A
Lab Record of

Python Programming with Problem Solving

Master of Computer Application - I Sem

RUNGTA INTERNATIONAL SKILLS UNIVERSITY

SESSION: 2025-26

Guided By:-
Mrs. Kavita Kanwar
(Assistant Professor)

Submitted By:-
Sumit Kumar
Ref No. 11485

Submitted To:

RUNGTA INTERNATIONAL SKILLS UNIVERSITY, CG

SCHOOL OF INFORMATION TECHNOLOGY

Index

S. No	Name Of Practical	Submission Date	Remarks
1.	Write a Python program to Print the Patterns.		
2.	Write a Python program to the probability of rolling a dice / flipping a coin.		
3.	Write a Python program to find factorial of a number using recursion.		
4.	Write a program to search for an item in a user-provided list and display the position if found, otherwise print “Item not found.”		
5.	Given a list of employee records as dictionaries, sort them by salary and display the sorted list.		
6.	Write a program that reads a text file and counts the number of lines, words, and characters.		
7.	Read a sentence and display how many times each word appears, ignoring case and punctuation.		
8.	Write a Python program that lists all files in a directory and categorizes them by file extension.		

PRACTICAL ASSIGNMENT

AIM 1 - Write a Python program to Print the Patterns

python_practical_1.ipynb

C: > Users > sumit > Desktop > MCA_1st_sem > SUMIT_KUMAR > python > python_practical_1.ipynb > print("Pyramid Pattern")

Generate | + Code | + Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline | ...

```
print("Number Pattern")
for i in range(1,6):#Row
    for j in range(1,i+1):#Colom
        print(j,end=' ')
    print() #For New Line
```

[1]

```
Number Pattern
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

```
print("Pyramid Pattern")
for i in range(1,6):#For Row
    for s in range(5-i):#For Space
        print(" ",end='')#Space print
    for j in range(1,i+1):#For Column
        print("*",end=' ')
    print()
```

[7] ✓ 0.0s

```
Pyramid Pattern
    *
   * *
  * * *
 * * * *
* * * * *
```

python_practical_1.ipynb

C: > Users > sumit > Desktop > MCA_1st_sem > SUMIT_KUMAR > python > python_practical_1.ipynb > print("Pyramid Pattern")

Generate | Code | Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline | ...

```
print("Diamond Payyern")

#upper diamond
space1=4
for i in range(1,6):
    for s in range(space1):
        print(" ",end='')
    for j in range(1,i+1):
        print("*",end=' ')
    print()
    space1-=1
#lower diamond
space2=0
for i in range(5,0,-1):
    for s in range(space2):
        print(" ",end='')
    for j in range(1,i+1):
        print("*",end=' ')
    print()
    space2+=1
```

[ ]

```
Diamond Payyern
    *
   * *
  * * *
 * * * *
* * * * *
* * * * *
 * * * *
  * * *
   * *
    *
```

python_practical_1.ipynb

C: > Users > sumit > Desktop > MCA_1st_sem > SUMIT_KUMAR > python > python_practical_1.ipynb > print("X Shaped Pattern")

Generate | Code | Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline | ...

```
print("Hallow Square Pattern")

for row in range(1,6):
    for col in range(1,6):
        if col==1 or col==5 or row==1 or row==5:
            print("*",end=' ')
        else:
            print(" ",end=' ')
    print()
```

[2]

