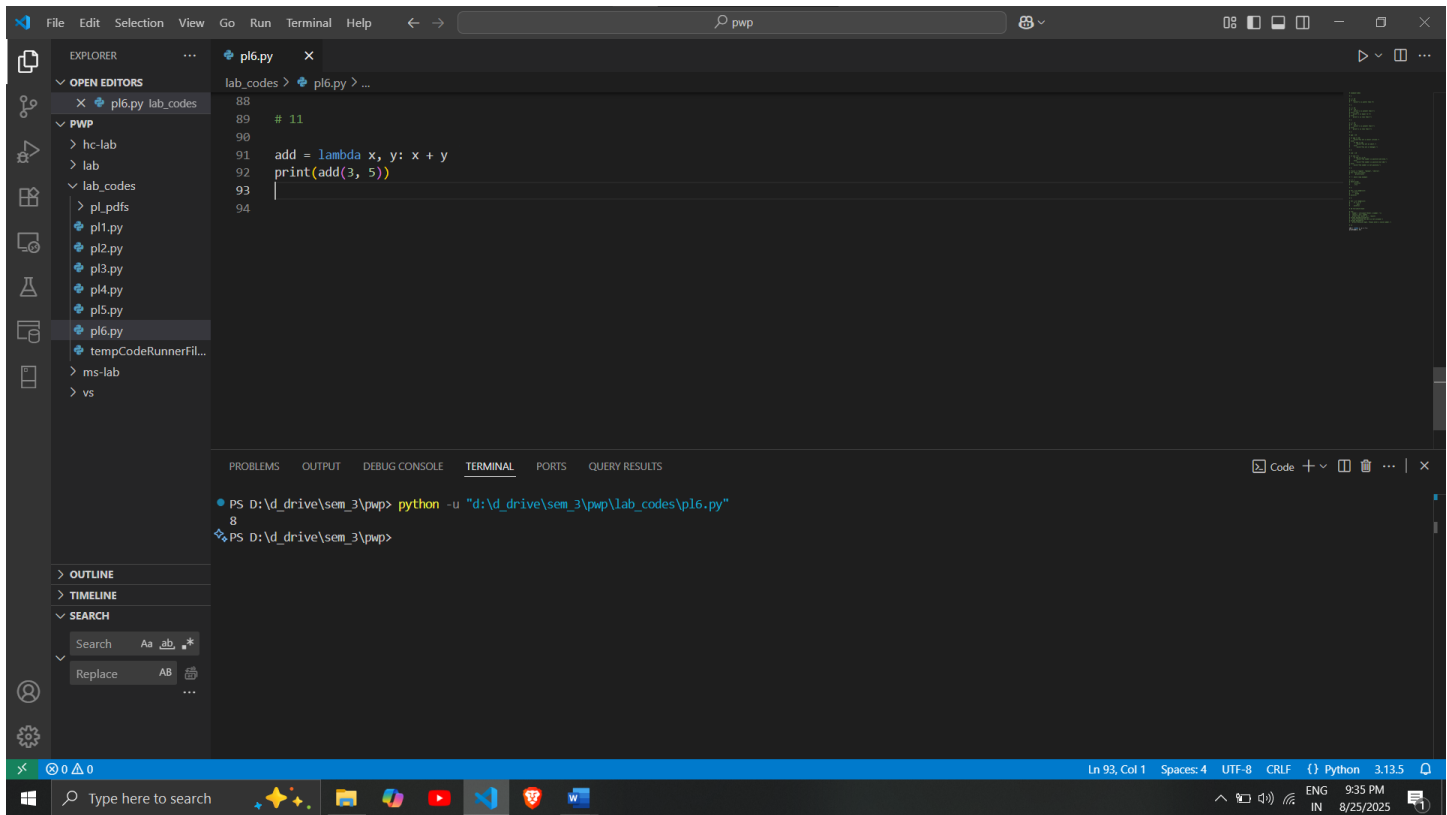
These are small, unnamed functions defined using the lambda keyword. They are typically used for short, simple operations.

Example

```
add = lambda x, y: x + y
print(add(3, 5))
```

Output

 Marwadi University 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:11-8-2025	Enrollment No:92400133108



```

88
89 # 11
90
91 add = lambda x, y: x + y
92 print(add(3, 5))
93
94

```

```

PS D:\d_drive\sem_3\pwp> python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"
8
PS D:\d_drive\sem_3\pwp>

