

A.Yuvan Charan

192424383

SLOT-B

PYTHON PROGRAMMING FOR BLOCK CHAIN PROJECTS

CSA0815

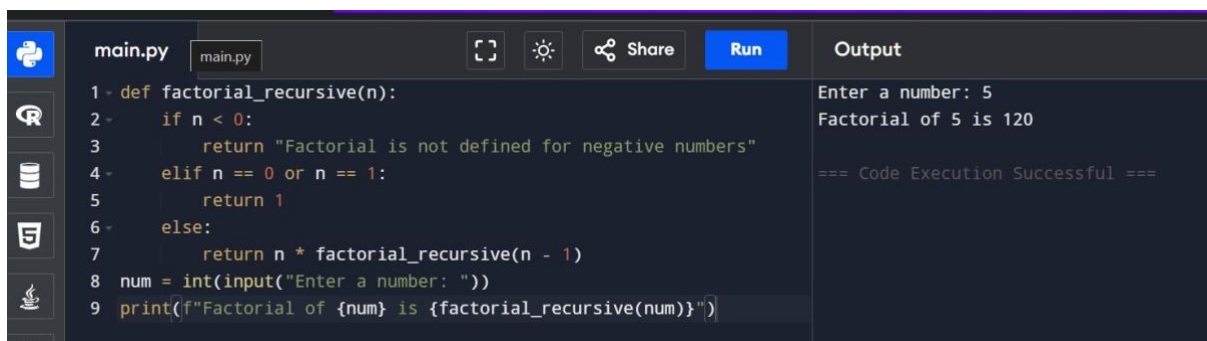
1. Write a program for Function to add two numbers



The screenshot shows a Python IDE with a file named 'main.py'. The code defines a function 'add_numbers(a, b)' that returns the sum of 'a' and 'b'. It then assigns 'num1 = 5' and 'num2 = 3', calls the function, and prints the result. The output pane shows 'The sum of 5 and 3 is: 8' and '=== Code Execution Successful ==='.

```
1- def add_numbers(a, b):
2-     return a + b
3- num1 = 5
4- num2 = 3
5- result = add_numbers(num1, num2)
6-
7- print("The sum of", num1, "and", num2, "is:", result)
8-
```

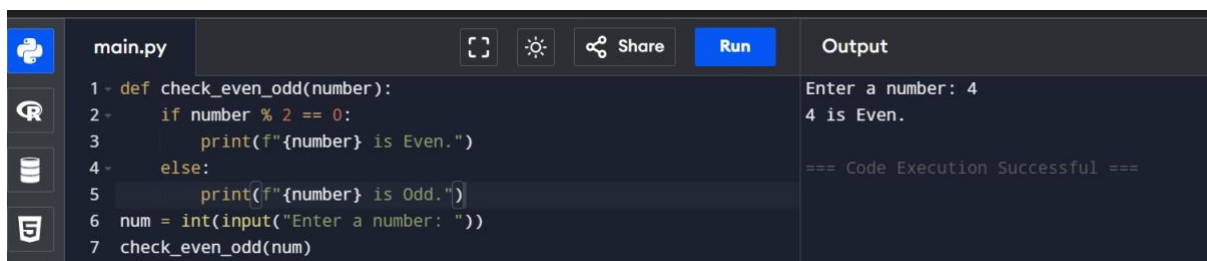
2. Write a program for Function to find factorial of a number



The screenshot shows a Python IDE with a file named 'main.py'. The code defines a recursive function 'factorial_recursive(n)' that handles negative numbers, base cases (0 and 1), and recursive calls. It prompts the user to 'Enter a number: ', takes input, and prints the factorial. The output pane shows 'Enter a number: 5', 'Factorial of 5 is 120', and '=== Code Execution Successful ==='.

```
1- def factorial_recursive(n):
2-     if n < 0:
3-         return "Factorial is not defined for negative numbers"
4-     elif n == 0 or n == 1:
5-         return 1
6-     else:
7-         return n * factorial_recursive(n - 1)
8- num = int(input("Enter a number: "))
9- print(f"Factorial of {num} is {factorial_recursive(num)}")
```

