



## **ARTIFICIAL INTELLIGENCE LAB**

**BSCYS-3rd Semester**

**Fall 2025**

**Lab Report # 1**

**Submitted: To:** Sir Mubashir Iqbal

**Submitted By:** M.Umer

**Reg No:** (24-CyS-024)

**Department:** Cyber Security (A)

DEPARTMENT OF COMPUTER SCIENCE HITEC UNIVERSITY TAXILA

BS CYBER SECURITY PROGRAM

## TASK 1:

### Code:

```
CountMatchingStrings.py X
C: > Users > FAIZAN COMPUTER > CountMatchingStrings.py > ...

1 class CountMatchingStrings:
2     def count_strings(self):
3         strings = ['abc', 'xyz', 'aba', '1221', 'xyzzxy', 'aa', '122']
4         count = 0
5
6         for s in strings:
7             if len(s) >= 2 and s[0] == s[-1]:
8                 count += 1
9
10        print("Number of matching strings:", count)
11
12 obj = CountMatchingStrings()
13 obj.count_strings()
14
```

### Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\FAIZAN COMPUTER> & "C:/Users/FAIZAN COMPUTER/AppData/Local/Programs/Python/Python314
/python.exe" "c:/Users/FAIZAN COMPUTER/CountMatchingStrings.py"
Number of matching strings: 4
PS C:\Users\FAIZAN COMPUTER>
```

## Task 2: Sum and Product of Numbers in a Tuple

### Code:

```
... CountMatchingStrings.py TupleSumProduct.py X
C: > Users > FAIZAN COMPUTER > TupleSumProduct.py > ...
1 class TupleSumProduct:
2     def calculate(self):
3         numbers = (1, 2, 4, 2, 6)
4         total_sum = 0
5         product = 1
6
7         for num in numbers:
8             total_sum += num
9             product *= num
10
11         print("Sum is:", total_sum)
12         print("Product is:", product)
13
14 obj = TupleSumProduct()
15 obj.calculate()
16
```

**Output:**

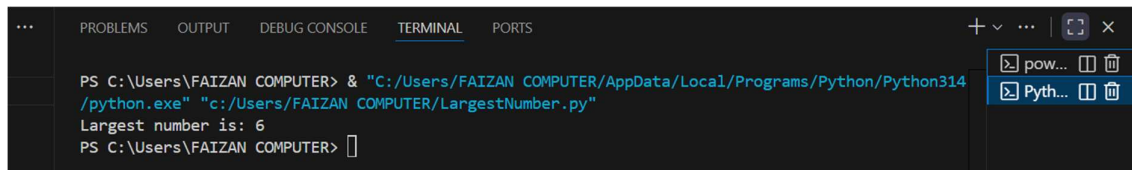
```
... PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
/python.exe" "c:/Users/FAIZAN COMPUTER/TupleSumProduct.py"
Sum is: 15
Product is: 96
PS C:\Users\FAIZAN COMPUTER>
```

### Task 3: Largest Number in a List

**Code:**

```
... CountMatchingStrings.py TupleSumProduct.py LargestNumber.py X
C: > Users > FAIZAN COMPUTER > LargestNumber.py > ...
1 class LargestNumber:
2     def find_largest(self):
3         numbers = [1, 2, 4, 2, 6]
4         largest = numbers[0]
5
6         for num in numbers:
7             if num > largest:
8                 largest = num
9
10        print("Largest number is:", largest)
11
12 obj = LargestNumber()
13 obj.find_largest()
14
```

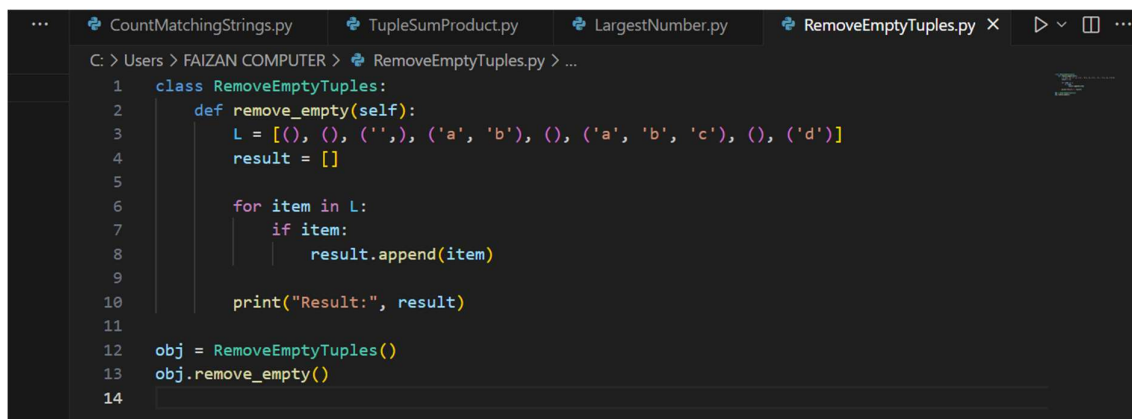
## Output:



```
PS C:\Users\FAIZAN COMPUTER> & "C:/Users/FAIZAN COMPUTER/AppData/Local/Programs/Python/Python314/python.exe" "c:/Users/FAIZAN COMPUTER/LargestNumber.py"
Largest number is: 6
PS C:\Users\FAIZAN COMPUTER>
```

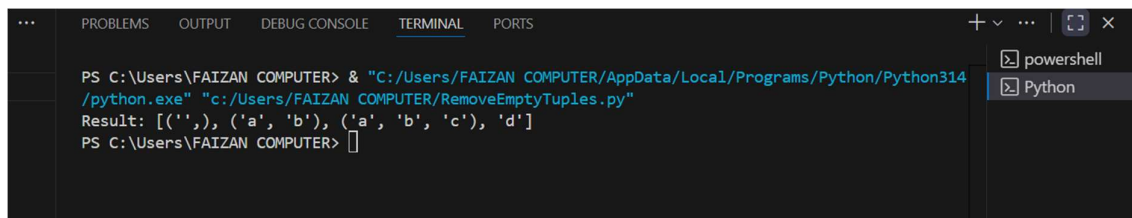
## Task 4: Remove Empty Tuples from a List

## Code:



```
1 class RemoveEmptyTuples:
2     def remove_empty(self):
3         L = [(), (), ('',), ('a', 'b'), (), ('a', 'b', 'c'), (), ('d')]
4         result = []
5
6         for item in L:
7             if item:
8                 result.append(item)
9
10        print("Result:", result)
11
12    obj = RemoveEmptyTuples()
13    obj.remove_empty()
14
```

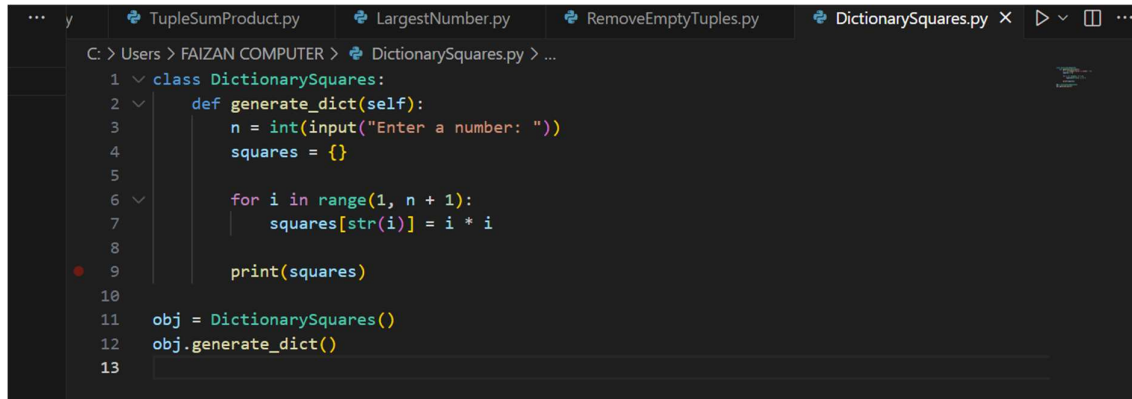
## Output:



```
PS C:\Users\FAIZAN COMPUTER> & "C:/Users/FAIZAN COMPUTER/AppData/Local/Programs/Python/Python314/python.exe" "c:/Users/FAIZAN COMPUTER/RemoveEmptyTuples.py"
Result: [('',), ('a', 'b'), ('a', 'b', 'c'), 'd']
PS C:\Users\FAIZAN COMPUTER>
```

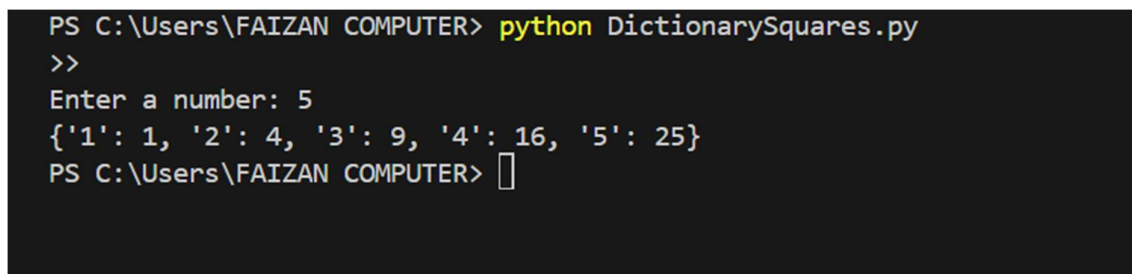
## Task 5: Generate Dictionary of Squares

### Code:

A screenshot of a Python IDE with a dark theme. The top bar shows several open files: TupleSumProduct.py, LargestNumber.py, RemoveEmptyTuples.py, and DictionarySquares.py. The main editor window displays the code for DictionarySquares.py. The code defines a class DictionarySquares with a method generate\_dict that takes user input, calculates squares, and prints them. Below the class, an object is instantiated and the method is called.

