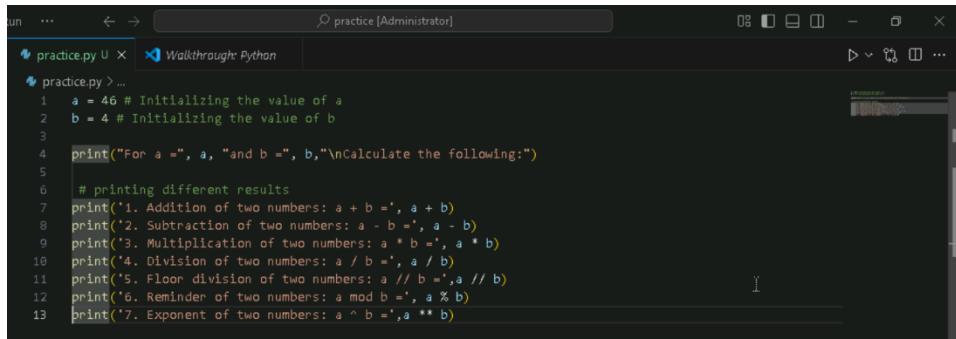


1) Arithmetic Operators



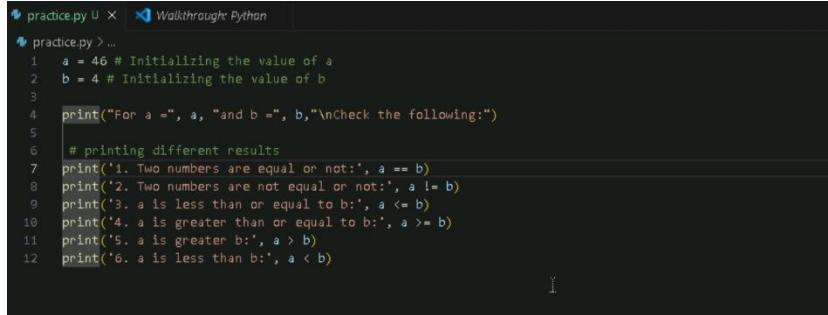
A screenshot of a Windows command prompt window titled "practice [Administrator]". The current directory is "C:\Users\user\PycharmProjects\Walkthrough_Python". The command entered is "practice.py >...". The code in practice.py initializes variables a = 46 and b = 4, then prints various arithmetic operations: addition (a + b), subtraction (a - b), multiplication (a * b), division (a / b), floor division (a // b), modulus (a % b), and exponentiation (a ** b). The output shows the results for each operation.

```
a = 46 # Initializing the value of a
b = 4 # Initializing the value of b
print("For a =", a, "and b =", b,"\\nCalculate the following:")
# printing different results
print('1. Addition of two numbers: a + b =', a + b)
print('2. Subtraction of two numbers: a - b =', a - b)
print('3. Multiplication of two numbers: a * b =', a * b)
print('4. Division of two numbers: a / b =', a / b)
print('5. Floor division of two numbers: a // b =', a // b)
print('6. Remainder of two numbers: a mod b =', a % b)
print('7. Exponent of two numbers: a ^ b =', a ** b)
```

Output:

```
Calculate the following:
1. Addition of two numbers: a + b = 50
2. Subtraction of two numbers: a - b = 42
3. Multiplication of two numbers: a * b = 184
4. Division of two numbers: a / b = 11.5
5. Floor division of two numbers: a // b = 11
6. Remainder of two numbers: a mod b = 2
7. Exponent of two numbers: a ^ b = 4477456
```

2) Comparison Operators



A screenshot of a Windows command prompt window titled "practice.py U X Walkthrough Python". The current directory is "C:\Users\user\PycharmProjects\Walkthrough_Python". The command entered is "practice.py >...". The code in practice.py initializes variables a = 46 and b = 4, then prints various comparison operations: equality (a == b), inequality (a != b), less than or equal (a <= b), greater than or equal (a >= b), greater than (a > b), and less than (a < b). The output shows the results for each comparison.

```
a = 46 # Initializing the value of a
b = 4 # Initializing the value of b
print("For a =", a, "and b =", b,"\\nCheck the following:")
# printing different results
print('1. Two numbers are equal or not:', a == b)
print('2. Two numbers are not equal or not:', a != b)
print('3. a is less than or equal to b:', a <= b)
print('4. a is greater than or equal to b:', a >= b)
print('5. a is greater b:', a > b)
print('6. a is less than b:', a < b)
```

Output:

```
For a = 46 and b = 4 8-8835-bccf062c07f7For a =
Check the following:
1. Two numbers are equal or not: False
2. Two numbers are not equal or not: True
3. a is less than or equal to b: False
4. a is greater than or equal to b: True
5. a is greater b: True
6. a is less than b: False
```

3) Assignment Operators

A screenshot of the Visual Studio Code interface. The title bar says "practice [Administrator]". The left sidebar shows a file tree with "practice.py" selected. The main editor area contains the following Python code:

```
a = 34 # Initialize the value of a
b = 6 # Initialize the value of b
# printing the different results
print('a += b:', a + b)
print('a -= b:', a - b)
print('a *= b:', a * b)
print('a /= b:', a / b)
print('a %= b:', a % b)
print('a **= b:', a ** b)
print('a // b:', a // b)
```

Output:

```
a += b: 40
a -= b: 28
a *= b: 284
a /= b: 5.666666666666667
a %= b: 4
a **= b: 1544804416
a // b: 5
```

4) Bitwise Operators

A screenshot of the Visual Studio Code interface. The title bar says "practice [Administrator]". The left sidebar shows a file tree with "practice.py" selected. The main editor area contains the following Python code:

```
a = 7 # initializing the value of a
b = 8 # initializing the value of b
# printing different results
print('a & b :', a & b)
print('a | b :', a | b)
print('a ^ b :', a ^ b)
print('~a :', ~a)
print('a << b :', a << b)
print('a >> b :', a >> b)
```

Output:

```
PS C:\Users\Administrator\Documents\practice> c:; cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter/../debugpy\launcher' '49571' '--' 'C:\Users\Administrator\Documents\practice\practice.py'
x5cadapter/...\\x5cdebugpy\\x5clauncher' '49571' '--' 'C:\\x5cUsers\\x5cAdministrator\\x5cDocuments\\x5cpractice\\x5cpractice.py
a & b : 0      52a6-48f8-8835-bccf062c07f7a & b :
a | b : 15
a ^ b : 15
~a : -8
a << b : 1792
a >> b : 0
```

5) Logical Operators

```
Run ... ← → ⌂ practice [Administrator]
* practice.py U X Walkthrough: Python
* practice.py > ...
1 a = 7 # initializing the value of a
2
3 # printing different results
4 print("For a = 7, checking whether the following conditions are True or False:")
5 print('`a > 5 and a < 7` =>', a > 5 and a < 7)
6 print('`a > 5 or a < 7` =>', a > 5 or a < 7)
7 print(`\`not (a > 5 and a < 7)` =>', not(a > 5 and a < 7))
```

Output:

```
x5cadapter/.../x5cdebugpy/x5clauncher' '49613' '--' 'C:\x5cUsers\x5cAdministrator\x5cDocuments\x5cpractice\x5cpractice.py' ;ec1bef83-52a6-40f8-8835-bccf062c07f7For a = 7, checking whether the following conditions are True or False:
`a > 5 and a < 7` => False
`a > 5 or a < 7` => True
`\`not (a > 5 and a < 7)` => True
```

6) Membership Operators

```
Run ... ← → ⌂ practice [Administrator]
* practice.py U X Walkthrough: Python
* practice.py > ...
1 myList = [12, 22, 28, 35, 42, 49, 54, 65, 92, 103, 245, 874]
2
3 # initializing x and y with some values
4 x = 31
5 y = 28
6
7 # printing the given list
8 print("Given List:", myList)
9
10 # checking if x is present in the list or not
11 if (x not in myList):
12     print("x =", x,"is NOT present in the given list.")
13 else:
14     print("x =", x,"is present in the given list.")
15
16 # checking if y is present in the list or not
17 if (y in myList):
18     print("y =", y,"is present in the given list.")
19 else:
20     print("y =", y,"is NOT present in the given list.")
```

Output:

```
bugpy\x5clauncher' '49721' '--' 'C:\x5cUsers\x5cAdministrator\x5cDocuments\x5cpractice\x5cpractice.py' ;ec1bef83-52a6-40f8-8835-bccf062c07f7
Given List: [12, 22, 28, 35, 42, 49, 54, 65, 92, 103, 245, 874]
x = 31 is NOT present in the given list.
y = 28 is present in the given list.
```

7) Identity Operators

```
Run ... ← → ⌂ practice [Administrator]
* practice.py U X Walkthrough: Python
* practice.py > ...
1 a = ["Rose", "Lotus"]
2 b = ["Rose", "Lotus"]
3
4 # initializing a variable c and storing the value of a in c
5 c = a
6
7 # printing the different results
8 print("a is c => ", a is c)
9 print("a is not c => ", a is not c)
10 print("a is b => ", a is b)
11 print("a is not b => ", a is not b)
12 print("a == b => ", a == b)
13 print("a != b => ", a != b)
```

Output:

```
PS C:\Users\Administrator\Documents\practice> c; cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python310\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.python.debugpy-2024.14.0-win32-x64\bundledlibs\debugpy\adapter/../.debugpylauncher' '49794' '--' 'C:\Users\Administrator\Documents\practice\practice.py'
x5cadapter/.../.x5cdebugpy\x5clauncher' '49794' '--' 'C:\x5cUsers\x5cAdministrator\x5cDocuments\x5cpractice\x5cpractice.p
a is c => True      f8-8835-bccf062c07f7a is c =>
a is not c => False
a is b => False
a is not b => True
a == b => True
a != b => False
PS C:\Users\Administrator\Documents\practice>
```