3. Write a program for Function to check even or odd



The screenshot shows a Python IDE with a file named 'main.py'. The code defines a function 'check_even_odd(number)' that uses modulo arithmetic to check if a number is even or odd and prints the result. It prompts the user to 'Enter a number: ', takes input, and calls the function. The output pane shows 'Enter a number: 4', '4 is Even.', and '=== Code Execution Successful ==='.

```
1- def check_even_odd(number):
2-     if number % 2 == 0:
3-         print(f"{number} is Even.")
4-     else:
5-         print(f"{number} is Odd.")
6- num = int(input("Enter a number: "))
7- check_even_odd(num)
```

4. Write a program for Function to find power of a number

```
main.py
1 - def power(base, exponent):
2     result = 1
3     for _ in range(abs(exponent)):
4         result *= base
5     if exponent < 0:
6         return 1 / result
7     return result
8 base = float(input("Enter the base: "))
9 exponent = int(input("Enter the exponent: "))
10 print(f"{base} raised to the power of {exponent} is {power(base, exponent)}")
```

Output

```
Enter the base: 2
Enter the exponent: 6
2.0 raised to the power of 6 is 64.0

=== Code Execution Successful ===
```

5. Write a program for Function to swap two numbers

```
main.py
1 - def swap_numbers(a, b):
2     print("Before swapping: a =", a, "b =", b)
3     a, b = b, a
4     print("After swapping: a =", a, "b =", b)
5     return a, b
6 x = 10
7 y = 20
8 x, y = swap_numbers(x, y)
9
```

Output

```
Before swapping: a = 10 b = 20
After swapping: a = 20 b = 10

=== Code Execution Successful ===
```

6. 1. len() - Get string length

#Python Program

#Simple string program using built in function

```
text = "Hello, #Python" print(len(text))
```

Output: 14

```
main.py
1 text = "Hello, #Python"
2 print(len(text))
```

Output

```
14

=== Code Execution Successful ===
```

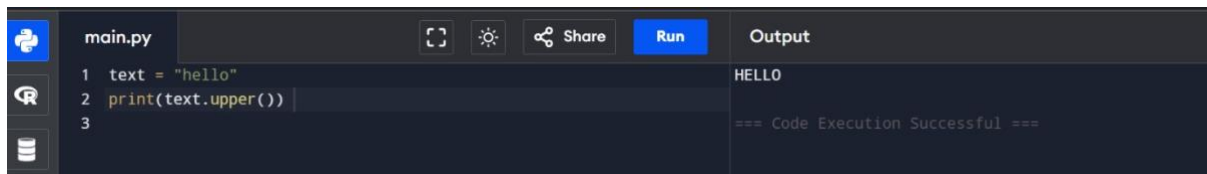
2. upper() - Convert to uppercase

#Python Program

#Simple string program using built in function text

```
= "hello"
```

```
print(text.upper()) # Output: HELLO
```



```
main.py
1 text = "hello"
2 print(text.upper())
3
```

Output

```
HELLO
=== Code Execution Successful ===
```

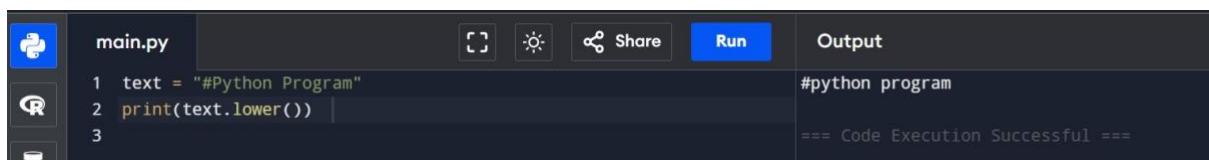
3. lower() - Convert to lowercase

#Python Program

#Simple string program using built in function

```
text = "Python Program"
```

```
print(text.lower()) # Output: python program
```



```
main.py
1 text = "Python Program"
2 print(text.lower())
3
```