```

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):



```

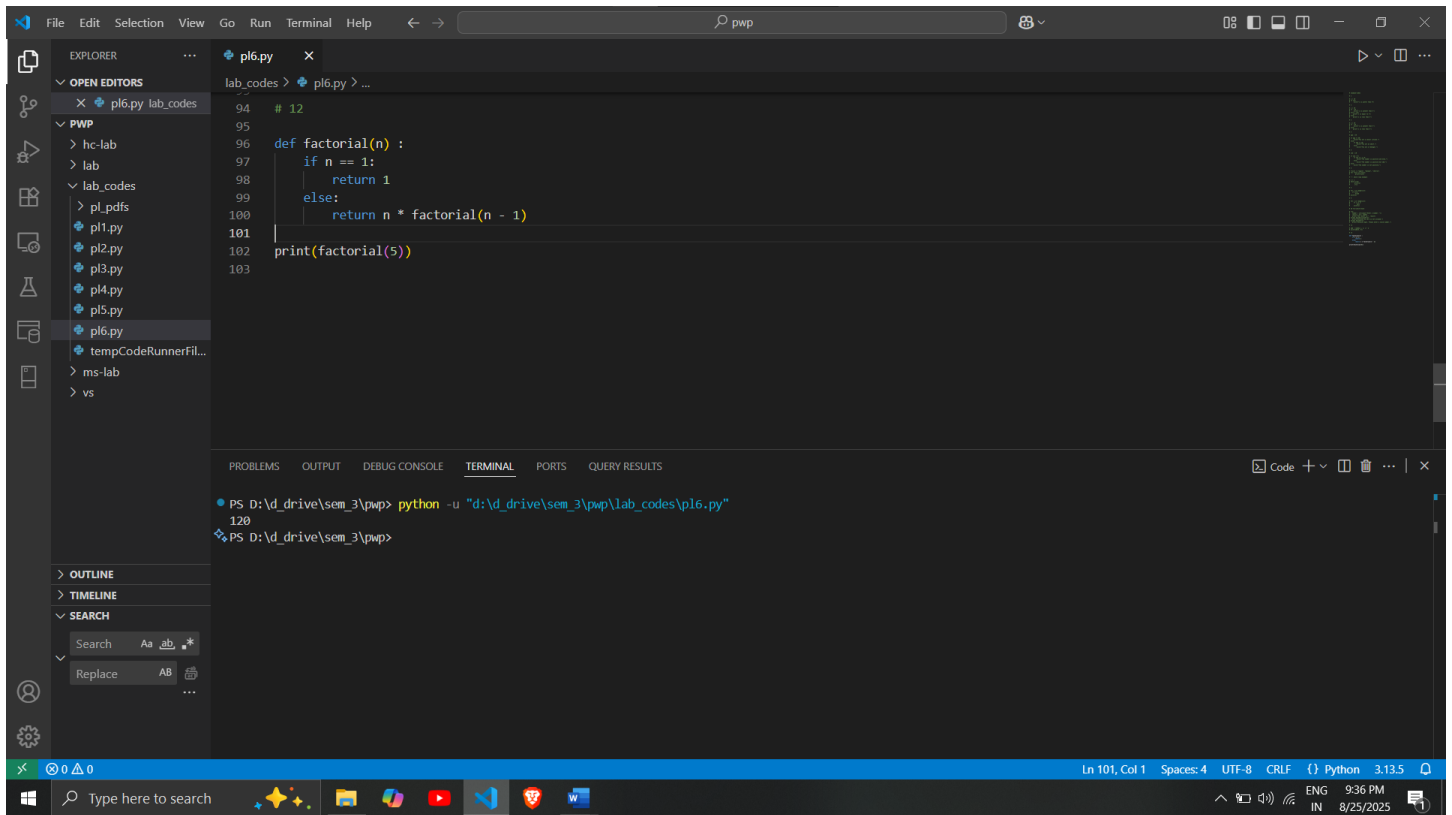
    if n == 1:
        return 1
    else:
        return n * factorial(n - 1)

```

print(factorial(5))

Output

 Marwadi University 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:11-8-2025	Enrollment No:92400133108



```

94 # 12
95
96 def factorial(n):
97     if n == 1:
98         return 1
99     else:
100         return n * factorial(n - 1)
101
102 print(factorial(5))
103

```

```

PS D:\drive\sem_3\pwp> python -u "d:\drive\sem_3\pwp\lab_codes\pl6.py"
120
PS D:\drive\sem_3\pwp>

```

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)

```

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():

```

 Marwadi University 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:11-8-2025	Enrollment No:92400133108

