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.

# grade\_checker.py

score = int(input("Enter the score: "))

if score >= 90:

grade = "A"

elif score >= 80:

grade = "B"

elif score >= 70:

grade = "C"

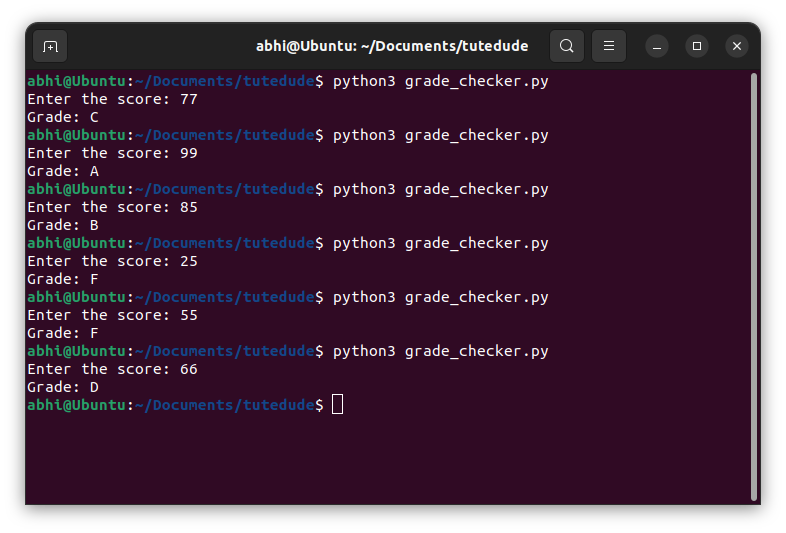
elif score >= 60:

grade = "D"

else:

grade = "F"

print("Grade:", grade)



**Explanation**

* The script takes a numeric score.
* Uses if-elif-else statements to compare ranges.
* Prints the corresponding grade.

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.

# student\_grades.py

students = {}

while True:

print("\nOptions:")

print("1. Add student")

print("2. Update student grade")

print("3. Print all grades")

print("4. Exit")

choice = input("Enter choice: ")

if choice == "1":

name = input("Enter student name: ")

grade = input("Enter grade: ")

students[name] = grade

print(f"Added {name} with grade {grade}.")

elif choice == "2":

name = input("Enter student name to update: ")

if name in students:

grade = input("Enter new grade: ")

students[name] = grade

print(f"Updated {name}'s grade to {grade}.")

else:

print("Student not found.")

elif choice == "3":

print("Student Grades:")

for name, grade in students.items():

print(f"{name}: {grade}")

elif choice == "4":

print("Exiting...")