Output

```
python program
=== Code Execution Successful ===
```

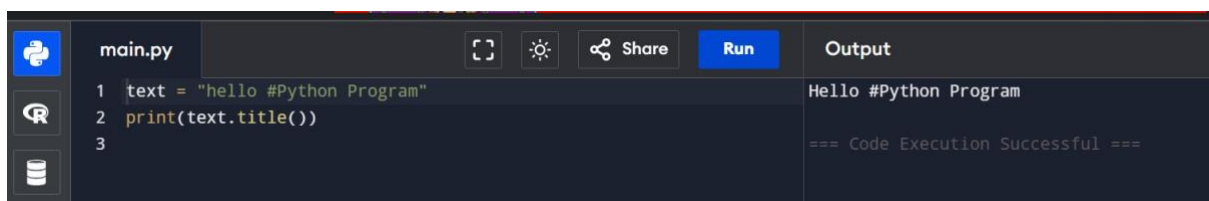
4. title() - Convert to title case

#Python Program

#Simple string program using built in function

```
text = "hello Python Program"
```

```
print(text.title()) # Output: Hello Python Program
```



```
main.py
1 text = "hello Python Program"
2 print(text.title())
3
```

Output

```
Hello Python Program
=== Code Execution Successful ===
```

5. capitalize() - Capitalize first letter

#Python Program

#Simple string program using built in function

```
text = "hello Python Program"
```

```
print(text.capitalize()) # Output: Hello Python Program
```

```
main.py
1 text = "hello #Python Program"
2 print(text.capitalize())
3
```

Output

```
Hello #python program
=== Code Execution Successful ===
```

6. strip() - Remove leading and trailing spaces

#Python Program

#Simple string program using built in function

```
text = "  #Python Program  "; print(text.strip())
```

Output: #Python Program

```
main.py
1 text = " python "
2 print(text.rstrip())
3
```

Output

```
python
=== Code Execution Successful ===
```

7. lstrip() - Remove leading spaces

#Python Program

#Simple string program using built in function

```
text = "  Hello  "; print(text.lstrip())
```

Output: Hello

```
main.py
1 text = " Hello"
2 print(text.lstrip())
3
```

Output

```
Hello
=== Code Execution Successful ===
```

8. rstrip() - Remove trailing spaces

#Python Program

#Simple string program using built in function text

```
text = "#Python Program "
```

print(text.rstrip()) # Output: #Python Program

```
main.py
1 text = "#Python Program"
2 print(text.lower())
3
```

Output

```
#python program
=== Code Execution Successful ===
```

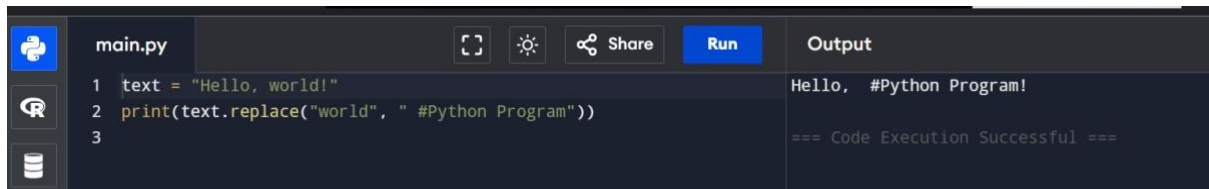
9. replace() - Replace substring

#Python Program

#Simple string program using built in function text

= "Hello, world!";

print(text.replace("world", " #Python Program")) # Output: Hello, #Python Program!



The screenshot shows a Python IDE with a file named 'main.py'. The code contains three lines: `1 text = "Hello, world!"`, `2 print(text.replace("world", " #Python Program"))`, and `3`. The 'Run' button is highlighted. The 'Output' pane on the right shows the result: `Hello, #Python Program!` and a success message: `=== Code Execution Successful ===`.

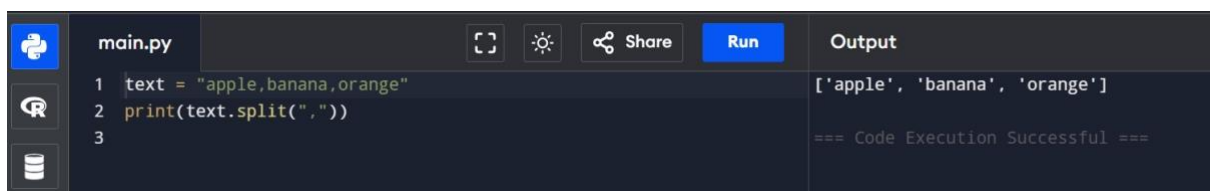
10. split() - Split string into list

#Python Program

#Simple string program using built in function text =

"apple,banana,orange"; print(text.split(",")) #

Output: [apple; banana ;orange;]



The screenshot shows a Python IDE with a file named 'main.py'. The code contains three lines: `1 text = "apple,banana,orange"`, `2 print(text.split(","))`, and `3`. The 'Run' button is highlighted. The 'Output' pane on the right shows the result: `['apple', 'banana', 'orange']` and a success message: `=== Code Execution Successful ===`.