8) Using for loop

```
File Edit Selection View Go Run ... ⏎ → practice [Administrator]
EXPLORER PRACTICE .vscode practice practice.py
practice.py > ...
1 def reverse_string(str):
2     str1 = "" # Declaring empty string to store the reversed string
3     for i in str:
4         str1 = i + str1
5     return str1 # It will return the reverse string to the caller function
6
7 str = "Javatpoint" # Given String
8 print("The original string is : ",str)
9 print("The reverse string is",reverse_string(str)) # Function call
```

Output:

```
PS C:\Users\Administrator\Documents\practice> c; cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python310\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.python.debugpy-2024.14.0-win32-x64\bundledlibs\debugpy\adapter/../.debugpylauncher' '52924' '--' 'C:\Users\Administrator\Documents\practice\x5cpractice.py'
The original string is: Javatpoint
The reverse string is: tnioptavaJ
PS C:\Users\Administrator\Documents\practice>
```

9) Using while loop

```
File Edit Selection View Go Run ... ⏎ → practice [Administrator]
EXPLORER PRACTICE .vscode practice practice.py
practice.py > ...
1 str = "JavaPoint" # string variable
2 print ("The original string is : ",str)
3 reverse_String = "" # Empty String
4 count = len(str) # Find length of a string and save in count variable
5 while count > 0:
6     reverse_String += str[ count - 1 ] # save the value of str[count-1] in reverseString
7     count = count - 1 # decrement index
8 print ("The reversed string using a while loop is : ",reverse_String)# reversed string
```

Output:

```
PS C:\Users\Administrator\Documents\practice> c; cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python310\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.python.debugpy-2024.14.0-win32-x64\bundledlibs\debugpy\adapter/../.debugpylauncher' '53004' '--' 'C:\Users\Administrator\Documents\practice\practice.py'
The original string is : JavaPoint
The reversed string using a while loop is : tnioptavaJ
PS C:\Users\Administrator\Documents\practice>
```

10) Using the slice ([] operator

File Edit Selection View Go Run ... ← → ⌘ practice [Administrator]

EXPLORER PRACTICE .vscode practice practice.py

```
practice.py > reverse
1 def reverse(str):
2     str = str[::-1]
3     return str
4
5 s = "JavaPoint"
6 print ("The original string is : ",s)
7 print ("The reversed string using extended slice operator is : ",reverse(s))
8
```

Output:

```
Dashter /.../ Debugging\Python\Chapter_5\Practice > practice.py
The original string is : JavaPoint
The reversed string using extended slice operator is : tniopTavaJ
PS C:\Users\Administrator\Documents\practice>
```

11) Using reverse function with join

File Edit Selection View Go Run ... ← → ⌘ practice [Administrator]

EXPLORER PRACTICE .vscode practice practice.py

```
practice.py > reverse
1 def reverse(str):
2     string = "".join(reversed(str)) # reversed() function inside the join() function
3     return string
4
5 s = "JavaPoint"
6
7 print ("The original string is : ",s)
8 print ("The reversed string using reversed() is : ",reverse(s))
```

Output:

```
Dashter /.../ Debugging\Python\Chapter_5\Practice > practice.py
The original string is : JavaPoint
The reversed string using reversed() is : tniopTavaJ
PS C:\Users\Administrator\Documents\practice>
```

12) Using recursion()

File Edit Selection View Go Run ... ← → ⌘ practice [Administrator]

EXPLORER PRACTICE .vscode practice practice.py

```
practice.py > ...
1 def reverse(str):
2     if len(str) == 0: # Checking the length of string
3         return str
4     else:
5         return reverse(str[1:]) + str[0]
6
7 str = "Devansh shams"
8 print ("The original string is : ",str)
9 print ("The reversed string(using recursion) is : ", reverse(str))
```

Output:

```
Dashter /.../ Debugging\Python\Chapter_5\Practice > practice.py
The original string is : Devansh shams
The reversed string(using recursion) is : smahs hsnavED
PS C:\Users\Administrator\Documents\practice>
```

13) To read CSV file in Python

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows a folder named "PRACTICE" containing ".vscode", "practice", and "practice.py".
- Code Editor:** Displays the content of "practice.py":

```
1 import csv
2 # open file by passing the file path.
3 with open('C:\Users\Administrator\Downloads\student_records.csv') as csv_file:
4     csv_read = csv.reader(csv_file, delimiter=',') #Delimiter is comma
5     count_line = 0
6     # Iterate the file object or each row of the file
7     for row in csv_read:
8         if count_line == 0:
9             print(f'Column names are {" ".join(row)}')
10            count_line += 1
11        else:
12            print(f'\t{row[0]} roll number is: {row[1]} and department is: {row[2]}.')
13            count_line += 1
14    print(f'Processed {count_line} lines.') # This line will print number of line fro the file
```
- Terminal:** Shows the output of running the script:

```
Column names are Name, Roll Number, Department
Charlie White roll number is: 105 and department is: Information Technology.
Processed 2 lines.
```

Output:

The terminal window shows the command run and its output:

```
python C:/Users/Administrator/Documents/practice/practice.py
Column names are Name, Roll Number, Department
Charlie White roll number is: 105 and department is: Information Technology.
Processed 2 lines.
```

If-else statements

14) If statement

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows a folder named "PRACTICE" containing ".vscode", "practice", and "practice.py".
- Code Editor:** Displays the content of "practice.py":

```
1 num = int(input("enter the number:"))
2 # Here, we are taking an integer num and taking input dynamically
3 if num%2 == 0:
4     # Here, we are checking the condition. If the condition is true, we will enter the block
5     print("The Given number is an even number")
```

Output:

The terminal window shows the command run and its output:

```
python C:/Users/Administrator/Documents/practice/practice.py
enter the number:20
The Given number is an even number
```

15) Program to print the largest of the three numbers.

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows a folder named "PRACTICE" containing ".vscode", "practice", and "practice.py".
- Code Editor:** Displays the content of "practice.py":

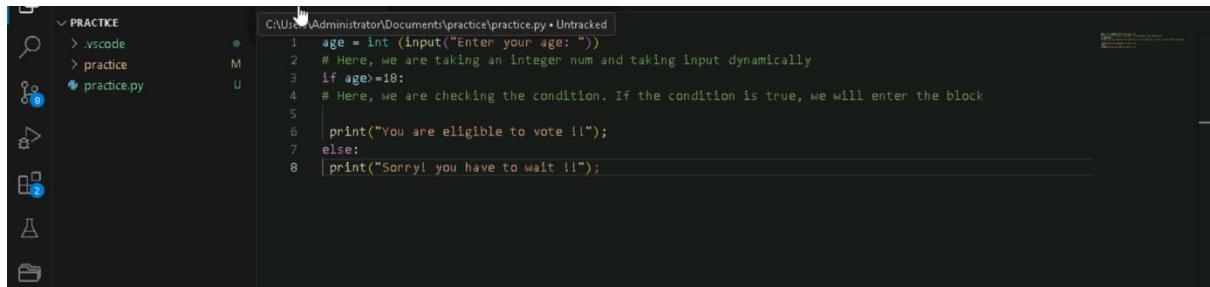
```
1 a = int(input("Enter a: "));
2 b = int (input("Enter b: "));
3 c = int (input("Enter c: "));
4 if a>b and a>c:
5     # Here, we are checking the condition. If the condition is true, we will enter the block
6     print ("From the above three numbers given a is largest");
7     if b>a and b>c:
8         # Here, we are checking the condition. If the condition is true, we will enter the block
9         print ("From the above three numbers given b is largest");
10        if c>a and c>b:
11            # Here, we are checking the condition. If the condition is true, we will enter the block
12            print ("From the above three numbers given c is largest");
```

Output:

The terminal window shows the command run and its output:

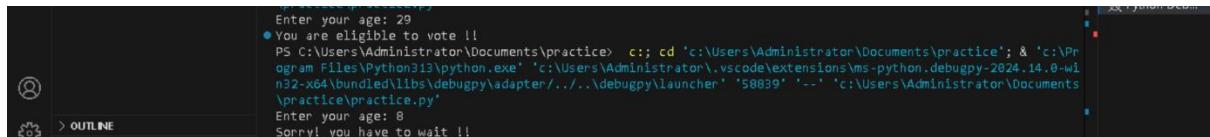
```
python C:/Users/Administrator/Documents/practice/practice.py
Enter a: 20
Enter b: 27
Enter c: 23
From the above three numbers given b is largest
```

16) Program to check whether a person is eligible to vote or not.



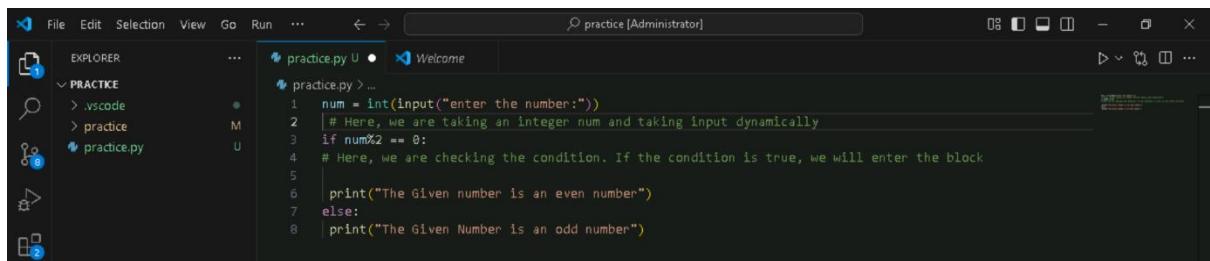
```
C:\Users\Administrator\Documents\practice> practice.py * Untracked
1 age = int(input("Enter your age: "))
2 # Here, we are taking an integer num and taking input dynamically
3 if age>=18:
4     # Here, we are checking the condition. If the condition is true, we will enter the block
5
6     print("You are eligible to vote !!");
7 else:
8     print("Sorry! you have to wait !!");
```

Output:



```
Enter your age: 29
● You are eligible to vote !!
PS C:\Users\Administrator\Documents\practice> c;; cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python310\python.exe' 'c:\Users\Administrator\vscode\extensions\ms-python.debugpy-2024.14.0-wi
n32-x64\bundled\libs\debugpy\adapter\...\\debugpy\launcher' '58839' '---' 'c:\Users\Administrator\Documents\prac
tice\practice.py'
Enter your age: 8
Sorry! you have to wait !!
```

17) Program to check whether a number is even or not.