```
Hallow Square Pattern
* * * * *
*       *
*       *
*       *
* * * * *
```

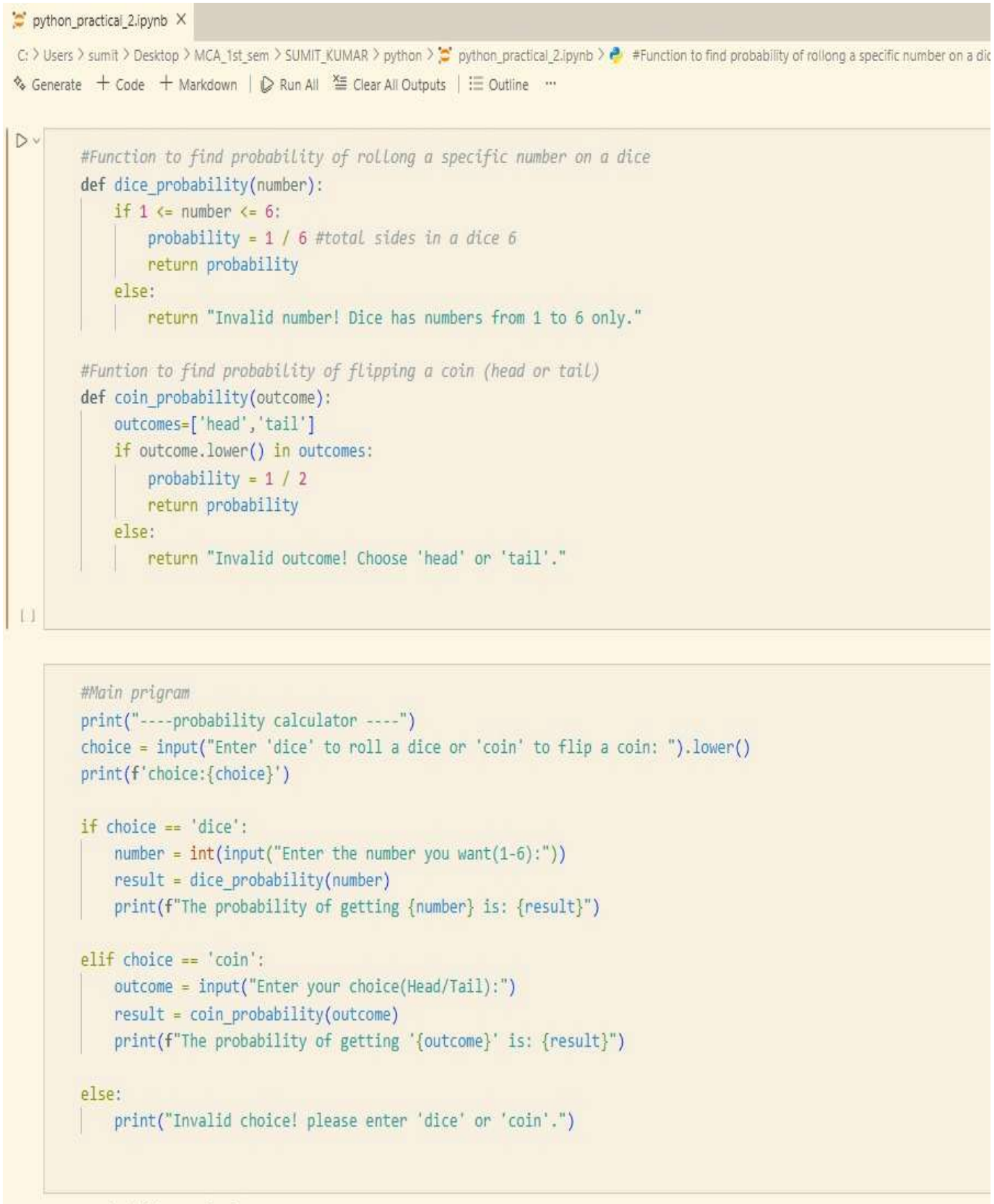
```
print("X Shaped Pattern")