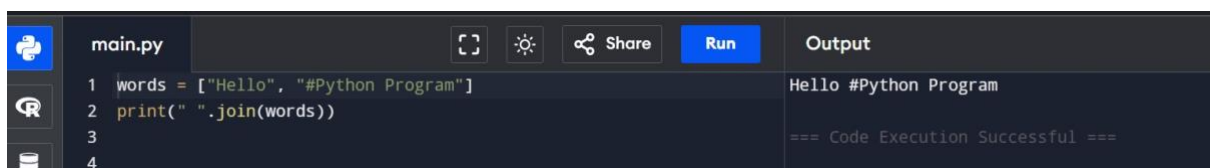
11. join() - Join list into string

#Python Program

#Simple string program using built in function

words = ["Hello", " #Python Program"]; print(";

words)) # Output: Hello #Python Program



The screenshot shows a Python IDE with a file named 'main.py'. The code contains four lines: `1 words = ["Hello", " #Python Program"]`, `2 print(";".join(words))`, `3`, and `4`. The 'Run' button is highlighted. The 'Output' pane on the right shows the result: `Hello #Python Program` and a success message: `=== Code Execution Successful ===`.

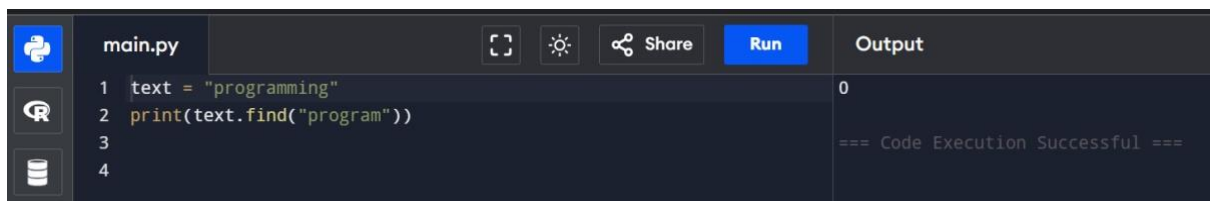
12. find() - Find substring index

#Python Program

#Simple string program using built in function

```
text = "programming"
```

```
print(text.find("program")) # Output: 7
```



The screenshot shows a Python IDE with a file named 'main.py'. The code in the editor is:

```
1 text = "programming"
2 print(text.find("program"))
3
4
```

The 'Output' pane on the right shows the result of the execution:

```
0
=== Code Execution Successful ===
```

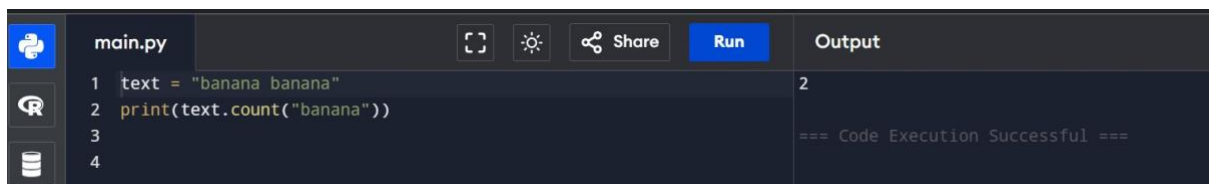
13. count() - Count occurrences of substring

#Python Program

#Simple string program using built in function text

```
text = "banana banana"
```

```
print(text.count("banana")) # Output: 2
```



The screenshot shows a Python IDE with a file named 'main.py'. The code in the editor is:

```
1 text = "banana banana"
2 print(text.count("banana"))
3
4
```

The 'Output' pane on the right shows the result of the execution:

```
2
=== Code Execution Successful ===
```