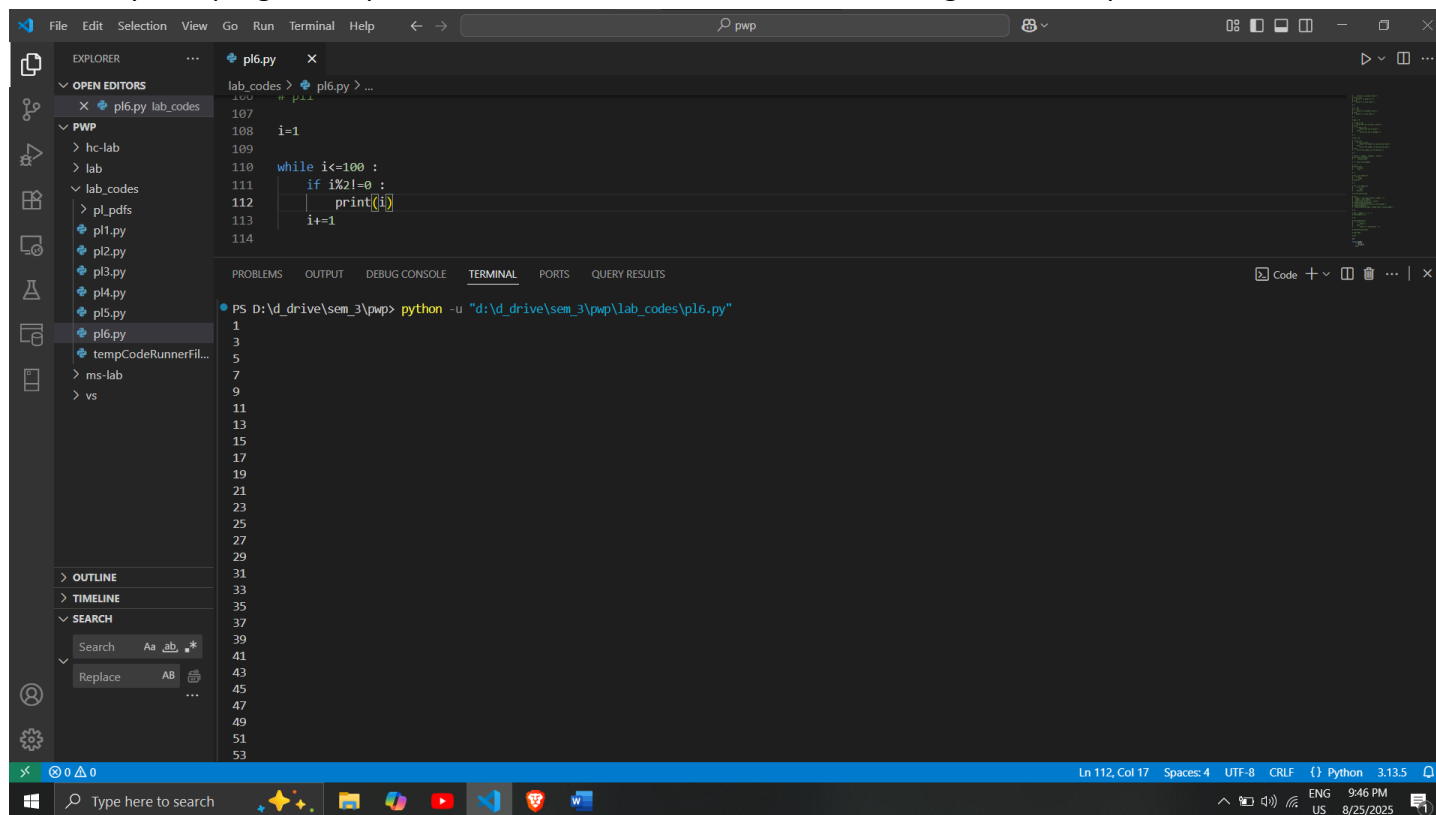
```

for i in range(1, 6):
    yield i
for number in generate_numbers():
    print(number)

```

Post Lab Exercise:

- Write a Python program to print all odd numbers between 1 to 100 using a while loop.





