

Name: Rohan Vinayak Chaudhari

Batch: Data Engineering

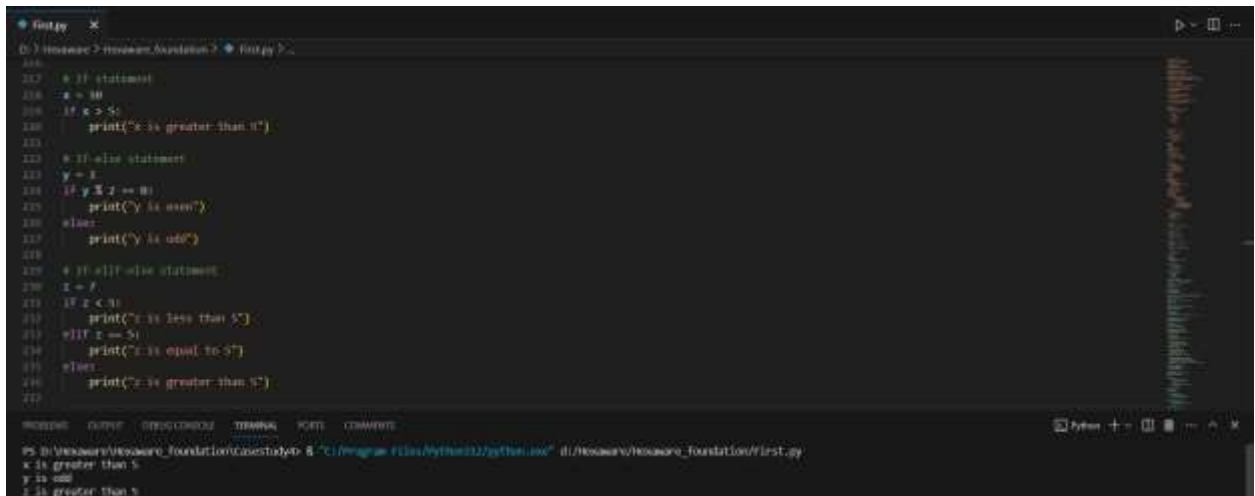
Date:29/01/2024

Topic: Python(control structures,list,sets,tuple,dict,oops,reading csv)

Solution:

1.Python:

1)If,ifelse,nested if elif:



```
227 # if statement
228 x = 30
229 if x > 5:
230     print("x is greater than 5")
231
232 # if-else statement
233 y = 1
234 if y % 2 == 0:
235     print("y is even")
236 else:
237     print("y is odd")
238
239 # if-elif-else statement
240 z = 7
241 if z < 3:
242     print("z is less than 5")
243 elif z == 5:
244     print("z is equal to 5")
245 else:
246     print("z is greater than 5")
247
```

Output:

```
PS D:\Users\Vinayak_Foundation\Documents> python first.py
x is greater than 5
y is odd
z is greater than 5
```

2)for,while,nestedloop,break,continue,pass

```
◆ First.py ×
D:\> Hexaware > Hexaware_foundation > ◆ First.py > ...

238 # For Loop
239 for i in range(5):
240     print(i)
241
242 # While loop
243 count = 0
244 while count < 3:
245     print("Count:", count)
246     count += 1
247
248 # Nested loop
249 for i in range(2):
250     for j in range(2):
251         print(i, j)
252
253 # Break, Continue & Pass
254 for num in range(5):
255     if num == 3:
256         break
257     print(num)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

0
1
2
3
4
Count: 0
Count: 1
Count: 2
0 0
0 1
1 0
1 1
2 0
2 1
0
1
2
```

3)input,list,list methods,set

```
◆ First.py ×
D:\> Hexaware > Hexaware_foundation > ◆ First.py > ...

259 # Input and Output
260 user_input = input("Enter something: ")
261 print("You entered:", user_input)
262
263 # Introduction to lists
264 my_list = [1, 2, 3, 4]
265 print(my_list)
266
267 # List Methods and Slicing
268 my_list.append(5)
269 print(my_list[1:3])
270
271 # Introduction to Dictionaries & Dictionary Methods
272 my_dict = {'key1': 'value1', 'key2': 'value2'}
273 print(my_dict['key1'])
274
275 # Introduction to Set & Set Methods
276 my_set = {1, 2, 3, 3}
277 print(my_set)
278

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

Enter something: 1
You entered: 1
[1, 2, 3, 4]
[2, 3]
value1
{1, 2, 3}
```