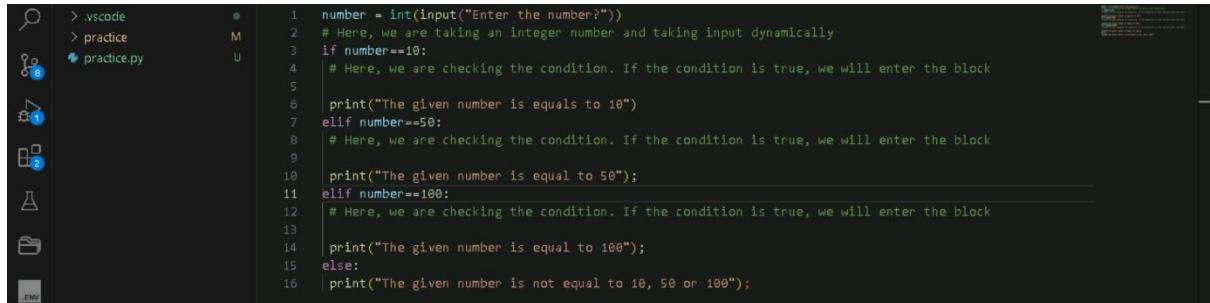
```
File Edit Selection View Go Run ... ← → ○ practice [Administrator]
EXPLORER PRACTICE > .vscode > practice.py > ...
1 num = int(input("enter the number:"))
2 # Here, we are taking an integer num and taking input dynamically
3 if num%2 == 0:
4     # Here, we are checking the condition. If the condition is true, we will enter the block
5
6     print("The Given number is an even number")
7 else:
8     print("The Given Number is an odd number")
```

Output:



```
Enter the number:39
The Given Number is an odd number
```

18) Elif statement



```
> .vscode > practice.py > ...
1 number = int(input("Enter the number?"))
2 # Here, we are taking an integer number and taking input dynamically
3 if number==10:
4     # Here, we are checking the condition. If the condition is true, we will enter the block
5
6     print("The given number is equals to 10")
7 elif number==50:
8     # Here, we are checking the condition. If the condition is true, we will enter the block
9
10    print("The given number is equal to 50");
11 elif number==100:
12     # Here, we are checking the condition. If the condition is true, we will enter the block
13
14    print("The given number is equal to 100");
15 else:
16    print("The given number is not equal to 10, 50 or 100");
```

Output:



```
Enter the number?50
The given number is equal to 50
```

19) Elif statement

```
practice.py >...
1 marks = int(input("Enter the marks? "))
2 if marks > 85 and marks <= 100:
3     # Here, we are checking the condition. If the condition is true, we will enter the block
4
5     print("Congrats! you scored grade A ...")
6 elif marks > 60 and marks <= 85:
7     # Here, we are checking the condition. If the condition is true, we will enter the block
8
9     print("You scored grade B + ...")
10 elif marks > 40 and marks <= 60:
11     # Here, we are checking the condition. If the condition is true, we will enter the block
12
13     print("You scored grade B ...")
14 elif (marks > 30 and marks <= 40):
15     # Here, we are checking the condition. If the condition is true, we will enter the block
16
17     print("You scored grade C ...")
18 else:
```

Output:

```
Enter the marks? 85
You scored grade B + ...
PS C:\Users\Administrator\Documents\practice> c:; cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python310\python.exe' 'c:/Users/Administrator/.vscode/extensions/ms-python.python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter/../..debugpy\launcher' '59071' --- 'c:/Users/Administrator/Documents/practice/practice.py'
x5cadapter/.../..x5cdebugpy\x5clauncher' '59071' --- 'c:/x5cUsers/x5cAdministrator/x5cDocuments/x5cpractice.py' ;d11dc8b3-3ca1-44d6-a22a-2a26ea78d530Enter the marks? 86
Congrats! you scored grade A ...
```

Python Loops

20) For Loop

```
practice.py >...
1 numbers = [4, 2, 6, 7, 3, 5, 8, 10, 6, 1, 9, 2]
2
3 # variable to store the square of the number
4 square = 0
5
6 # Creating an empty list
7 squares = []
8
9 # Creating a for loop
10 for value in numbers:
11     square = value ** 2
12     squares.append(square)
13 print("The list of squares is", squares)
```

Output:

```
The list of squares is [16, 4, 36, 49, 9, 25, 64, 100, 36, 1, 81, 4]
```

21) Using else Statement with for Loop

```
practice.py >...
1 string = "Python Loop"
2
3 # Initiating a loop
4 for s in string:
5     # giving a condition in if block
6     if s == "o":
7         print("If block")
8     # if condition is not satisfied then else block will be executed
9     else:
10        print(s)
```

Output:

```
P
y
t
h
I
f
b
l
o
p
u
n
L
I
f
b
l
o
p
```

22) Using else Statement with for Loop

A screenshot of the Visual Studio Code interface. The Explorer sidebar shows a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'practice.py' file is open in the editor, displaying the following Python code:

```
tuple_ = (3, 4, 6, 8, 9, 2, 3, 8, 9, 7)
# Initiating the loop
for value in tuple_:
    if value % 2 != 0:
        print(value)
    # giving an else statement
else:
    print("These are the odd numbers present in the tuple")
```

Output:

A screenshot of the Visual Studio Code interface. The terminal window shows the output of the 'practice.py' script. The output consists of several lines of text, each starting with 'These are the odd numbers present in the tuple' followed by a list of odd numbers from 3 to 9.

```
These are the odd numbers present in the tuple
These are the odd numbers present in the tuple
These are the odd numbers present in the tuple
9
These are the odd numbers present in the tuple
3
These are the odd numbers present in the tuple
9
7
```

23) Range() Function

A screenshot of the Visual Studio Code interface. The Explorer sidebar shows a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'practice.py' file is open in the editor, displaying the following Python code:

```
print(range(15))
print(list(range(15)))
print(list(range(4, 9)))
print(list(range(5, 25, 4)))
```

Output:

A screenshot of the Visual Studio Code interface. The terminal window shows the output of the 'practice.py' script. The output consists of four lists of integers generated by the range function.

```
range(0, 15)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]
[4, 5, 6, 7, 8]
[5, 9, 13, 17, 21]
```

24) Range() Function

A screenshot of the Visual Studio Code interface. The Explorer sidebar shows a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'practice.py' file is open in the editor, displaying the following Python code:

```
tuple_ = ("Python", "Loops", "Sequence", "Condition", "Range")
# iterating over tuple_ using range() function
for iterator in range(len(tuple_)):
    print(tuple_[iterator].upper())
```

Output:

A screenshot of the Visual Studio Code interface. The terminal window shows the output of the 'practice.py' script. The output consists of five uppercase words: PYTHON, LOOPS, SEQUENCE, CONDITION, and RANGE.

```
PYTHON
LOOPS
SEQUENCE
CONDITION
RANGE
```

25) While Loop

A screenshot of the Visual Studio Code interface. The Explorer sidebar shows a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'practice.py' file is open in the editor, displaying the following Python code:

```
counter = 0
# Initiating the loop
while counter < 10: # giving the condition
    counter = counter + 3
    print("Python Loops")
```

Output:

A screenshot of the Visual Studio Code interface. The terminal window shows the output of the 'practice.py' script. The output consists of three lines of text: 'Python Loops', 'Python Loops', and 'Python Loops'.

```
Python Loops
Python Loops
Python Loops
```

26) Using else Statement with while Loops

A screenshot of the VS Code interface. The Explorer sidebar shows a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'CODE' tab is active, displaying the contents of 'practice.py'. The code is as follows:

```
1 counter = 0
2
3 # Iterating through the while loop
4 while (counter < 10):
5     counter = counter + 3
6     print("Python Loops") # Executed until condition is met    "until": Unknown word.
7     # Once the condition of while loop gives False this statement will be executed
8 else:
9     print("Code block inside the else statement")
```

Output:

A screenshot of the terminal window in VS Code. The output shows the execution of the 'practice.py' script, which prints 'Python Loops' ten times, followed by the message 'Code block inside the else statement'.

```
e:\x5cp\practice.py' ;d11dc8b3-3ca1-44d6-a22a-2a26ea78d530Python Loops
Python Loops
Code block inside the else statement
```

27) Single statement while Block

A screenshot of the VS Code interface. The Explorer sidebar shows a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'CODE' tab is active, displaying the contents of 'practice.py'. The code is as follows:

```
1 for string in "Python Loops":
2     if string == "o" or string == "p" or string == "t":
3         continue
4     print('Current Letter:', string)
```

Output:

A screenshot of the terminal window in VS Code. The output shows the execution of the 'practice.py' script, which prints 'Python Loops' ten times, skipping the letters 'o', 'p', and 't' due to the 'continue' statement.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SPELL CHECKER
+
+ ... ^ x
powershell
Python Loops
```

28) Continue Statement

A screenshot of the VS Code interface. The Explorer sidebar shows a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'CODE' tab is active, displaying the contents of 'practice.py'. The code is as follows:

```
1 for string in "Python Loops":
2     if string == "o" or string == "p" or string == "t":
3         continue
4     print('Current Letter:', string)
```

Output:

A screenshot of the terminal window in VS Code. The output shows the execution of the 'practice.py' script, which prints 'Python Loops' ten times, skipping the letters 'o', 'p', and 't' due to the 'continue' statement.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SPELL CHECKER
+
+ ... ^ x
powershell
program Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.python.debugpy-2024.14.0-wi
n32-x64\bundled\libs\debugpy\adapter\...\\debugpy\launcher' '59498' '--- 'c:\Users\Administrator\Documents
\practice\practice.py'
Current Letter: P
Current Letter: y
Current Letter: h
Current Letter: n
Current Letter:
Current Letter: L
Current Letter: s
```

29) Break Statement

A screenshot of the VS Code interface. The Explorer sidebar shows a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'CODE' tab is active, displaying the contents of 'practice.py'. The code is as follows:

```
1 for string in "Python Loops":
2     if string == 'L':
3         break
4     print('Current Letter: ', string)
```

Output:

The screenshot shows the VS Code interface with the terminal tab active. The output window displays the following text:

```
program Files\Python313\python.exe' 'c:\Users\Administrator\vscode\extensions\ms-python.debugpy-2024.14.0-wi
n32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '59528' '--' 'c:\Users\Administrator\Documents
\practice\practice.py'
Current Letter: P
Current Letter: y
Current Letter: t
Current Letter: h
Current Letter: o
Current Letter: n
Current Letter: :
```

30) Pass Statement

The screenshot shows the VS Code interface with the code editor tab active. The file 'practice.py' contains the following code:

```
pass
print('Last Letter:', string)
```

Output:

The screenshot shows the VS Code interface with the terminal tab active. The output window displays the following text:

```
n32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '59566' '--' 'c:\Users\Administrator\Documents
\practice\practice.py'
Last Letter: :
```

31) for Loop

The screenshot shows the VS Code interface with the code editor tab active. The file 'practice.py' contains the following code:

```
numbers = [3, 5, 23, 6, 5, 1, 2, 9, 8]
# initializing a variable that will store the sum
sum_ = 0
# using for loop to iterate over the list
for num in numbers:
    sum_ = sum_ + num ** 2
print("The sum of squares is: ", sum_)
```

Output:

The screenshot shows the VS Code interface with the terminal tab active. The output window displays the following text:

```
e\5cp\practice.py';d11dc8b3-3ca1-44d6-a22a-2a26ea78d530The sum of squares is: 774
```

32) range() Function

The screenshot shows the VS Code interface with the code editor tab active. The file 'practice.py' contains the following code:

```
my_list = [3, 5, 6, 8, 4]
for iter_var in range( len( my_list ) ):
    my_list.append(my_list[iter_var] + 2)
print( my_list )
```

Output:

The screenshot shows the VS Code interface with the terminal tab active. The output window displays the following text:

```
[3, 5, 6, 8, 4, 5, 7, 8, 10, 6]
```

33) Iterating by Using Index of Sequence

The screenshot shows the VS Code interface with the code editor tab active. The file 'practice.py' contains the following code:

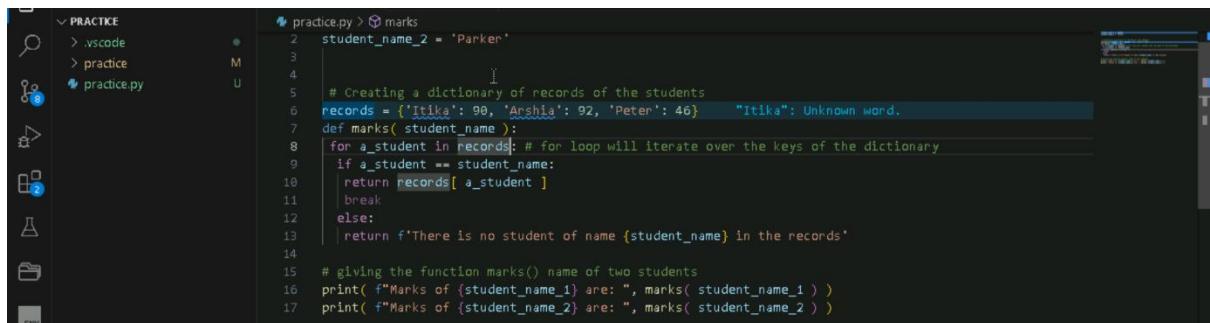
```
numbers = [3, 5, 23, 6, 5, 1, 2, 9, 8]
# initializing a variable that will store the sum
sum_ = 0
# using for loop to iterate over list
for num in range( len(numbers) ):
    sum_ = sum_ + numbers[num] ** 2
print("The sum of squares is: ", sum_)
```

Output:

The screenshot shows the VS Code interface with the terminal tab active. The output window displays the following text:

```
e\5cp\practice.py';d11dc8b3-3ca1-44d6-a22a-2a26ea78d530The sum of squares is: 774
```

34) Using else Statement with for Loop



```
practice.py > marks
1 student_name_2 = 'Parker'
2
3
4
5 # Creating a dictionary of records of the students
6 records = {'Itika': 90, 'Arshia': 92, 'Peter': 46} "Itika": Unknown word.
7 def marks( student_name ):
8     for a_student in records: # for loop will iterate over the keys of the dictionary
9         if a_student == student_name:
10             return records[ a_student ]
11         break
12     else:
13         return f'There is no student of name {student_name} in the records'
14
15 # giving the function marks() name of two students
16 print( f'Marks of {student_name_1} are: ', marks( student_name_1 ) )
17 print( f'Marks of {student_name_2} are: ', marks( student_name_2 ) )
```

Output:



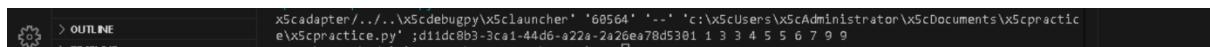
```
Marks of Itika are: 90
Marks of Parker are: There is no student of name Parker in the records
```

35) Nested Loops



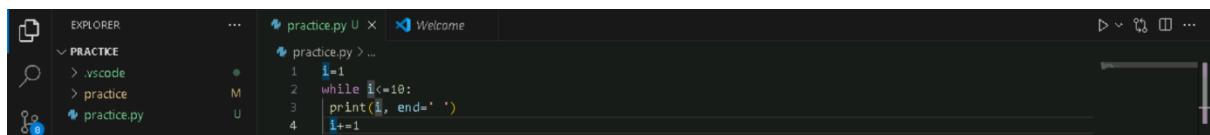
```
practice.py > ...
1 import random
2 numbers = [ ]
3 for val in range(0, 11):
4     numbers.append( random.randint( 0, 11 ) )
5 for num in range( 0, 11 ):
6     for i in numbers:
7         if num == i:
8             print( num, end = " " )
```

Output:



```
x5cadapter/... .\x5cdebugpy\x5clauncher' '60564' ... 'c:\x5cUsers\x5cAdministrator\x5cDocuments\x5cpractice.py';d11dc8b3-3ca1-44d6-a22a-2426ea78d5301 1 3 3 4 5 5 6 7 9 9
```

36) Printing numbers from 1 to 10



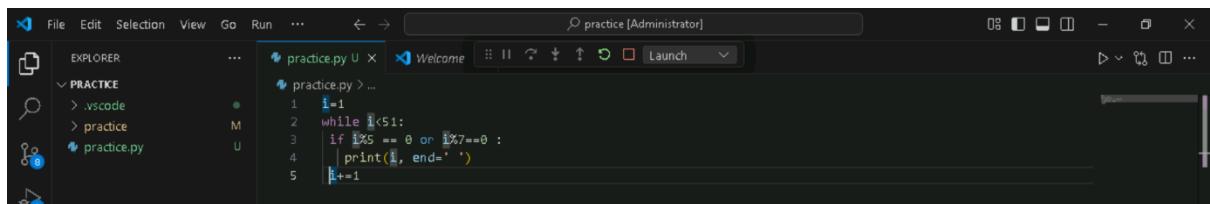
```
EXPLORER          practice.py U X  Welcome
PRACTICE
> .vscode
> practice
> practice.py
1 i=1
2 while i<=10:
3     print(i, end=' ')
4 i+=1
```

Output:



```
1 2 3 4 5 6 7 8 9 10
```

37) Printing those numbers divisible by either 5 or 7 within 1 to 50 using a while loop.



```
File Edit Selection View Go Run ... ← → ⌂ practice [Administrator]
EXPLORER          practice.py U X  Welcome
PRACTICE
> .vscode
> practice
> practice.py
1 i=1
2 while i<51:
3     if i%5 == 0 or i%7==0 :
4         print(i, end=' ')
5     i+=1
```

Output:



```
5 7 10 14 15 20 21 25 28 30 35 40 42 45 49 50
```

38) The sum of squares of the first 15 natural numbers using a while loop.

```
File Edit Selection View Go Run ... ← → ⌂ practice [Administrator]
EXPLORER PRACTICE .vscode practice practice.py
practice.py > ...
1 num = 15
2
3 # initializing summation and a counter for iteration
4 summation = 0
5 c = 1
6
7 while c <= num: # specifying the condition of the loop
8     # beginning the code block
9     summation = c**2 + summation
10    c = c + 1 # incrementing the counter
11
12 # print the final sum |
13 print("The sum of squares is", summation)
```