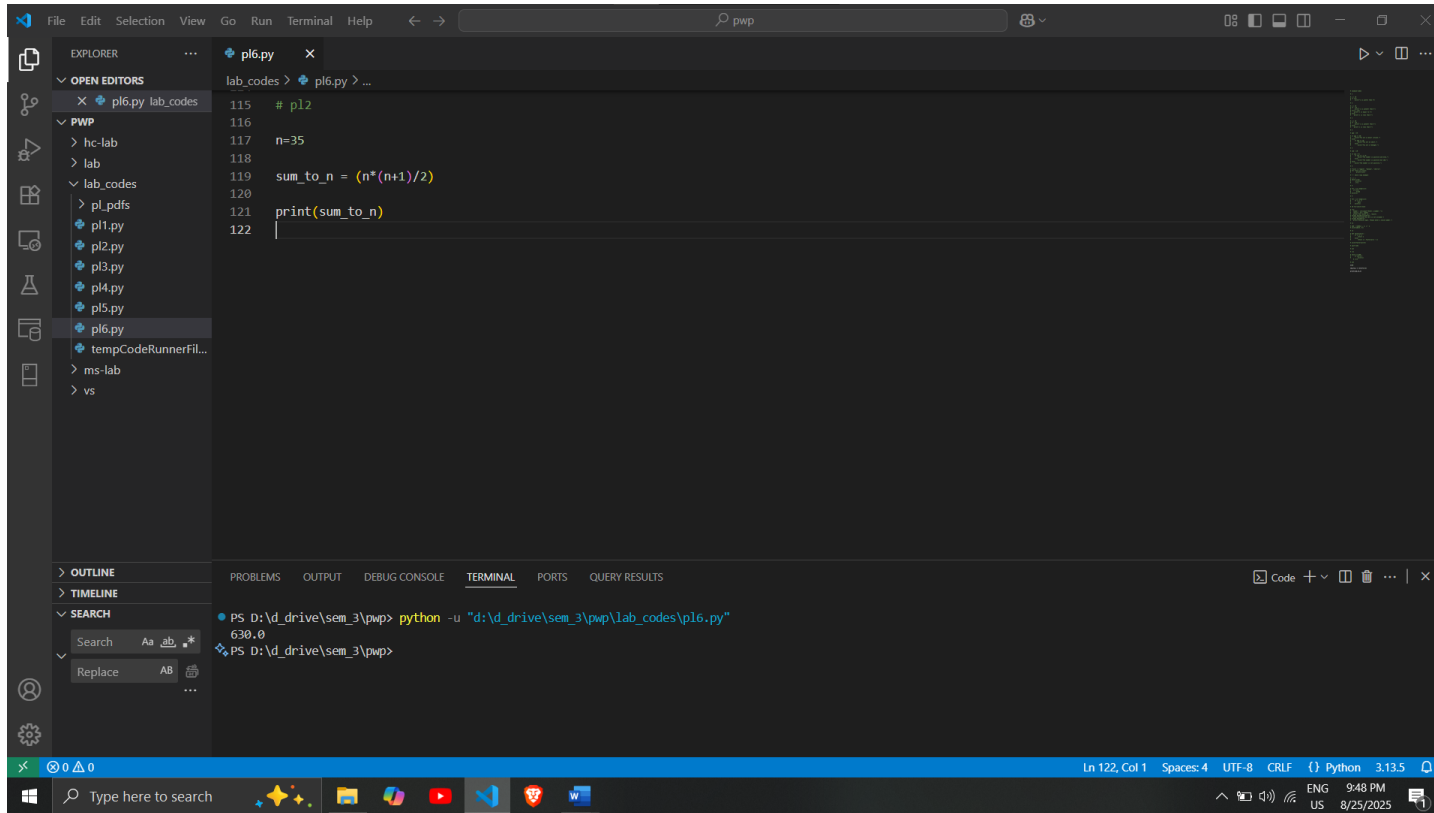
```

lab_codes > pl6.py > ...
107
108 i=1
109
110 while i<=100 :
111     if i%2!=0 :
112         print(i)
113     i+=1
114
PS D:\d_drive\sem_3\pwp> python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"
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

```

- Write a Python program to find the sum of all natural numbers between 1 to n.

 Marwadi University 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:11-8-2025	Enrollment No:92400133108



```

lab_codes > pl6.py > ...
115 # p12
116
117 n=35
118
119 sum_to_n = (n*(n+1))/2
120
121 print(sum_to_n)
122

```



Terminal Output:

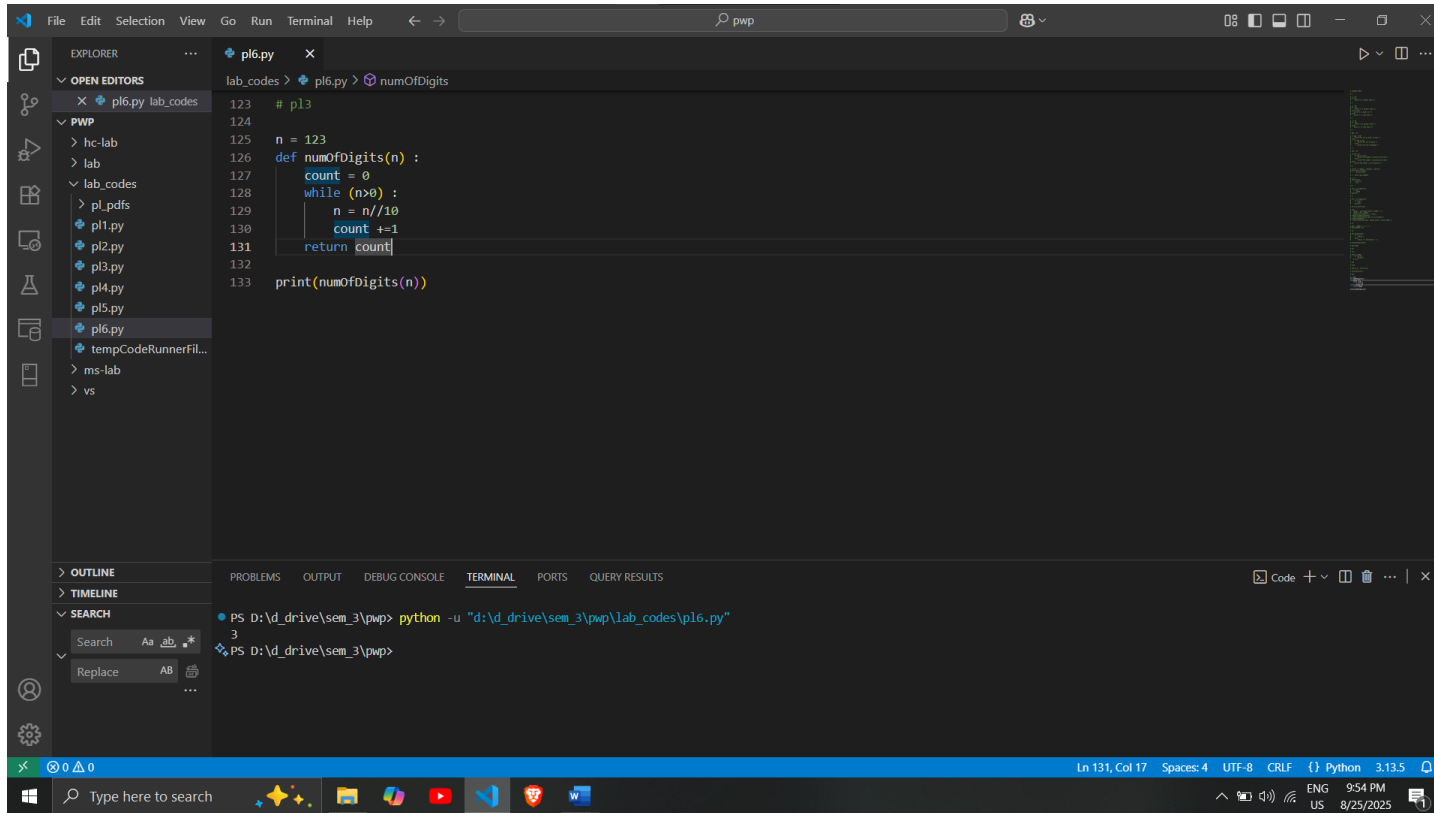
```

PS D:\d_drive\sem_3\pwp> python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"
630.0
PS D:\d_drive\sem_3\pwp>