14. startswith() - Check if string starts with substring

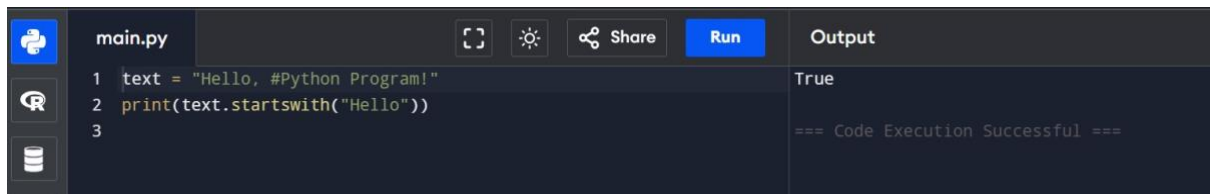
#Python Program

#Simple string program using built in function text =

```
text = "Hello, #Python Program!"
```

```
print(text.startswith("Hello")) # Output:
```

```
True
```



```
main.py
1 text = "Hello, #Python Program!"
2 print(text.startswith("Hello"))
3
```

Output

```
True
=== Code Execution Successful ===
```

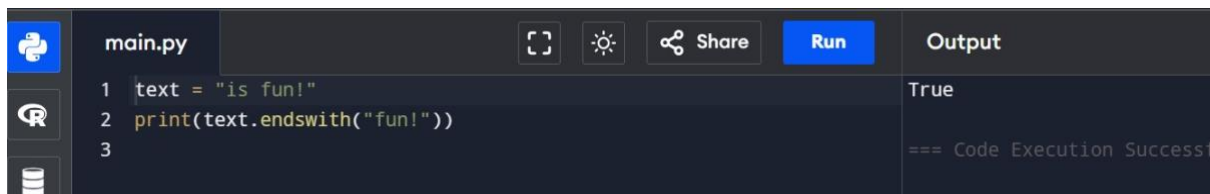
15. endswith() - Check if string ends with substring

#Python Program

#Simple string program using built in function text =

"is fun!"

print(text.endswith("fun!")) # Output: True



```
main.py
1 text = "is fun!"
2 print(text.endswith("fun!"))
3
```

Output

```
True
=== Code Execution Successful ===
```

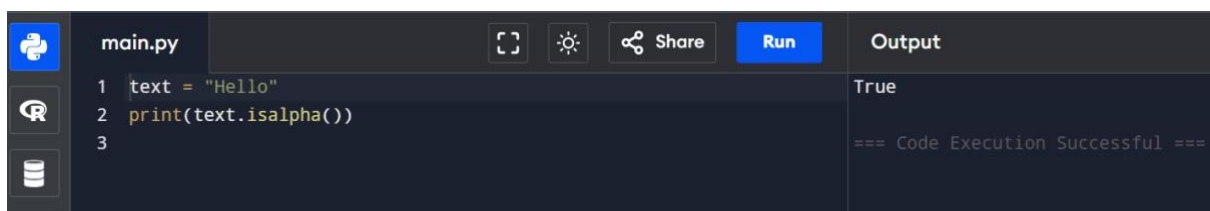
16. isalpha() - Check if all characters are alphabets

#Python Program

#Simple string program using built in function

text = "Hello" print(text.isalpha())

Output: True



```
main.py
1 text = "Hello"
2 print(text.isalpha())
3
```

Output

```
True
=== Code Execution Successful ===
```

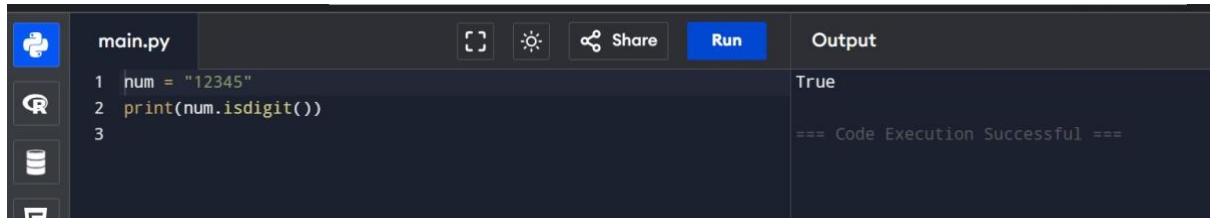
17. isdigit() - Check if all characters are digits

#Python Program

#Simple string program using built in function

num = "12345"; print(num.isdigit())