Output:

```
The sum of squares is 1240
```

39) While loops in Python for a number is Prime number or not.

```
File Edit Selection View Go Run ... ← → ⌂ practice [Administrator]
EXPLORER PRACTICE .vscode practice practice.py
practice.py > prime_number
C:\Users\Administrator\Documents\practice\practice.py • Untracked
2
3 def prime_number(number):
4     condition = 0
5     iteration = 2
6     while iteration <= number / 2:
7         if number % iteration == 0:
8             condition = 1
9             break
10            iteration = iteration + 1
11
12 if condition == 0:
13     print(f"{number} is a PRIME number")
14 else:
15     print(f"{number} is not a PRIME number")
16 for i in num:
17     prime_number(i)
```

Output:

```
12 is not a PRIME number
54 is not a PRIME number
23 is a PRIME number
75 is not a PRIME number
34 is not a PRIME number
11 is a PRIME number
```

40) Armstrong and Python While Loop

```
File Edit Selection View Go Run ... ← → ⌂ practice [Administrator]
EXPLORER PRACTICE .vscode practice practice.py
practice.py > run
1 n=int(input())
2 n1=str(n)
3 l=len(n1)
4 temp=n
5 s=0
6 while n1!=0:
7     r=n%10
8     s+=r**l
9     n=n//10
10    if s==temp:
11        print("it is an Armstrong number")
12    else:
13        print("it is not an Armstrong number ")
```

Output:

```
153
It is an Armstrong number
```

41) Multiplication Table using While Loop

```
practice.py U X Welcome
practice.py > ...
1 num = 21
2 counter = 1
3 # we will use a while loop for iterating 10 times for the multiplication table
4 print("The Multiplication Table of: ", num)
5 while counter <= 10: # specifying the condition
6     ans = num * counter
7     print( num, 'x', counter, '=', ans)
8     counter += 1 # expression to increment the counter
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SPELL CHECKER
21 x 1 = 21
21 x 2 = 42
21 x 3 = 63
21 x 4 = 84
21 x 5 = 105
21 x 6 = 126
21 x 7 = 147
21 x 8 = 168
21 x 9 = 189
21 x 10 = 210
```

42) Python While Loop with List

```
PRACTICE
practice.py > ...
C:\User\Administrator\Documents\practice\practice.py • Untracked
1 squares = []
2 # programming a while loop      "programming": Forbidden word.
3 while list_ : # until list is not empty this expression will give boolean True after that False
4     squares.append( (list_.pop())**2 )
5 # Print the squares of all numbers.
6
7 print( squares )
```

Output:

```
[36, 16, 1, 25, 9]
```

43) Determine odd and even number

```
File Edit Selection View Go Run ... ← → practice [Administrator]
PRACTICE
practice.py U X Welcome
practice.py > ...
1 index = 0
2 while index < len(list_):
3     element = list_[index]
4     if element % 2 == 0:
5         print('It is an even number') # Print if the number is even.
6     else:
7         print('It is an odd number') # Print if the number is odd.
8     index += 1
```

Output:

```
e:\x5cpractice.py' ;d11dc8b3-3ca1-44d6-a22a-2a26ea78d530It is an odd number
It is an even number
It is an odd number
It is an odd number
It is an even number
```

44) Determine the number letter

```
PRACTICE
practice.py > ...
1 List_= ['Priya', 'Neha', 'Cow', 'To']    "Priya": Unknown word.
2 index = 0
3 while index < len(List_):
4     element = List_[index]
5     print(len(element))
6     index += 1
```

Output:

```
5
4
3
2
```

45) While Loop Multiple Conditions

```
1 num1 = 17
2 num2 = -12
3
4 while num1 > 5 and num2 < -5 : # multiple conditions in a single while loop
5 | num1 -= 2
6 | num2 += 3
7 | print( (num1, num2) )
```

Output:

```
(15, -9)
(13, -6)
(11, -3)
```

46) While Loop Multiple Conditions

```
1 num1 = 17
2 num2 = -12
3
4 while num1 > 5 or num2 < -5 :
5 | num1 -= 2
6 | num2 += 3
7 | print( (num1, num2) )
```

Output:

```
(15, -9)
(13, -6)
(11, -3)
(9, 0)
(7, 3)
(5, 6)
```

47) While Loop Multiple Conditions

```
1 num1 = 9
2 num = 14
3 maximum_value = 4
4 counter = 0
5 while (counter < num1 or counter < num2) and not counter >= maximum_value: # grouping multiple conditions
6 | print(f"Number of iterations: {counter}")
7 | counter += 1
```

Output:

```
e:\x5cpractice.py' ;d1dc8b3-3ca1-44d6-a22a-2a26ea78d530Number of iterations: 0
Number of iterations: 1
Number of iterations: 2
Number of iterations: 3
```

Activate Windows
Go to Settings to activate Windows.

48) Single Statement While Loop

```
1 counter = 1
2 while counter: print('Python While Loops')
```

Output:

```
Python While Loops
```

Activate Windows
Go to Settings to activate Windows.

49) Loop Control Statements

```
for string in "While Loops":
    if string == "a" or string == "i" or string == "e":
        continue
    print('Current Letter:', string)
```

Output:

```
program Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter/../debugpy\launcher' '61614' '--' 'c:\Users\Administrator\Documents\practice\practice.py'
Current Letter: W
Current Letter: h
Current Letter: l
Current Letter:
Current Letter: L
Current Letter: p
Current Letter: s
```

Activate Windows
Go to Settings to activate Windows.

50) Break Statement

```
for string in "Python Loops":
    if string == 'n':
        break
    print('Current Letter: ', string)
```

Output:

```
python practice.py
Scadapters/...\\x5cdebugpy\\x5clauncher' '50601' '--' 'c:\\x5cUsers\\x5cAdministrator\\x5cDocuments\\x5cpractice\\x5cpрактиce.py' ;5f018326-7814-4700-ac48-33a739a6869c
Current Letter: P
Current Letter: y
Current Letter: t
Current Letter: h
Current Letter: o
```

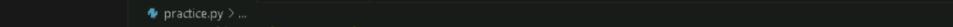
51) Pass Statement

```
for string in "Python Loops":
    pass
print('The Last Letter of given string is:', string)
```

Output:

```
PS C:\Users\Administrator\Documents\practice> & 'c:\\Program Files\\Python313\\python.exe' 'c:\\Users\\Administrator\\.vscode\\extensions\\ms-python.debugpy-2024.14.0-win32-x64\\bundled\\libs\\debugpy\\adapter/..\\..\\debugpy\\launcher' '50679' '--' 'c:\\Users\\Administrator\\Documents\\practice\\practice.py'
The Last Letter of given string is: s
PS C:\\Users\\Administrator\\Documents\\practice>
```

52) break statement with for loop



```
practice.py > ...
1 my_list = [1, 2, 3, 4]
2 count = 1
3 for item in my_list:
4     if item == 4:
5         print("Item matched")
6         count += 1
7         break
8     print("Found at location", count)
9
```

Output:

The screenshot shows the VS Code interface with the Python Debug Console tab selected. The output window displays the following text:

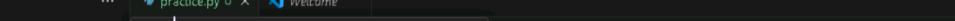
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GIT LENS SPELL CHECKER
```

Python Debug Console + ✎ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂

```
Found at location 1
Found at location 1
● Item matched
PS C:\Users\Administrator\Documents\practice> c;; cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python310\python.exe' 'c:\Users\Administrator\.vscodeextensions\ms-python.python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter...'...'debugpy\launcher' '50757' '--' 'c:\Users\Administrator\Documents\practice\practice.py'
Found at location 1
Found at location 1
Found at location 1
Item matched
```

On the left sidebar, there are icons for File, Edit, Search, and others, along with sections for OUTLINE and TERMINAL.

53) Breaking out of a loop early

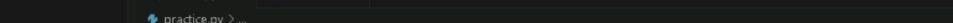


A screenshot of the Visual Studio Code interface. The title bar shows 'practice.py U' and 'Welcome'. The left sidebar has icons for Explorer, Search, and Problems, with 'PRACTICE' expanded. The main editor area shows a Python script:

```
CAUser\Documents\practice\practice.py * Untracked
1 my_str = "python"
2 for char in my_str:
3     if char == 'o':
4         break
5     print(char)
6
```

Output:

54) break statement with while loop



The screenshot shows the VS Code interface with the following details:

- Explorer View:** Shows a folder named "PRACTICE" containing ".vscode" and "practice". A file named "practice.py" is currently selected.
- Code Editor:** Displays the following Python code:

```
i = 0;
while i < 10:
    print(i, ",end="")
    i+=1;
    if i == 10:
        break;
print("came out of while loop");
```
- Status Bar:** Shows "10 lines" and "100%".

Output:

55) break statement with nested loops

```
practice.py U X Welcome
practice.py > ...
1 n = 2
2 while True:
3     i = 1
4     while i <= 10:
5         print("%d X %d = %d\n" % (n, i, n * i))
6         i += 1
7     choice = int(input("Do you want to continue printing the table? Press 0 for no: "))
8     if choice == 0:
9         print("Exiting the program...")
10        break
11    n += 1
12 print("Program finished successfully.")
```

Output:

```
2 X 1 = 2
2 X 2 = 4
2 X 3 = 6
2 X 4 = 8
2 X 5 = 10
2 X 6 = 12
2 X 7 = 14
2 X 8 = 16
2 X 9 = 18
2 X 10 = 20
Do you want to continue printing the table? Press 0 for no:
```

56) Continue Statements in for Loop

```
practice.py U X Welcome
practice.py > ...
1 for iterator in range(10, 21):
2
3     # If iterator is equals to 15, loop will continue to the next iteration
4     if iterator == 15:
5         continue
6     # otherwise printing the value of iterator
7     print(iterator)
8
```

Output:

```
10
11
12
13
14
15
16
17
18
19
20
```

57) Continue Statements in while Loop

```
practice.py U X Welcome
practice.py > ...
4
5     # starting a while loop
6     while iterator < len(string):
7         # if loop is at letter a it will skip the remaining code and go to next iteration
8         if string[iterator] == 'a':
9             iterator += 1
10            continue
11         # otherwise it will print the letter
12         print(string[iterator])
13         iterator += 1
14
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SPELL CHECKER + ⌂ ... ⌂ x
rator\vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '51040' ... 'c:\Users\Administrator\Documents\practice\practice.py'
Administrator\x5cDocuments\x5cpractice\x5cpractice.py' ;e66cd539-ffd-410b-84b3-d0bf84e40e90]
v
T
p
o
i
n
t
```

```
[4, 16, 36, 64, 100]
```

58) Continue statement in list comprehension

```
EXPLORER ... practice.py U × Welcome ⌂ ... ⌂ x
practice.py > ...
1 numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
2
3 # Using a list comprehension with continue
4 sq_num = [num ** 2 for num in numbers if num % 2 == 0]
5 # This will skip odd numbers and only square the even numbers
6 print(sq_num)
7
```

Output:

```
EXPLORER ... OUTLINE ... ⌂ ... ⌂ x
TIMELINE ... PS C:\Users\Administrator\Documents\practice>
```

```
[4, 16, 36, 64, 100]
```

Python String

59) Creating String in Python

```
File Edit Selection View Go Run ... ⌂ ... ⌂ x
EXPLORER ... practice.py U × Welcome ... ⌂ ... ⌂ x
practice.py > ...
1 str1 = 'Hello Python'
2 print(str1)
3 #Using double quotes
4 str2 = "Hello Python"
5 print(str2)
6 #Using triple quotes
7 str3 = """Triple quotes are generally used for
8 represent the multiline or docstring...
9
10 print(str3)
```

Output:

```
e\x5cpractice.py' ;e0e54214-3642-4a80-8626-5056f910d47dHello Python
Hello Python
'''Triple quotes are generally used for
represent the multiline or docstring...
PS C:\Users\Administrator\Documents\practice>
```

60) Strings indexing and splitting

```
File Edit Selection View Go Run ... ⌂ ... ⌂ x
EXPLORER ... practice.py U × Welcome ... ⌂ ... ⌂ x
practice.py > ⌂ tuple_
1 list_ = ["Python", "Lists", "Tuples", "Differences"]
2 tuple_ = ("Python", "Lists", "Tuples", "Differences")
3 # printing sizes
4 print("Size of tuple: ", tuple_.__sizeof__())
5 print("Size of list: ", list_.__sizeof__())
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SPELL CHECKER + ⌂ ... ⌂ x
program Files\Python313\python.exe' 'c:\Users\Administrator\vscode\extensions\ms-python.debugpy-2024.14.0-wi
n32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56449' ... 'c:\Users\Administrator\Documents
\practice\practice.py'
['Python', 'Lists', 'Tuples', 'Mutable']
Tuples cannot be modified because they are immutable
```

61) Strings indexing and splitting

The screenshot shows the VS Code interface with a dark theme. The left sidebar has a tree view with 'PRACTICE' expanded, showing '.vscode', 'practice', and 'practice.py'. The main editor window displays the following Python code:

```
1 str = "JAVATPOINT"
2 # Start 0th index to end
3 print(str[0:])
4 # Starts 1th index to 4th index
5 print(str[1:5])
6 # Starts 2nd index to 3rd index
7 print(str[2:4])
8 # Starts 0th to 2nd index
9 print(str[:3])
10 #Starts 4th to 6th index
11 print(str[4:7])
12
```

The terminal tab shows the output of running the script:

```
PS C:\Users\Administrator\Documents\practice> c: cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python310\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter/../..\\debugpy\launcher' '55225' --- 'c:\Users\Administrator\Documents\practice\practice.py'
JAVATPOINT
AVAT
VA
JAV
TPO
```

Output:

The screenshot shows the VS Code interface with a dark theme. The left sidebar has a tree view with 'PRACTICE' expanded, showing '.vscode', 'practice', and 'practice.py'. The main editor window displays the same Python code as before.

The terminal tab shows the output of running the script, including an error message:

```
PS C:\Users\Administrator\Documents\practice> c: cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python310\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter/../..\\debugpy\launcher' '55225' --- 'c:\Users\Administrator\Documents\practice\practice.py'
JAVATPOINT
AVAT
VA
JAV
TPO
● IndexError: string index out of range
PS C:\Users\Administrator\Documents\practice> c: cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python310\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter/../..\\debugpy\launcher' '55225' --- 'c:\Users\Administrator\Documents\practice\practice.py'
```

62) Strings indexing and splitting

The screenshot shows a Windows desktop environment with the Visual Studio Code (VS Code) application open. The code editor displays a Python file named `practice.py` containing the following code:

```
1 str = 'JAVATPOINT'
2 print(str[-1])
3 print(str[-3])
4 print(str[-2:])
5 print(str[-4:-1])
6 print(str[-7:-2])
7 # Reversing the given string
8 print(str[::-1])
9 print(str[-12])
```

The terminal tab is active, showing the output of the script. The output starts with the string "JAVATPOINT" followed by several blank lines. Then it shows a traceback and an `IndexError`:

```
T
I
N
OIN
ATPOI
TNIOPTAVAJ
Traceback (most recent call last):
  File "<:Users\Administrator\Documents\practice\practice.py", line 9, in <module>
    print(str[-12])
               ^
IndexError: string index out of range
```

The status bar at the bottom indicates the file is 178 bytes long and was last modified on 06-02-2025 at 17:40.

Output:

This screenshot shows the same setup as the previous one, but the code has been modified. The `print(str[-12])` line has been removed. The terminal output now correctly prints the reversed string "TAVAJOPTINAVATVAJ".

```
T
I
N
OIN
ATPOI
TNIOPTAVAJ
```

63) Reassigning Strings

A screenshot of the Visual Studio Code interface. The left sidebar shows a project named 'PRACTICE' with files '.vscode', 'practice', and 'practice.py'. The 'TERMINAL' tab is active, displaying a command-line session. The code editor shows a Python script with the following content:

```
1 str = "HELLO"
2 str[0] = "h"
3 print(str)
```

The terminal output shows two errors:

```
IndexError: string index out of range
PS C:\Users\Administrator\Documents\practice> c: cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '55868' --- 'c:\Users\Administrator\Documents\practice\practice.py'
\x5cadapter/.../\x5cdebugpy\x5clauncher' '55868' --- 'c:\x5cUsers\x5cAdministrator\x5cDocuments\x5cpractice\practice.py' ;e0e54214-3642-4488-8626-5856f910d47dTraceback (most recent call last):
  File "c:\Users\Administrator\Documents\practice\practice.py", line 2, in <module>
    str[0] = "h"
~~~^~~~~
TypeError: 'str' object does not support item assignment
```

Output:

A second screenshot of the Visual Studio Code interface, identical to the first one. It shows the same project structure, code editor content, and terminal output with the same two errors.

64) Reassigning Strings

A screenshot of the Visual Studio Code interface. The left sidebar shows a project named 'PRACTICE' with files '.vscode', 'practice', and 'practice.py'. The 'practice.py' file is open in the editor, containing the following code:

```
1 str = "HELLO"
2 print(str)
3 str = "hello"
4 print(str)
```

The terminal tab shows the output of running the script:

```
\practice\practice.py
Traceback (most recent call last):
  File "C:/Users/Administrator/Documents/practice/practice.py", line 2, in <module>
    str[0] = "h"
               ^
● TypeError: 'str' object does not support item assignment
```

The status bar at the bottom indicates the file is 56 bytes long and was last saved on 06-02-2025 at 18:08.

Output:

A screenshot of the Visual Studio Code interface, identical to the previous one except for the terminal output. The terminal now shows:

```
\practice\practice.py
IndexError: string index out of range
PS C:/Users/Administrator/Documents/practice> c: cd 'c:/Users/Administrator/Documents/practice'; & 'c:\Program Files\Python313\python.exe' 'c:/Users/Administrator/.vscode/extensions/ms-python.python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter/../..\\debugpy\\launcher' '55003' '' 'c:/Users/Administrator/Documents/practice/practice.py'
x5cadapter/.../..\\x5cdebugpy\\x5clauncher' '55006' '' 'c:\\x5cUsers\\x5cAdministrator\\x5cDocuments\\x5cpractice\\x5cpractice.py' ;e0e054214-3642-4488-8626-5856f910d47dTraceback (most recent call last):
  File "C:/Users/Administrator/Documents/practice/practice.py", line 2, in <module>
    str[0] = "h"
               ^
● IndexError: string index out of range
```

The status bar at the bottom indicates the file is 42 bytes long and was last saved on 06-02-2025 at 18:05.

65) Deleting the String

The screenshot shows the VS Code interface with a dark theme. The Explorer sidebar on the left lists a project folder 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'TERMINAL' tab is active, displaying a PowerShell session. The command `cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python310\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter/../debugpy\launcher' '55953' '---' 'c:\Users\Administrator\Documents\practice\practice.py'` is run, followed by a traceback:

```
Traceback (most recent call last):
  File "c:\Users\Administrator\Documents\practice\practice.py", line 2, in <module>
    del str[1]
TypeError: 'str' object doesn't support item deletion
```

The status bar at the bottom indicates the file is 31 bytes long and was last modified on 06-02-2025 at 18:08.

Output:

This screenshot is identical to the one above, showing the same code editor state and terminal output. The difference is in the status bar, which now shows the file is 0 bytes long and was last modified on 06-02-2025 at 18:08.

66) Python operators.