```

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

 Marwadi University 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:11-8-2025	Enrollment No:92400133108



```

lab_codes > pl6.py > numOfDigits
123 # pl3
124
125 n = 123
126 def numOfDigits(n) :
127     count = 0
128     while (n>0) :
129         n = n//10
130         count +=1
131     return count
132
133 print(numOfDigits(n))



```

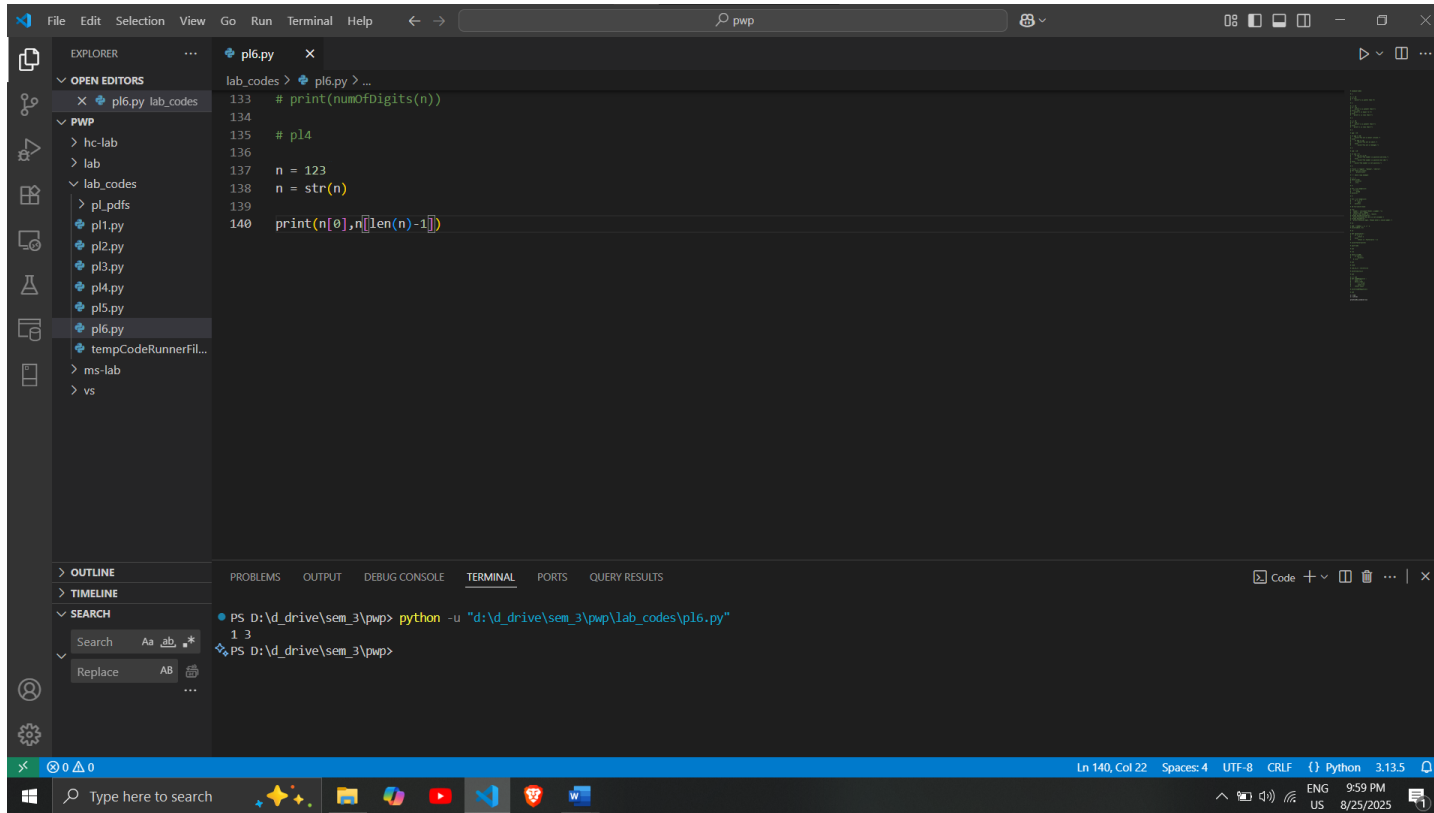
```

PS D:\d_drive\sem_3\pwp> python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"
3
PS D:\d_drive\sem_3\pwp>