n = 5   # must be odd
for i in range(n):
    for j in range(n):
        if i == j or i + j == n - 1:
            print("*", end="")
        else:
            print(" ", end="")
    print()
```

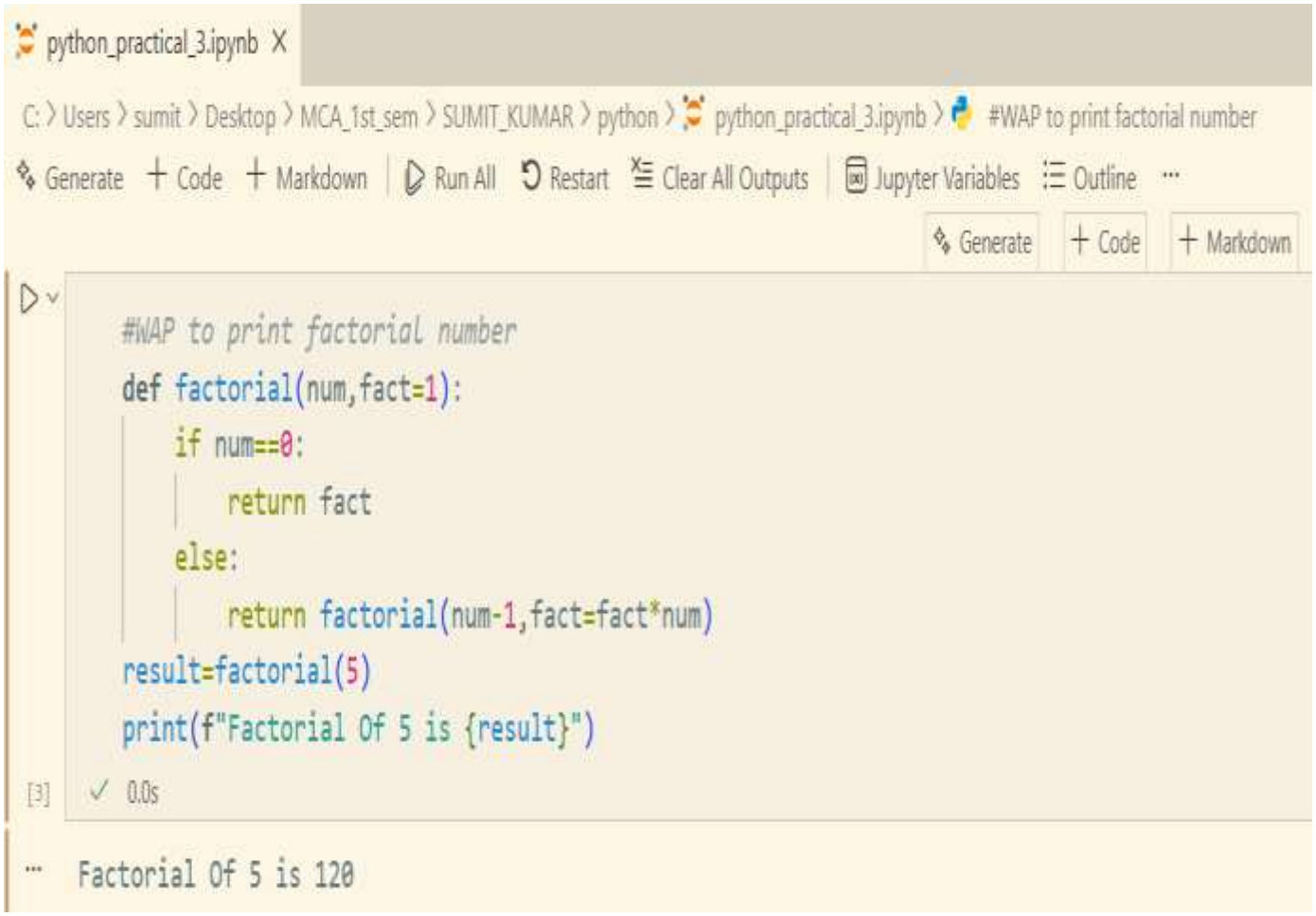
[8]