A screenshot of the Visual Studio Code (VS Code) interface. The left sidebar shows a project structure with a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The main editor area displays the following Python code:

```
1 str1 = "JAVATPOINT"
2 del str1
3 print(str1)
```

The code has two errors: 'str1' is undefined (Type Error) and 'del' is used on a string (Name Error). The terminal below shows the command run and the resulting error messages:

```
PS C:\Users\Administrator\Documents\practice> c: cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter/../..\\debugpy\launcher' '55978' '--' 'c:\Users\Administrator\Documents\practice\practice.py'
x5cadapter/.../.\\x5cdebugpy\\x5clauncher' '55978' '--' 'c:\\x5cUsers\\x5cAdministrator\\x5cDocuments\\x5cpractice\\x5cpractice.py' ;e0e54214-3642-4488-8626-5856f910d47dTraceback (most recent call last):
  File "c:\\Users\\Administrator\\Documents\\practice\\practice.py", line 3, in <module>
    print(str1)
           ^
TypeError: 'str' object doesn't support item deletion
NameError: name 'str1' is not defined. Did you mean: 'str'?>
```

Output:

A screenshot of the Visual Studio Code (VS Code) interface, identical to the previous one but with syntax highlighting. The code is the same:

```
1 str1 = "JAVATPOINT"
2 del str1
3 print(str1)
```

The 'str1' variable is highlighted in red, and the 'del' keyword is highlighted in blue, indicating they are undefined or being used incorrectly.

67) Python String Formatting

The screenshot shows a Windows desktop environment with the Visual Studio Code application open. The title bar reads "practice [Administrator]". The left sidebar (EXPLORER) shows a folder structure with "PRACTICE" expanded, containing "vscode", "practice", and "practice.py". The main editor area displays the following Python code:

```
practice.py U x Welcome
practice.py > ...
1 str = "Hello"
2 str1 = " world"
3 print(str*3) # prints HelloHelloHello
4 print(str+str1) # prints Hello world
5 print(str[4]) # prints o
6 print(str[2:4]); # prints ll
7 print('w' in str) # prints false as w is not present in str
8 print('wo' not in str1) # prints false as wo is present in str1.
9 print(r'C://python37') # prints C://python37 as it is written
10 print("The string str : %s"%str) # prints The string str : Hello
```

The bottom status bar shows the file is 430 bytes long and the current line and column are Ln 10, Col 1. The terminal tab shows the following output:

```
●ogram Files\Python313\python.exe` 'c:\Users\Administrator\.vscode\extensions\ms-python.python.debugpy-2024.14.0-wi
n32-x64\bundled\libs\debugpy\adapter'...'..\..\debugpy\launcher' '56047' '' 'c:\Users\Administrator\Documents
\practice\practice.py'
x5cadapter'...'x5cdebugpy\x5clauncher' '56047' '' 'c:\x5cUsers\x5cAdministrator\x5cDocuments\x5cpractic
e\x5cpractice.py';e0e54214-3642-4a88-8626-5856f910d47dHelloHelloHello
Hello world
o
ll
False
False
C://python37
The string str : Hello
○ PS C:\Users\Administrator\Documents\practice>
```

The bottom right corner shows the date and time as 06-02-2025.

Output:

The screenshot shows a Windows desktop environment with the Visual Studio Code (VS Code) application open. The title bar reads "practice [Administrator]". The left sidebar (EXPLORER) shows a project structure with files like ".vscode", "practice", and "practice.py". The main editor area displays a Python script named "practice.py" containing the following code:

```
str = "Hello"
str1 = " world"
print(str*3) # prints HelloHelloHello
print(str+str1) # prints Hello world
print(str[4]) # prints o
print(str[2:4]) # prints ll
print('w' in str) # prints false as w is not present in str
print('wo' not in str) # prints false as wo is present in str.
print(r'C://python37') # prints C://python37 as it is written
print("The string str : %s"(str)) # prints The string str : Hello
```

The terminal below shows the execution of the script and its output:

```
●ogram Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-w1n32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '56047' '' 'c:\Users\Administrator\Documents\practice\practice.py'
x5cadapter/.../x5cdebugpy\x5clauncher' '56047' '' 'c:\x5cUsers\x5cAdministrator\x5cDocuments\x5cpractice\x5cpractice.py';oe0e54214-3642-4a88-8626-5856f910d47dHelloHelloHello
Hello world
o
ll
False
False
C://python37
The string str : Hello
○ PS C:\Users\Administrator\Documents\practice>
```

The status bar at the bottom indicates the file is "Ln 10, Col 1", has "430 bytes", and is in "Git Graph" mode. The bottom right corner shows the date and time as "06-02-2025 18:13".

68) Python String Formatting

A screenshot of the Visual Studio Code (VS Code) interface. The left sidebar shows a project named 'PRACTICE' with files '.vscode', 'practice', and 'practice.py'. The main editor window displays the file 'practice.py' with the following code:

```
1 str = "They said, "Hello what's going on?"" Statements must be separated by newlines or semicolons
2 print(str)
```

The second line contains a syntax error: an unterminated string literal. The status bar at the bottom indicates the file has 57 bytes.

Output:

A screenshot of the Visual Studio Code (VS Code) interface, identical to the previous one. The left sidebar shows the 'PRACTICE' project with 'practice.py' selected. The main editor window shows the same code with the same syntax error. The status bar at the bottom indicates the file has 57 bytes.

69) format() method

A screenshot of the Visual Studio Code (VS Code) interface. The left sidebar shows a project structure with a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'practice.py' file is selected. The main editor area contains the following Python code:

```
1 print(''''They said, "What's there?'''')
2 |
3 # escaping single quotes
4 print('They said, "What\'s going on?"')
5 |
6 # escaping double quotes
7 print("They said, \"What's going on?\\"")
```

The terminal tab at the bottom shows the output of running the script:

```
\practice\practice.py
File "c:/Users/Administrator/Documents/practice\practice.py", line 1
    str = "They said, "Hello what's going on?""
                                         ^
● SyntaxError: unterminated string literal (detected at line 1)
PS C:/Users/Administrator/Documents/practice> c; cd 'c:/Users/Administrator/Documents/practice'; & 'c:\Program Files\Python313\python.exe' 'c:/Users/Administrator/.vscode/extensions/ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter/../debugpy\launcher' '56127' --- 'c:/Users/Administrator/Documents/practice\practice.py'
''They said, "What's there?"
They said, "What's going on?"
They said, "What's going on?"
```

Output:

A second screenshot of the VS Code interface, identical to the first one. It shows the same code in 'practice.py' and the same error message in the terminal. The code and error output are identical to the first screenshot.

70) format() method

The screenshot shows the VS Code interface with a dark theme. The Explorer sidebar on the left lists a project folder 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'practice.py' file is selected and open in the main editor area. The code prints various strings and demonstrates string slicing and membership testing:

```
1 str = "Hello"
2 str1 = " world"
3 print(str*3) # prints HelloHelloHello
4 print(str+str1) # prints Hello world
5 print(str[4]) # prints o
6 print(str[2:4]) # prints ll
7 print('w' in str) # prints false as w is not present in str
8 print('wo' not in str) # prints false as wo is present in str.
9 print(r'C://python37') # prints C://python37 as it is written
10 print("The string str : %%(str)) # prints The string str : Hello
```

The terminal below the editor shows the output of running the script:

```
Hello world
o
ll
False
False
C://python37
The string str : Hello
```

Output:

This screenshot is identical to the one above, showing the same VS Code interface, code in 'practice.py', and output in the terminal. The output again shows 'Hello world', 'o', 'll', 'False', 'False', and 'C://python37' followed by the string 'The string str : Hello'.

71) Formatting Using % Operator

A screenshot of the Visual Studio Code interface. The left sidebar shows a project named 'PRACTICE' with files '.vscode', 'practice', and 'practice.py'. The main editor window displays the following Python code:

```
1 print("{} and {} both are the best friend".format("Devansh","Abhishek")) "Abhishek": Unknown word.
2
3 #Positional Argument
4 print("{} and {} best players ".format("Virat","Rohit")) "Virat": Unknown word.
5
6 #Keyword Argument
7 print("{a},{b},{c}".format(a = "James", b = "Peter", c = "Ricky"))
```

The terminal below shows the command-line output of running the script:

```
PS C:\Users\Administrator\Documents\practice> python practice.py
Rohit and Virat best players
James,Peter,Ricky
PS C:\Users\Administrator\Documents\practice>
```

Output:

A screenshot of the Visual Studio Code interface, identical to the one above, showing the same code and terminal output. The terminal output is identical, displaying:

```
PS C:\Users\Administrator\Documents\practice> python practice.py
Rohit and Virat best players
James,Peter,Ricky
PS C:\Users\Administrator\Documents\practice>
```

72) Formatting Using % Operator

The screenshot shows the Visual Studio Code interface. In the Explorer sidebar, there is a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'practice.py' file is selected. The code editor displays the following Python script:

```
1 Integer = 10;
2 Float = 1.290
3 String = "Devansh"
4 print("Hi I am Integer ... My value is %d\nHi I am float ... My value is %f\nHi I am string ... My value is %s")
```

The terminal tab shows the execution of the script:

```
PS C:\Users\Administrator\Documents\practice> c: cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56289' '---' 'c:\Users\Administrator\Documents\practice\practice.py'
>practice<practice.py'
Devansh and Abhishek both are the best friend
Rohit and Virat best players
James,Peter,Ricky
PS C:\Users\Administrator\Documents\practice> c: cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56289' '---' 'c:\Users\Administrator\Documents\practice\practice.py'
>x5cadapter/.../\x5cdebugpy\x5clauncher' '56289' '---' 'c:\x5cUsers\x5cAdministrator\x5cDocuments\x5cpractice\x5cpractice.py' ;ee0e54214-3642-4a88-8626-5856f910d47dHi I am Integer ... My value is 10
Hi I am float ... My value is 1.290000
Hi I am string ... My value is Devansh
○ PS C:\Users\Administrator\Documents\practice>
```

Output:

The screenshot shows the Visual Studio Code interface. In the Explorer sidebar, there is a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'practice.py' file is selected. The code editor displays the same Python script as the previous screenshot.