```
C: > Users > FAIZAN COMPUTER > DictionarySquares.py > ...  
1 class DictionarySquares:  
2     def generate_dict(self):  
3         n = int(input("Enter a number: "))  
4         squares = {}  
5  
6         for i in range(1, n + 1):  
7             squares[str(i)] = i * i  
8  
9         print(squares)  
10  
11 obj = DictionarySquares()  
12 obj.generate_dict()  
13
```

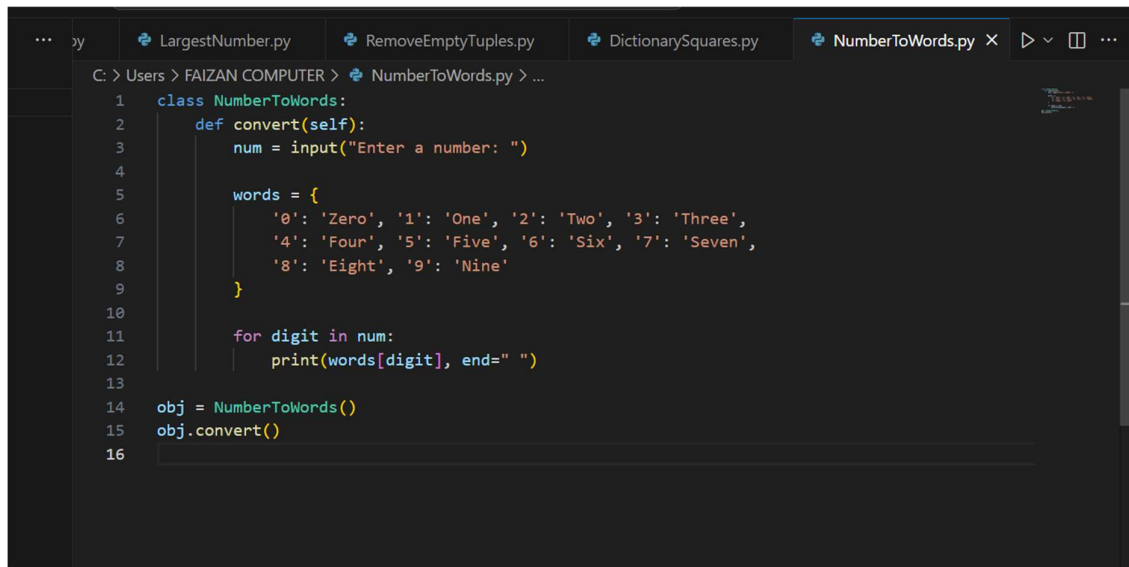
### Output:

A screenshot of a terminal window with a dark background. It shows the command to run the script, the user input '5', and the resulting dictionary output.

```
PS C:\Users\FAIZAN COMPUTER> python DictionarySquares.py  
>>  
Enter a number: 5  
{'1': 1, '2': 4, '3': 9, '4': 16, '5': 25}  
PS C:\Users\FAIZAN COMPUTER>
```

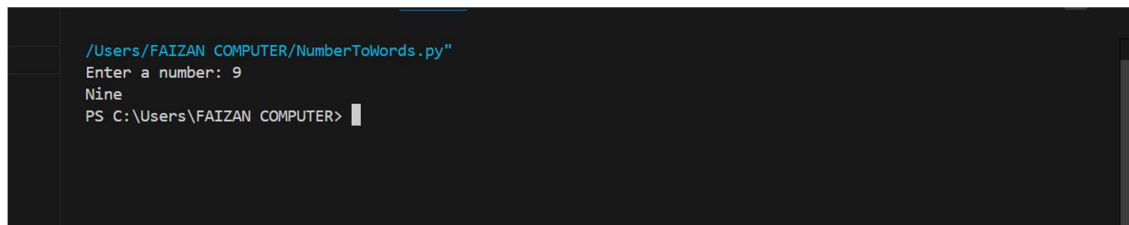
## Task 6: Numbers to Words using Dictionary

## Code:



```
1 class NumberToWords:
2     def convert(self):
3         num = input("Enter a number: ")
4
5         words = {
6             '0': 'Zero', '1': 'One', '2': 'Two', '3': 'Three',
7             '4': 'Four', '5': 'Five', '6': 'Six', '7': 'Seven',
8             '8': 'Eight', '9': 'Nine'
9         }
10
11         for digit in num:
12             print(words[digit], end=" ")
13
14 obj = NumberToWords()
15 obj.convert()
16
```

## Output:



```
/Users/FAIZAN COMPUTER/NumberToWords.py"
Enter a number: 9
Nine
PS C:\Users\FAIZAN COMPUTER>
```

## Overall conclusion:

In this lab report, the programs demonstrate string operations, tuple calculations, list processing, dictionary usage, and basic logic building in Python. Like the codes count matching strings, calculate sum and product of tuple elements, find the largest list value, remove empty tuples, generate a dictionary of squares, and convert numbers into words. This approach improves readability, follows object-oriented principles.