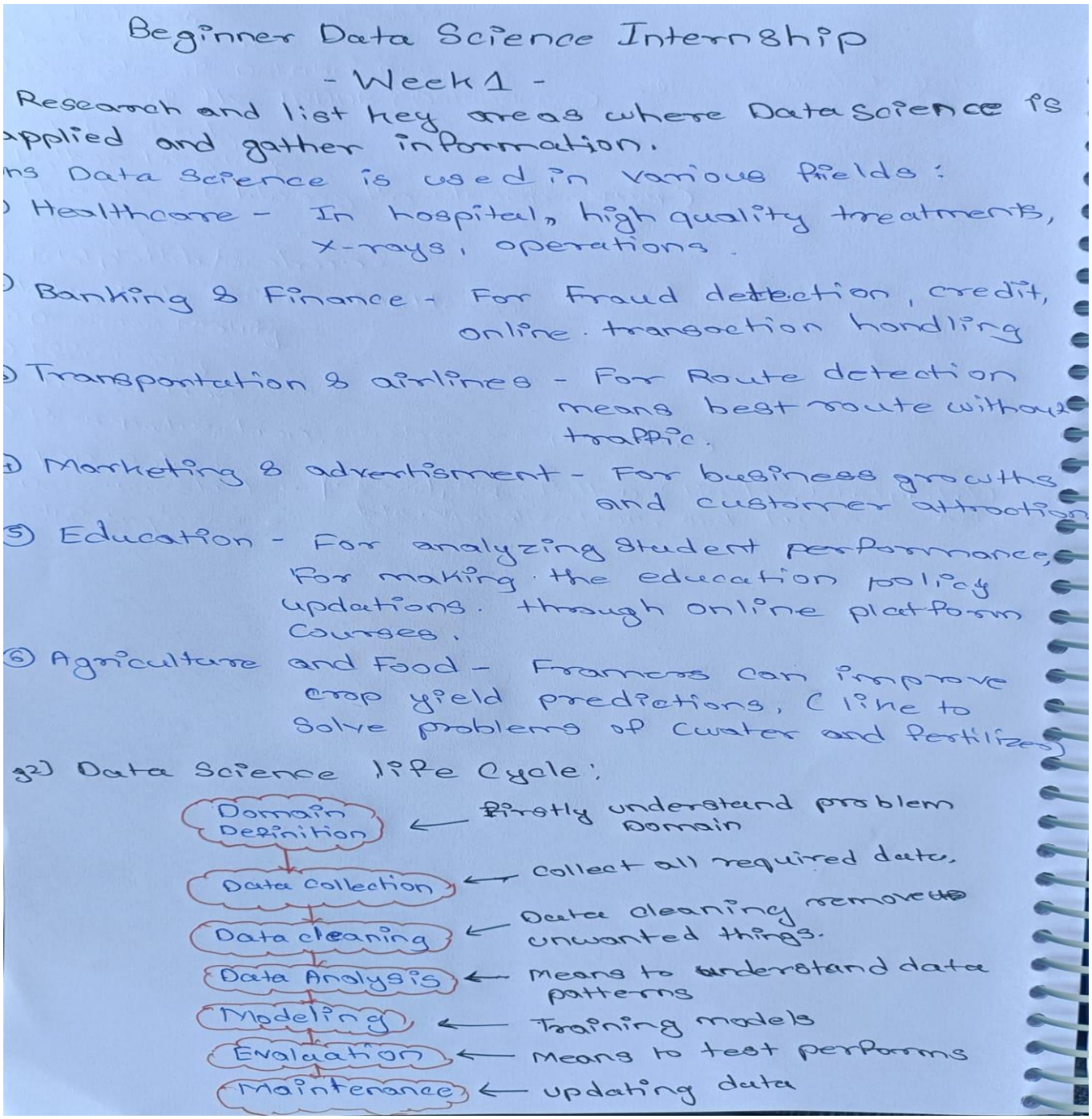
NAME:VEDANT ARVIND GOLE

DATA SCIENCE INTERNSHIP TASK NO. 1

1. Research and list key areas where Data Science is applied and gather information on how companies use data science to solve real-world problems.

2. Summarize the stages of the Data Science lifecycle and Create a diagram or flowchart to represent the lifecycle



b. Verify the installation by opening Jupyter Notebook.

