

## 1. Grade Checker

Take a score as input and print the grade based on the following:

90+ : "A"

80-89 : "B"

70-79 : "C"

60-69 : "D"

Below 60 : "F"

here we used a basic if else statement to carry out marks and all.

The screenshot shows a dark-themed version of Visual Studio Code. The left sidebar displays a file tree under the 'LEARNPY' folder, with 'Grade\_checker.py' selected. The main editor area contains the following Python code:

```
score=int(input("Enter your score: "))
if score==90:
    grade='A'
elif score==80:
    grade='B'
elif score<=70:
    grade='C'
elif score<=60:
    grade='D'
else:
    grade='F'
print("Grade:",grade)
```

The bottom status bar shows tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is active, displaying a terminal session where the script is run and the output 'Grade: C' is shown.

## 2 Student Grades

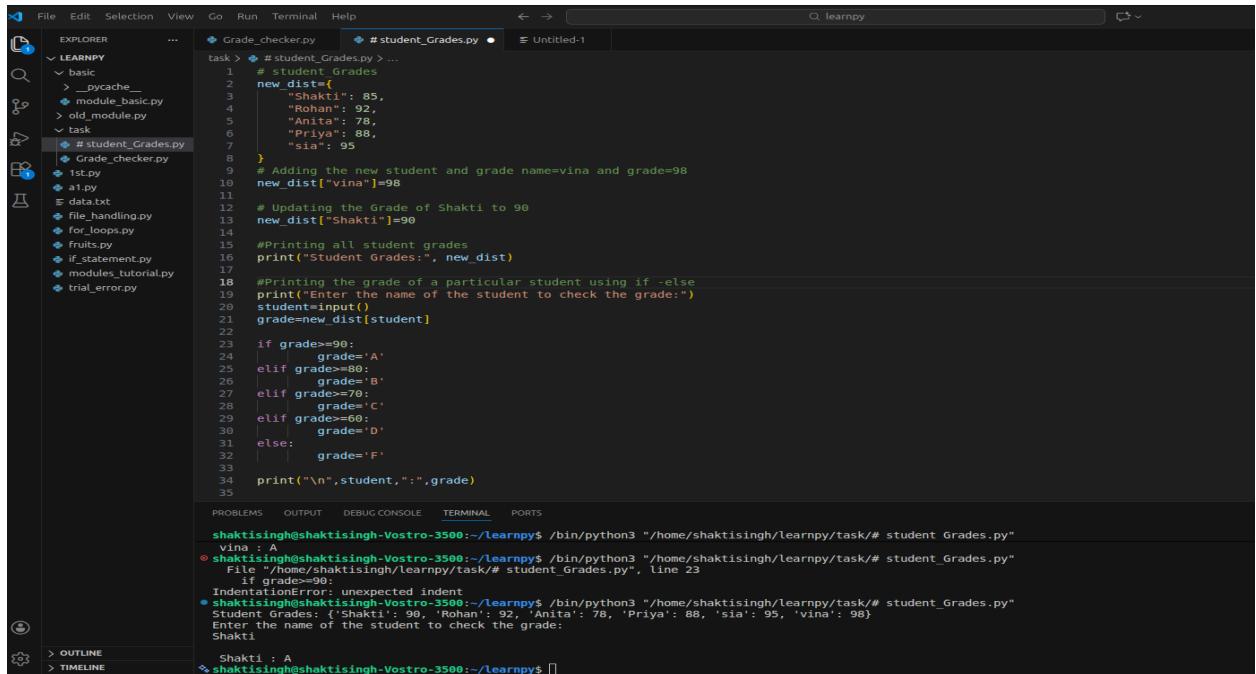
Create a dictionary where the keys are student names and the values are their grades. Allow the user to:

Add a new student and grade.

Update an existing student's grade.

Print all student grades.