```

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

 Marwadi University 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:11-8-2025	Enrollment No:92400133108



```

lab_codes > pl6.py > ...
133 # print(numOfDigits(n))
134
135 # pl4
136
137 n = 123
138 n = str(n)
139
140 print(n[0],n[len(n)-1])


```

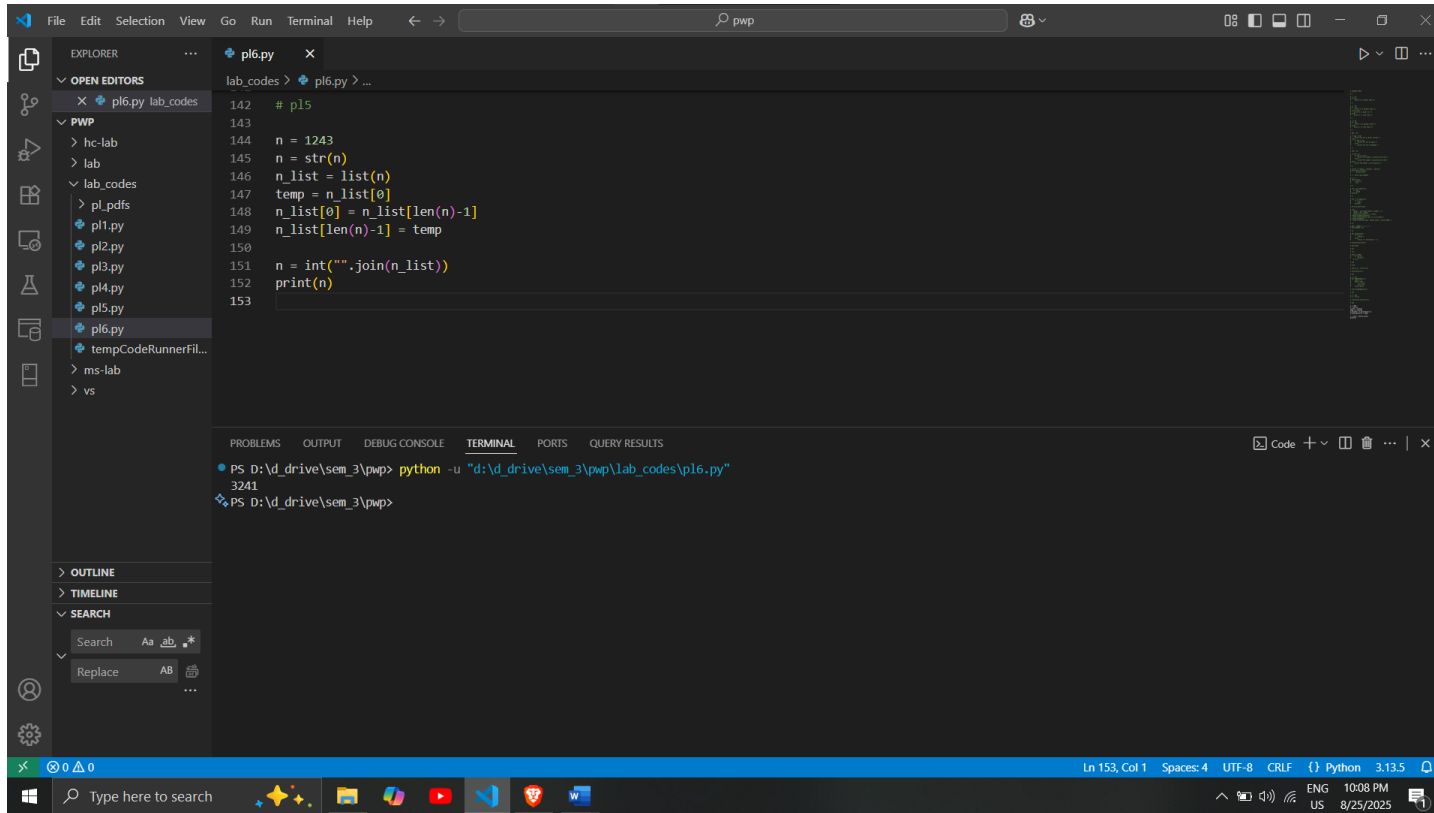
```

PS D:\d_drive\sem_3\pwp> python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"
1 3
PS D:\d_drive\sem_3\pwp>