Output: True



The screenshot shows a Python IDE with a file named 'main.py'. The code in the editor is:

```
1 num = "12345"
2 print(num.isdigit())
3
```

The 'Output' pane on the right shows the result of the execution:

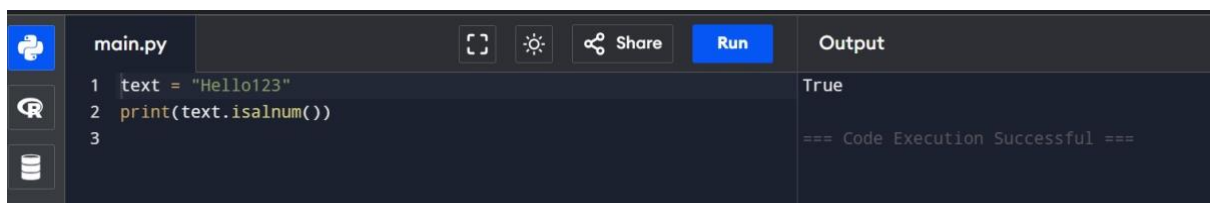
```
True
=== Code Execution Successful ===
```

18. isalnum() - Check if string is alphanumeric

#Python Program

#Simple string program using built in function text =

"Hello123"; print(text.isalnum()) # Output: True



The screenshot shows a Python IDE with a file named 'main.py'. The code in the editor is:

```
1 text = "Hello123"
2 print(text.isalnum())
3
```

The 'Output' pane on the right shows the result of the execution:

```
True
=== Code Execution Successful ===
```

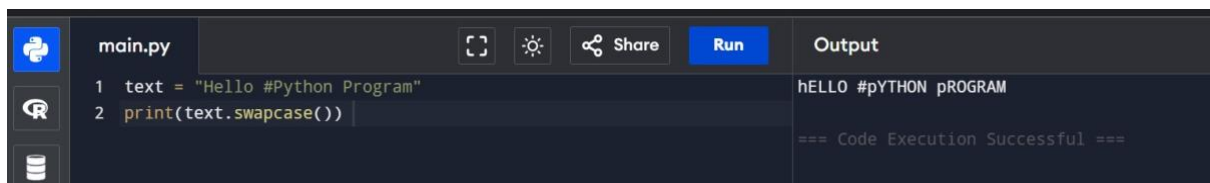
19. swapcase() - Swap case of characters

#Python Program

#Simple string program using built in function text =

"Hello #Python Program";

print(text.swapcase()) # Output: hELLO #Python Program



The screenshot shows a Python IDE with a file named 'main.py'. The code in the editor is:

```
1 text = "Hello #Python Program"
2 print(text.swapcase())
```

The 'Output' pane on the right shows the result of the execution:

```
hELLO #pYTHON pROGRAM
=== Code Execution Successful ===
```

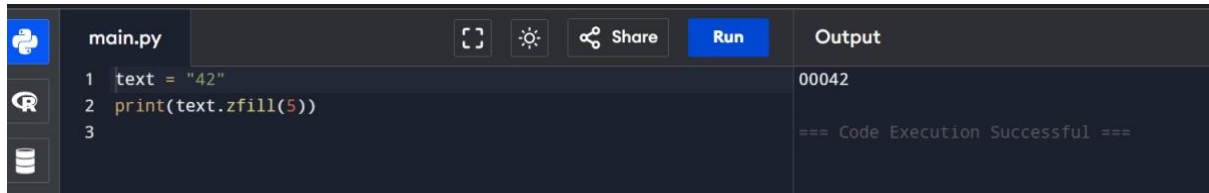
20. zfill() - Pad string with zeros

#Python Program

#Simple string program using built in function text

= "42";

print(text.zfill(5))



The screenshot shows a Python IDE interface with a dark theme. On the left, there is a sidebar with icons for Python, a file explorer, and a database. The main editor area displays a file named 'main.py' with the following code:

```
1 text = "42"  
2 print(text.zfill(5))  
3
```

At the top of the editor, there are icons for running the code, a settings gear, and a 'Share' button. A blue 'Run' button is also present. To the right of the editor, the 'Output' panel shows the result of the execution:

```
00042  
  
=== Code Execution Successful ===
```