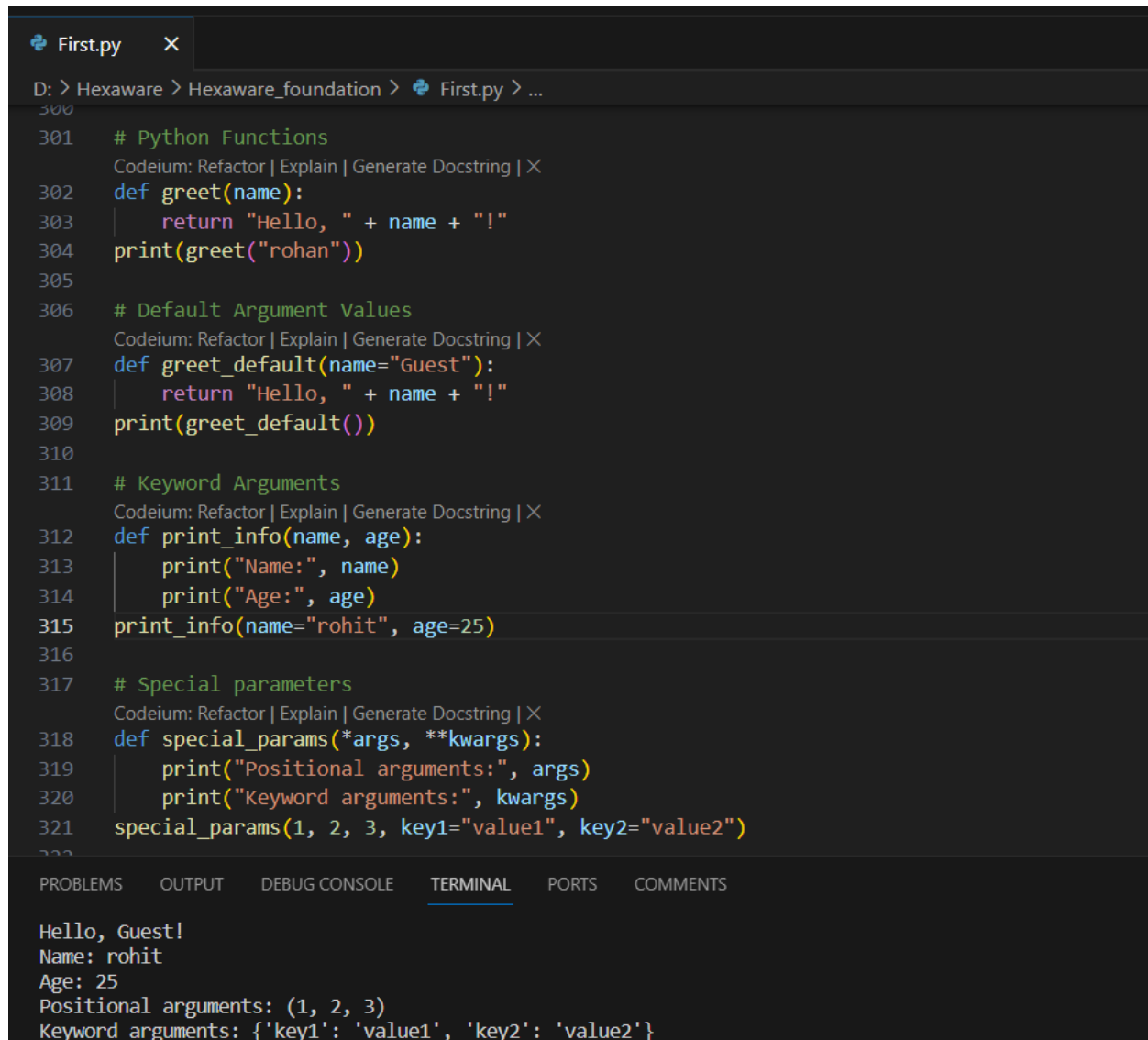
4)map,map methods,lambda function,date time.string function.mapping function,number function

```
First.py X
D: > Hexaware > Hexaware_foundation > First.py > ...

279 # Introduction to Map & Map Methods
280 my_map = map(lambda x: x * 2, [1, 2, 3])
281 print(list(my_map))
282
283 # Mapping function
284 Codeium: Refactor | Explain | Generate Docstring | X
285 def square(x):
286     return x ** 2
287 squared_values = list(map(square, [1, 2, 3]))
288
289 # String Function
290 string_example = "Hello, World!"
291 print(len(string_example))
292
293 # Number Function
294 num_example = 3.14
295 print(round(num_example))
296
297 # Date and Time Function
298 from datetime import datetime
299 current_time = datetime.now()
300 print(current_time)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
[2, 4, 6]
13
3
2024-01-29 21:25:51.500858
```