Used dictionary and basic operations. Using if else:



The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows a project structure under "LEARNPY" with files like Grade\_checker.py, student\_Grades.py, 1st.py, a1.py, data.txt, file\_handling.py, for\_loops.py, fruits.py, if\_statement.py, modules\_tutorial.py, and trial\_error.py.
- Code Editor:** Displays the content of student\_Grades.py. The code uses a dictionary to store student names and their grades, adds a new student "vina" with grade 98, updates the grade of "Shakti" to 90, prints all student grades, and then asks for a student name to print their grade using an if-else statement.
- Terminal:** Shows the command-line output of running the script. It shows the addition of "vina" with grade 98, an indentation error at line 23, and the final output showing "Shakti : A".

```
# student_Grades.py
task > # student_Grades.py ...
1 # student_Grades
2 new_dist={}
3     "Shakti": 85,
4     "Rohan": 92,
5     "Anita": 78,
6     "Priya": 88,
7     "sia": 95
8
9     # Adding the new student and grade name=vina and grade=98
10    new_dist["vina"] = 98
11
12     # Updating the Grade of Shakti to 90
13    new_dist["Shakti"] = 90
14
15     #Printing all student grades
16    print("Student Grades:", new_dist)
17
18     #Printing the grade of a particular student using if -else
19    print("Enter the name of the student to check the grade:")
20    student=input()
21    grade=new_dist[student]
22
23    if grade==90:
24        grade='A'
25    elif grade==80:
26        grade='B'
27    elif grade>=70:
28        grade='C'
29    elif grade>=60:
30        grade='D'
31    else:
32        grade='E'
33
34    print("\n",student,":",grade)
35

shaktisingh@shaktisingh-Vostro-3500:~/learnpy$ /bin/python3 "/home/shaktisingh/learnpy/task/#_student_Grades.py"
vina : A
shaktisingh@shaktisingh-Vostro-3500:~/learnpy$ /bin/python3 "/home/shaktisingh/learnpy/task/#_student_Grades.py"
File "/home/shaktisingh/learnpy/task/#_student_Grades.py", line 23
    new_dist["Shakti"]
IndentationError: unexpected indent
shaktisingh@shaktisingh-Vostro-3500:~/learnpy$ /bin/python3 "/home/shaktisingh/learnpy/task/#_student_Grades.py"
Student Grades: {'Shakti': 90, 'Rohan': 92, 'Anita': 78, 'Priya': 88, 'sia': 95, 'vina': 98}
Enter the name of the student to check the grade:
Shakti
Shakti : A
shaktisingh@shaktisingh-Vostro-3500:~/learnpy$
```

### 3. Write to a File

Write a program to create a text file and write some content to it.

Using file function like write and open.

A screenshot of a code editor interface, likely VS Code, showing a project structure and a terminal window. The project structure on the left includes files like Grade\_checker.py, #student\_Grades.py, file.py, and data.txt. The main editor area contains a Python script named file.py with the following code:

```
task > file.py > ...
1 #file
2 file=open("data.txt", "w")
3 file.write("Hello, World!")
4 file.write("\n Learning the Devops and Python")
5 file.close()
6
7 file=open("data.txt", "r")
8 data=file.read()
9
10 print("file data is :",data)
11 file.close()
12
13 print("file data is :")
14 if file.closed:
15     print("file is closed")
16 else:
17     print("File is open")
18
```

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

```
shaktisingh@shaktisingh-Vostro-3500:~/learnpy$ ./bin/python3 /home/shaktisingh/learnpy/task/file.py
Hello, World!
Learning the Devops and Python
file is closed
shaktisingh@shaktisingh-Vostro-3500:~/learnpy$
```

## 4. Read from a File

We used open in read mode and file.read to read and print to display.

A screenshot of a code editor interface, likely VS Code, showing a project structure and a terminal window. The project structure on the left includes files like Grade\_checker.py, #student\_Grades.py, file.py, and data.txt. The main editor area contains a Python script named #Read\_file.py with the following code:

```
task > #Read_file.py > ...
1 #Read file
2 file=open("data.txt","r")
3 data=file.read()
4 print("file data is :",data)
5 file.close()
6
7
```

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

```
shaktisingh@shaktisingh-Vostro-3500:~/learnpy$ ./bin/python3 /home/shaktisingh/learnpy/task/#Read_file.py
file data is : Hello, World!
Learning the Devops and Python
file data is : Hello, World!
Learning the Devops and Python
File is closed
shaktisingh@shaktisingh-Vostro-3500:~/learnpy$
```