```

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

 Marwadi University 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:11-8-2025
	Enrollment No:92400133108



```

142 # pl5
143
144 n = 1243
145 n = str(n)
146 n_list = list(n)
147 temp = n_list[0]
148 n_list[0] = n_list[len(n)-1]
149 n_list[len(n)-1] = temp
150
151 n = int("".join(n_list))
152 print(n)
153

```



Terminal Output:

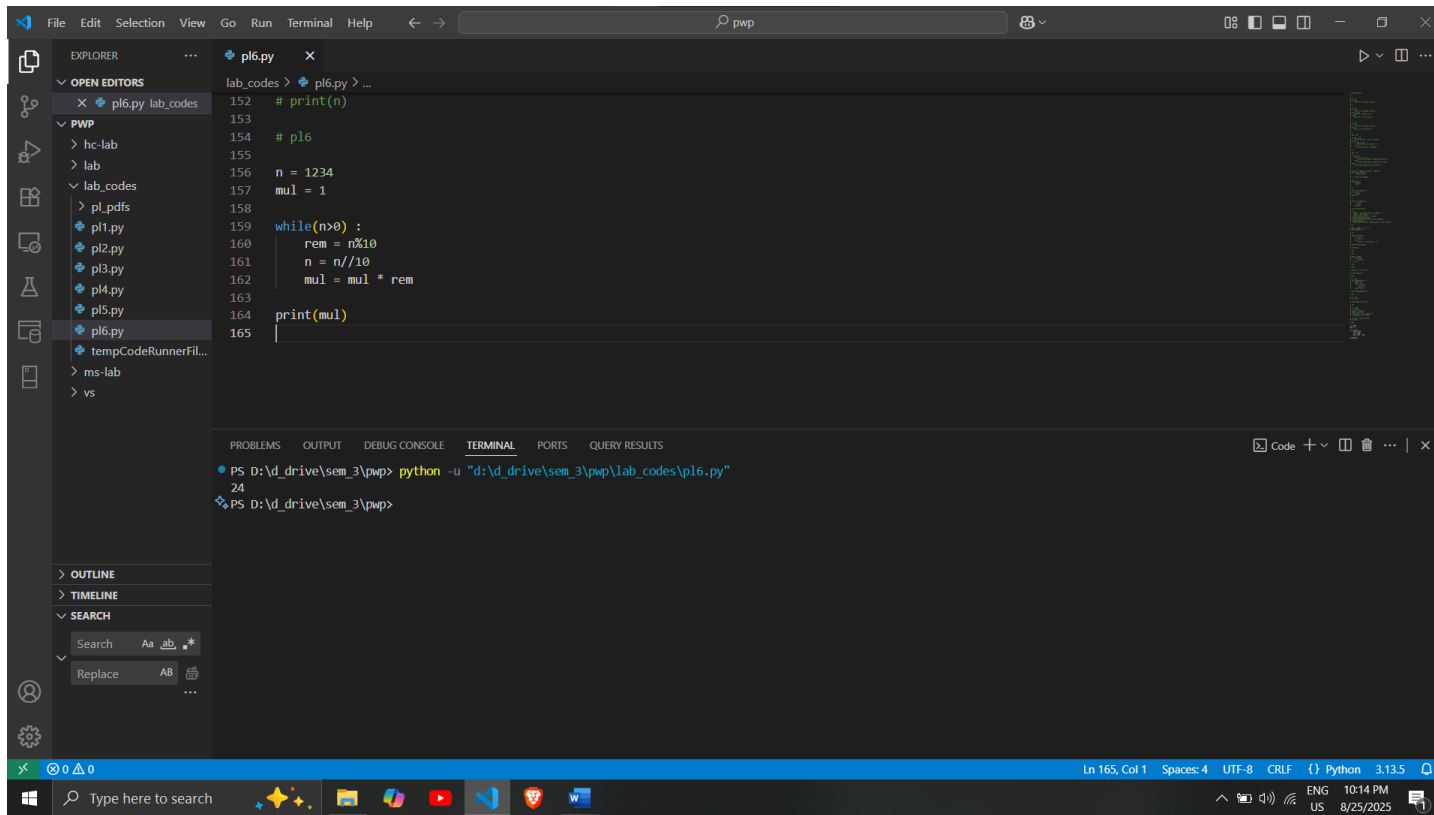
```

PS D:\drive\sem_3\pwp> python -u "d:\drive\sem_3\pwp\lab_codes\pl6.py"
3241
PS D:\drive\sem_3\pwp>

```

f. Write a Python program to calculate the product of digits of a number.