5) functions, different types of functions:



The image shows a code editor window with a file named 'First.py'. The code defines four functions: 'greet', 'greet_default', 'print_info', and 'special_params'. Each function is preceded by a comment and a Codeium suggestion bar. The 'greet' function takes a name and returns a greeting. 'greet_default' has a default value for the name. 'print_info' prints name and age. 'special_params' uses *args and **kwargs to handle variable arguments. The terminal at the bottom shows the output of these functions: 'Hello, Guest!', 'Name: rohit', 'Age: 25', 'Positional arguments: (1, 2, 3)', and 'Keyword arguments: {'key1': 'value1', 'key2': 'value2'}'.

```
300
301 # Python Functions
302 Codeium: Refactor | Explain | Generate Docstring | X
303 def greet(name):
304     return "Hello, " + name + "!"
305 print(greet("rohan"))
306
307 # Default Argument Values
308 Codeium: Refactor | Explain | Generate Docstring | X
309 def greet_default(name="Guest"):
310     return "Hello, " + name + "!"
311 print(greet_default())
312
313 # Keyword Arguments
314 Codeium: Refactor | Explain | Generate Docstring | X
315 def print_info(name, age):
316     print("Name:", name)
317     print("Age:", age)
318 print_info(name="rohit", age=25)
319
320 # Special parameters
321 Codeium: Refactor | Explain | Generate Docstring | X
322 def special_params(*args, **kwargs):
323     print("Positional arguments:", args)
324     print("Keyword arguments:", kwargs)
325 special_params(1, 2, 3, key1="value1", key2="value2")
326
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
Hello, Guest!
Name: rohit
Age: 25
Positional arguments: (1, 2, 3)
Keyword arguments: {'key1': 'value1', 'key2': 'value2'}
```

6)arbitrary argument, oops, class, obj, access specifiers

```
First.py X
D: > Hexaware > Hexaware_foundation > First.py > ...

323 # Arbitrary Argument Lists
    Codeium: Refactor | Explain | Generate Docstring | X
324 def sum_all(*numbers):
325     return sum(numbers)
326 result = sum_all(1, 2, 3, 4)
327
328
329 # OOPS
    Codeium: Explain
330 class MyClass:
    Codeium: Refactor | Explain | Generate Docstring | X
331     def __init__(self, value):
332         self.value = value
333
    Codeium: Refactor | Explain | Generate Docstring | X
334     def display(self):
335         print("Value:", self.value)
336
337 # Class and Object
338 obj = MyClass(10)
339 obj.display()
340
341 # Access Specifiers
    Codeium: Explain
342 class MyClassPrivate:
    Codeium: Refactor | Explain | Generate Docstring | X
343     def __init__(self, value):
344         self.__value = value
345
346
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
Positional arguments: (1, 2, 3)
Keyword arguments: {'key1': 'value1', 'key2': 'value2'}
Value: 10
```