The terminal tab shows the execution of the script, identical to the previous one:

```
PS C:\Users\Administrator\Documents\practice> c: cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56289' '---' 'c:\Users\Administrator\Documents\practice\practice.py'
>practice<practice.py'
Devansh and Abhishek both are the best friend
Rohit and Virat best players
James,Peter,Ricky
PS C:\Users\Administrator\Documents\practice> c: cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Program Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56289' '---' 'c:\Users\Administrator\Documents\practice\practice.py'
>x5cadapter/.../\x5cdebugpy\x5clauncher' '56289' '---' 'c:\x5cUsers\x5cAdministrator\x5cDocuments\x5cpractice\x5cpractice.py' ;ee0e54214-3642-4a88-8626-5856f910d47dHi I am Integer ... My value is 10
Hi I am float ... My value is 1.290000
Hi I am string ... My value is Devansh
○ PS C:\Users\Administrator\Documents\practice>
```

73) Difference between creating a list and a tuple

```
File Edit Selection View Go Run ... ← → search practice [Administrator]
EXPLORER PRACTICE .vscode practice practice.py
C:\Users\...Administrator\Documents\practice\practice.py + Untracked
1 list_ = [4, 5, 7, 1, 7]
2 tuple_ = (4, 1, 8, 3, 9)
3
4 print("List is: ", list_)
5 print("Tuple is: ", tuple_)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SPELL CHECKER
n32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56289' ... 'c:\Users\Administrator\Documents\practice\practice.py'
Hi I am Integer ... My value is 10
Hi I am float ... My value is 1.290000
Hi I am string ... My value is Devansh
PS C:\Users\Administrator\Documents\practice> c;; cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Pr...
n32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56327' ... 'c:\Users\Administrator\Documents\...
\practice\practice.py'
x5cadapter\..\..\x5cdebugpy\x5clauncher' '56327' ... 'c:\x5cUsers\x5cAdministrator\x5cDocuments\x5cprac...
e\x5cpрактиce.py'; e0e54214-3642-4a88-8626-5856f910d47dlist is: [4, 5, 7, 1, 7]
Tuple is: (4, 1, 8, 3, 9)
○ PS C:\Users\Administrator\Documents\practice> []

Ln 2, Col 1 Spaces: 4 UTF-8 CRLF Python 3.13.1 Go Live Prettier
Type here to search powershell Python Deb...
Launchpad 0 0 0 111 bytes Git Graph 18:24 ENG US 06-02-2025
```

Output:

```
File Edit Selection View Go Run ... ← → search practice [Administrator]
EXPLORER PRACTICE .vscode practice practice.py
C:\Users\...Administrator\Documents\practice\practice.py + Untracked
1 list_ = [4, 5, 7, 1, 7]
2 tuple_ = (4, 1, 8, 3, 9)
3
4 print("List is: ", list_)
5 print("Tuple is: ", tuple_)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SPELL CHECKER
n32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56289' ... 'c:\Users\Administrator\Documents\...
\practice\practice.py'
Hi I am Integer ... My value is 10
Hi I am float ... My value is 1.290000
Hi I am string ... My value is Devansh
PS C:\Users\Administrator\Documents\practice> c;; cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Pr...
n32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56327' ... 'c:\Users\Administrator\Documents\...
\practice\practice.py'
x5cadapter\..\..\x5cdebugpy\x5clauncher' '56327' ... 'c:\x5cUsers\x5cAdministrator\x5cDocuments\x5cprac...
e\x5cpрактиce.py'; e0e54214-3642-4a88-8626-5856f910d47dlist is: [4, 5, 7, 1, 7]
Tuple is: (4, 1, 8, 3, 9)
○ PS C:\Users\Administrator\Documents\practice> []

Ln 2, Col 1 Spaces: 4 UTF-8 CRLF Python 3.13.1 Go Live Prettier
Type here to search powershell Python Deb...
Launchpad 0 0 0 111 bytes Git Graph 18:24 ENG US 06-02-2025
```

74) type() function

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows a folder named "PRACTICE" containing ".vscode", "practice", and "practice.py".
- Editor:** Displays the code for "practice.py":

```
1 list_ = [4, 5, 7, 1, 7]
2 tuple_ = (4, 1, 8, 3, 9)
3 print( type(list_) )
4 print( type(tuple_) )
```

- Terminal:** Shows a terminal window with the command "python practice.py" and its output:

```
Traceback (most recent call last):
  File "c:/Users/Administrator/Documents/practice/practice.py", line 1, in <module>
    print( type(list_) )
          ^
NameError: name 'list_' is not defined. Did you mean: 'list'?
```

Output:

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows a folder named "PRACTICE" containing ".vscode", "practice", and "practice.py".
- Editor:** Displays the code for "practice.py":

```
1 list_ = [4, 5, 7, 1, 7]
2 tuple_ = (4, 1, 8, 3, 9)
3 print( type(list_) )
4 print( type(tuple_) )
```

- Terminal:** Shows a terminal window with the command "python practice.py" and its output:

```
list
tuple
```

75) difference between lists and tuples in immutability and mutability.

```
File Edit Selection View Go Run ... ← → search practice [Administrator]
EXPLORER PRACTICE .vscode practice practice.py ...
practice.py > ...
1 list_ = ["Python", "Lists", "Tuples", "Differences"]
2 tuple_ = ("Python", "Lists", "Tuples", "Differences")
3
4 # modifying the last string in both data structures
5 list_[3] = "Mutable"
6 print(list_)
7 try:
8     tuple_[3] = "Immutable"
9     print(tuple_)
10 except TypeError:
11     print("Tuples cannot be modified because they are immutable")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SPELL CHECKER

```
● ogram Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-w1n32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56383' --- 'c:\Users\Administrator\Documents\practice\practice.py'
<class 'list'>
<class 'tuple'>
PS C:\Users\Administrator\Documents\practice> c: cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Pr
n32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56449' --- 'c:\Users\Administrator\Documents\p
\practice\practice.py'
x5cadapter\..\..\x5cdebugpy\x5clauncher' '56449' --- 'c:\x5cUsers\x5cAdministrator\x5cDocuments\x5cpracti
e\x5cpractice.py':0xe0e54214-3642-4a88-8626-5856f910d47d['Python', 'Lists', 'Tuples', 'Mutable']
Tuples cannot be modified because they are immutable
○ PS C:\Users\Administrator\Documents\practice> []
Ln 11, Col 2 Spaces: 4 UTF-8 CRLF Python 3.13.1 Go Live Prettier
```

Type here to search

Output:

```
File Edit Selection View Go Run ... ← → search practice [Administrator]
EXPLORER PRACTICE .vscode practice practice.py ...
practice.py > ...
1 list_ = ["Python", "Lists", "Tuples", "Differences"]
2 tuple_ = ("Python", "Lists", "Tuples", "Differences")
3
4 # modifying the last string in both data structures
5 list_[3] = "Mutable"
6 print(list_)
7 try:
8     tuple_[3] = "Immutable"
9     print(tuple_)
10 except TypeError:
11     print("Tuples cannot be modified because they are immutable")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SPELL CHECKER

```
● ogram Files\Python313\python.exe' 'c:\Users\Administrator\.vscode\extensions\ms-python.debugpy-2024.14.0-w1n32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56383' --- 'c:\Users\Administrator\Documents\practice\practice.py'
<class 'list'>
<class 'tuple'>
PS C:\Users\Administrator\Documents\practice> c: cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Pr
n32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56449' --- 'c:\Users\Administrator\Documents\p
\practice\practice.py'
x5cadapter\..\..\x5cdebugpy\x5clauncher' '56449' --- 'c:\x5cUsers\x5cAdministrator\x5cDocuments\x5cpracti
e\x5cpractice.py':0xe0e54214-3642-4a88-8626-5856f910d47d['Python', 'Lists', 'Tuples', 'Mutable']
Tuples cannot be modified because they are immutable
○ PS C:\Users\Administrator\Documents\practice> []
Ln 11, Col 2 Spaces: 4 UTF-8 CRLF Python 3.13.1 Go Live Prettier
```

Type here to search

76) Size Difference

The screenshot shows the Visual Studio Code interface. In the Explorer sidebar, there is a folder named 'PRACTICE' containing '.vscode', 'practice', and 'practice.py'. The 'practice.py' file is selected. The code in 'practice.py' is:

```
list_ = ["Python", "Lists", "Tuples", "Differences"]
tuple_ = ("Python", "Lists", "Tuples", "Differences")
print("Size of tuple: ", tuple_.__sizeof__())
print("Size of list: ", list_.__sizeof__())
```

In the terminal tab, the output of running the script is shown:

```
python.exe 'c:\Users\Administrator\vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56479' '' 'c:\Users\Administrator\Documents\practice\practice.py'
[Python, 'Lists', 'Tuples', 'Mutable']
Tuples cannot be modified because they are immutable
PS C:\Users\Administrator\Documents\practice> cd 'c:\Users\Administrator\Documents\practice'; & 'c:\Users\Administrator\vscode\extensions\ms-python.debugpy-2024.14.0-win32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '56479' '' 'c:\Users\Administrator\Documents\practice\practice.py'
x5cadapter\..\..\x5cdebugpy\x5clauncher' '56479' '' 'c:\x5cUsers\x5cAdministrator\x5cDocuments\x5cpractice\x5cpractice.py'; e0e54214-3642-4a88-8626-5856f910d47dSize of tuple:  56
Size of list: 72
PS C:\Users\Administrator\Documents\practice>
```

Output:

This screenshot is identical to the one above, showing the VS Code interface with the 'practice.py' file open and its execution output in the terminal.