```
X Shaped Pattern
*   *
 * *
  *
 * *
*   *
```

AIM 2 - Write a Python program to the probability of rolling a dice / flipping a coin.



AIM 3 - Write a Python program to find factorial of a number using recursion.



AIM 4 - Write a program to search for an item in a user-provided list and display the position if found, otherwise print “Item not found.”

AIM 5 - Given a list of employee records as dictionaries, sort them by salary and display the sorted list.

AIM 6 - Write a program that reads a text file and counts the number of lines, words, and characters.

python_practical_6.ipynb

C: > Users > sumit > Desktop > MCA_1st_sem > SUMIT_KUMAR > python > python_practical_6.ipynb > def count_line():

Generate | Code | Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline | ...

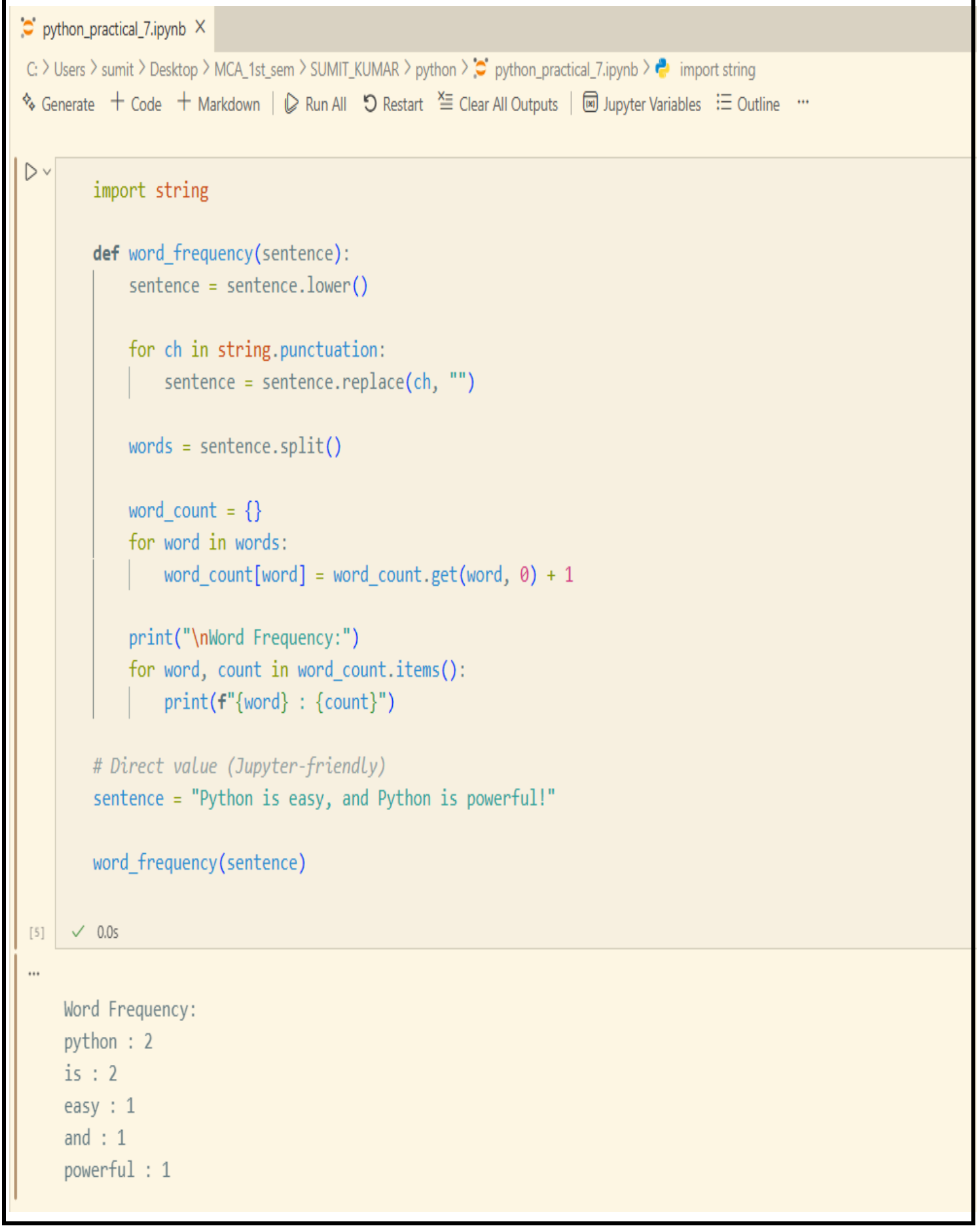
```
def count_line():
    line_count=0
    words_count=0
    characters_count=0

    while True:
        with open("sample.txt",'r') as fobj:
            if line_count == 0:
                print("Sample text file(sample.txt)")
                for interate in fobj.readlines():
                    print(interate,end='')
                    line_count+=1
                print(f"\nLines:{line_count}")
            elif words_count == 0:
                for interate in fobj.readlines():
                    words_count+=len(interate.split())
                print(f'Words:{words_count}')
            elif characters_count == 0:
                for iterate in list(fobj.read()):
                    if iterate != ' ' and iterate != '\n':
                        characters_count+=1
                print(f'characters:{characters_count}')
            else:
                break
count_line()
```

[5] ✓ 0.0s

```
Sample text file(sample.txt)
Sample text file(sample.txt)
Lines:1
Words:3
characters:26
```

AIM 7 - Read a sentence and display how many times each word appears, ignoring case and punctuation.



AIM 8 - Write a Python program that lists all files in a directory and categorizes them by file extension.

python_practical_8.ipynb

C: > Users > sumit > Desktop > MCA_1st_sem > SUMIT_KUMAR > python > python_practical_8.ipynb > import os

Generate | Code | Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline | ...

```python
import os
from collections import defaultdict

def categorize_files(directory):
    file_types = defaultdict(list)  # Dictionary to hold extensions and file names

    # List all files in the directory
    for filename in os.listdir(directory):
        if os.path.isfile(os.path.join(directory, filename)):
            # Split filename and extension
            name, ext = os.path.splitext(filename)
            ext = ext.lower() if ext else "No Extension"
            file_types[ext].append(filename)

    # Display categorized files
    for ext, files in file_types.items():
        print(f"\n{ext}:")
        for f in files:
            print(f"  {f}")

# Set directory path (current directory in this case)
directory_path = "."
categorize_files(directory_path)
```

[1] ✓ 0.0s

```
.ipynb:
  python_practical_1.ipynb
  python_practical_2.ipynb
  python_practical_3.ipynb
  python_practical_4.ipynb
  python_practical_5.ipynb
  python_practical_6.ipynb
  python_practical_7.ipynb
  python_practical_8.ipynb

.txt:
  sample.txt
```