7) constructor, polymorphism, inheritance

```
First.py X
D: > Hexaware > Hexaware_foundation > First.py > ...

346 # Constructor
    Codeium: Explain
347 class MyClassConstructor:
    Codeium: Refactor | Explain | Generate Docstring | X
348     def __init__(self):
349         print("Constructor called")
350
351 # Inheritance
    Codeium: Explain
352 class ChildClass(MyClass):
    Codeium: Refactor | Explain | Generate Docstring | X
353     def __init__(self, value, additional_value):
354         super().__init__(value)
355         self.additional_value = additional_value
356
357 # Polymorphism
    Codeium: Explain
358 class sample:
    Codeium: Refactor | Explain | Generate Docstring | X
359     def __init__(self, *args):
360         if len(args) > 1:
361             self.ans = 0
362             for i in args:
363                 self.ans += i
364         elif isinstance(args[0], int):
365             self.ans = args[0] * args[0]
366         elif isinstance(args[0], str):
367             self.ans = len(args[0])
368
369 ...
```

8)overriding,file handling,exception handling

```
First.py X
D: > Hexaware > Hexaware_foundation > First.py > ...
369
370 # Method Overriding
371 class parent():
372     def greet(Self):
373         print('hello a')
374 class child(parent):
375     def greet(Self):
376         print('hello b')
377 p=parent()
378 c=child()
379 p.greet()
380 c.greet()
381
382 # File handling
383 with open("example.txt", "w") as file:
384     file.write("Hello, File!")
385
386 # Exception Handling
387 try:
388     result = 10 / 0
389 except ZeroDivisionError as e:
390     print("Error:", e)
391
392 import math
393
394 # Using math module functions
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
Value: 10
hello a
hello b
Error: division by zero
```

9)modules, math module

```
First.py
D: > Hexaware > Hexaware_foundation > First.py > ...

391
392 import math
393
394 # Using math module functions
395 print("Square root of 16:", math.sqrt(16))
396 print("Ceiling of 3.14:", math.ceil(3.14))
397 print("Factorial of 5:", math.factorial(5))
398 # Power function
399 print("2 to the power of 3:", math.pow(2, 3))
400
401
402
403
--
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

hello b
Error: division by zero
Square root of 16: 4.0
Ceiling of 3.14: 4
Factorial of 5: 120
2 to the power of 3: 8.0
PS D:\Hexaware\Hexaware_foundation\Casestudy4> []
```

10)reading CSV FILE:

```
Python
D:\code > data_analyt > pandas > Pandas.py > df.read()
+ Code + Markdown + Run All + Restart + Clear All Outputs + Variables + Outline + Python 3.12.0

import pandas as pd
(1) ✓ 0.0s Python

-- C:\Users\jason\Anaconda3\envs\pandas\lib\site-packages\pandas\compat\pyarrow.py:11: DeprecationWarning:
pyarrow will become a required dependency of pandas in the next major release of pandas (pandas 3.0),
(To allow more performant data types, such as the Arrow string type, and better interoperability with other libraries)
but was not found to be installed on your system.
If this would cause problems for you,
please provide us feedback at https://github.com/pandas-dev/pandas/issues/54986

import pandas as pd

D:\code > df=pd.read_csv("https://raw.githubusercontent.com/javiercasadojo/datasets/master/titanic.csv")
(1) ✓ 0.0s Python

D:\code > df.head()
(1) ✓ 0.0s Python

--
PassengerId  Survived  Pclass      Name  Sex  Age  SibSp  Parch    Ticket   Fare  Cabin  Embarked
0          1         0       1   Braund, Mr. Owen Harris  male  22.0    1    0   A/5 21171  7.2500  NaN      S
1          2         1       1  Cumings, Mrs. John Bradley (Florence Briggs Th... female  38.0    1    0   PC 17599  51.2833  C85      C
2          3         1       3  Heikinen, Miss. Laina          female  26.0    0    0  STON/O2. 3101282  7.9250  NaN      S
3          4         1       1  Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35.0    1    0  113803  53.1000  C123      S
4          5         0       1    Allen, Mr. William Henry    male  35.0    0    0  373450  80.5000  NaN      S
```