 Marwadi University 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:11-8-2025	Enrollment No:92400133108



```

152 # print(n)
153
154 # pl6
155
156 n = 1234
157 mul = 1
158
159 while(n>0) :
160     rem = n%10
161     n = n//10
162     mul = mul * rem
163
164 print(mul)
165

```



PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULTS

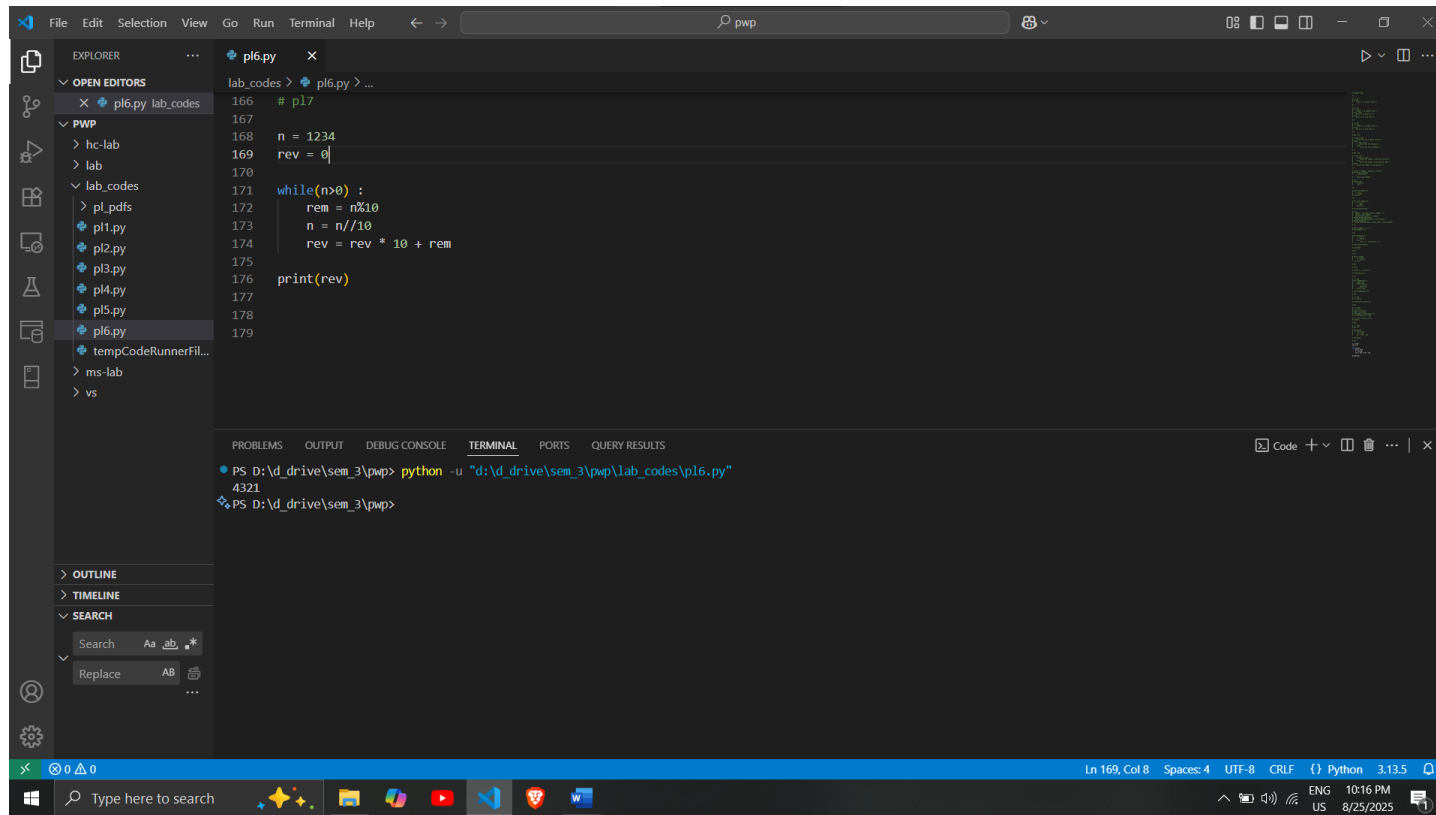
PS D:\d_drive\sem_3\pwp> python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"

24

PS D:\d_drive\sem_3\pwp>

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

 Marwadi University 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:11-8-2025	Enrollment No:92400133108



The screenshot shows the Visual Studio Code interface with a Python file named `pl6.py` open. The code implements a function to reverse a number `n` using a `while` loop. The logic involves extracting the last digit (`rem = n % 10`), updating the reversed number (`rev = rev * 10 + rem`), and removing the last digit from `n` (`n = n // 10`) until `n` becomes zero. The final reversed number is printed.

```

166 # pl7
167
168 n = 1234
169 rev = 0
170
171 while(n>0) :
172     rem = n%10
173     n = n//10
174     rev = rev * 10 + rem
175
176 print(rev)
177
178
179

```

The terminal at the bottom shows the command `python -u "d:\d_drive\sem_3\pwp\lab_codes\pl6.py"` being executed, resulting in the output `4321`.