- **Demonstrate the use of operators in python.**

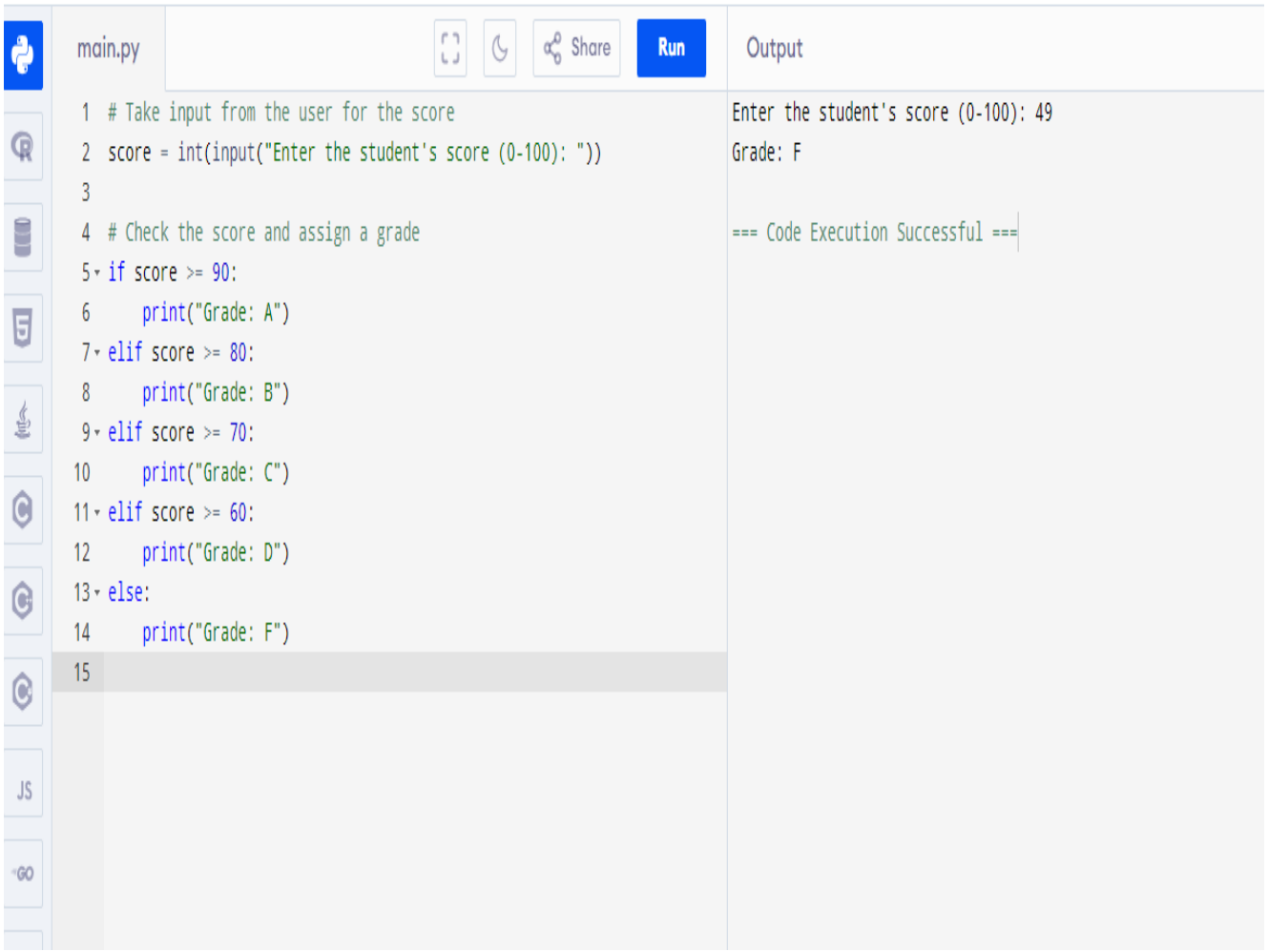
IDLE Shell 3.13.1 - C:/Users/HP/AppData/Local/Programs/Python/Python313/vedant48.py (3.13.1)

File Edit Shell Debug Options Window Help

```
Python 3.13.1 (tags/v3.13.1:0671451, Dec  3 2024, 19:06:28) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> a = 10
>>> b = 20
>>> print(a+b)
30
>>> #addition operator in python
>>>
>>> print(a-b)
-10
>>> #substraction operator in python
>>> print(a*b)
200
>>>
>>> #multiple operator in python
>>>
>>> print(a/b)
0.5
>>> #division in python
>>>
>>> print(a%b)
10
>>> #modules in python
>>>
>>> print(a**b)
100000000000000000000
>>> #Exponentiation in python
>>>
```

```
>>>
>>>
>>> #LOGICAL OPREATORS IN PYTHON
>>> a = 20
>>> b = 5
>>> print(a==b)
False
>>> print(a!=b)
True
>>> print(a>b)
True
>>> print(a<b)
False
>>>
>>>
>>> #LOGICAL OPERATORS IN PYTHON
>>> x = True
>>> y = False
>>> print(x and y)
False
>>> print(x or y)
True
>>> print(not x)
False
>>>
>>>
>>>
```

● **Demonstrate the use of elif in python:**



ELIF STATEMENT:

The elif statement in Python stands for "else if" and is used in conditional statements to check multiple expressions for truth value. It's part of an if-else block, allowing you to check for more than two conditions.

- **Demonstrate the use of collections in python:**

Collections in Python help you group and organize multiple related items, making your code cleaner and more efficient. Depending on the type of collection you choose, you can take advantage of various features, like mutability, uniqueness, and ordering.

Example :

```
fruits = ["apple", "banana", "cherry"]

fruits.append("orange")  # Add new item to list

print(fruits)

Output: ["apple", "banana", "cherry", "orange"]       #This is an example of lists in collection
```