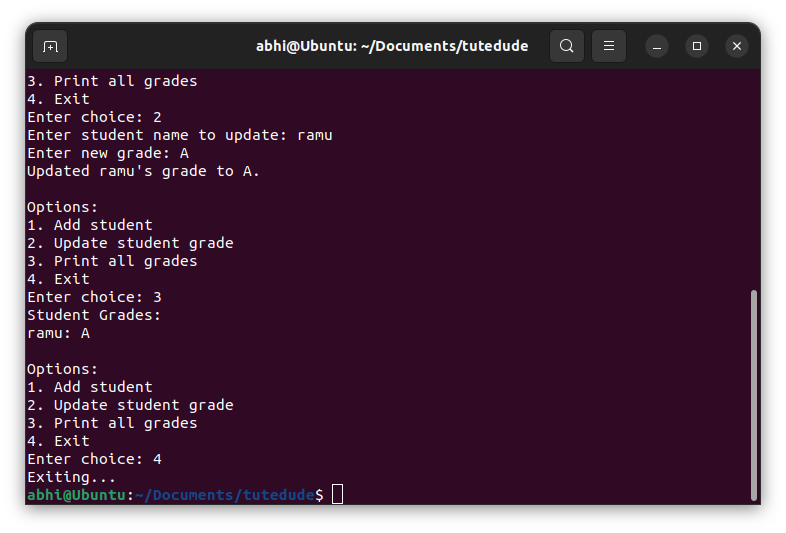
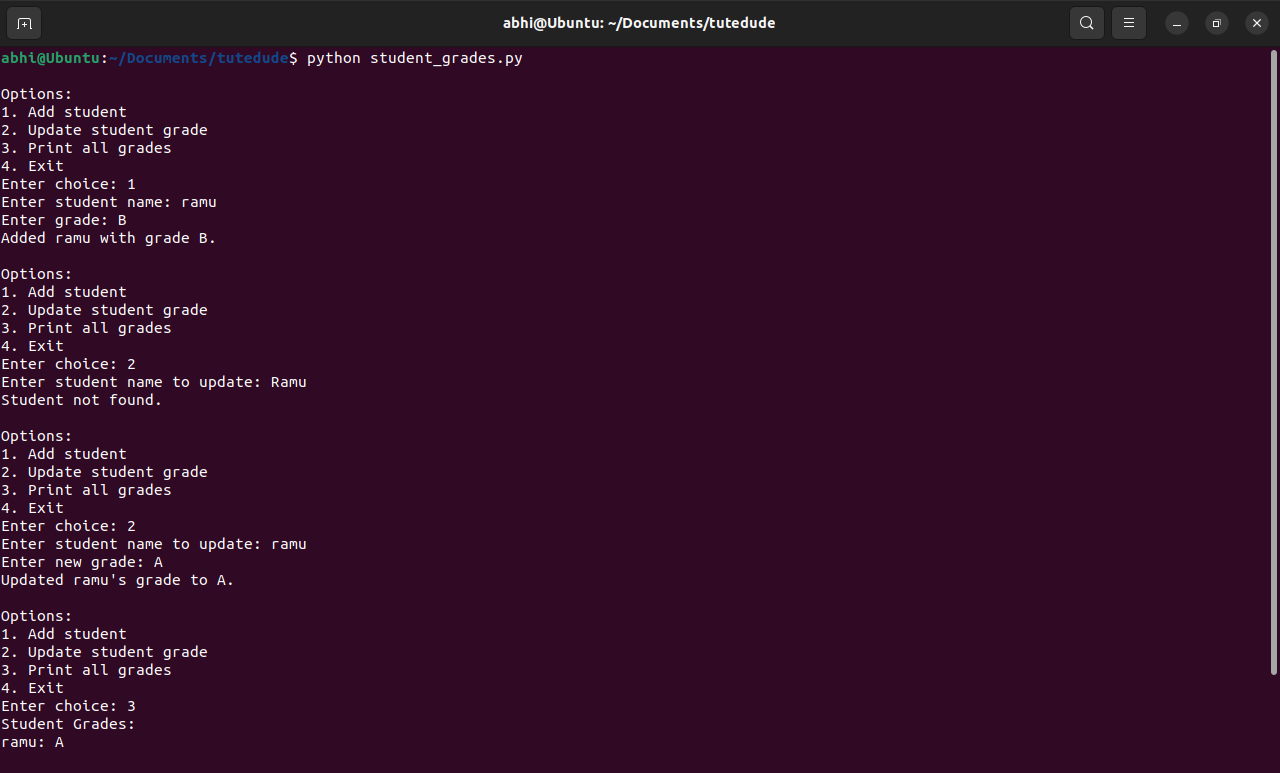
break

else:

print("Invalid choice.")

**Explanation**

* A dictionary students holds student names and grades.
* Menu-driven program allows:
  + Adding students
  + Updating grades
  + Printing all grades
  + Exiting the program
* Uses if-else to process menu options.



Used dictionary and basic operations. Using if else:

3.Write to a File

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

# write\_file.py

filename = "myfile.txt"

content = "This is some example text written to the file."

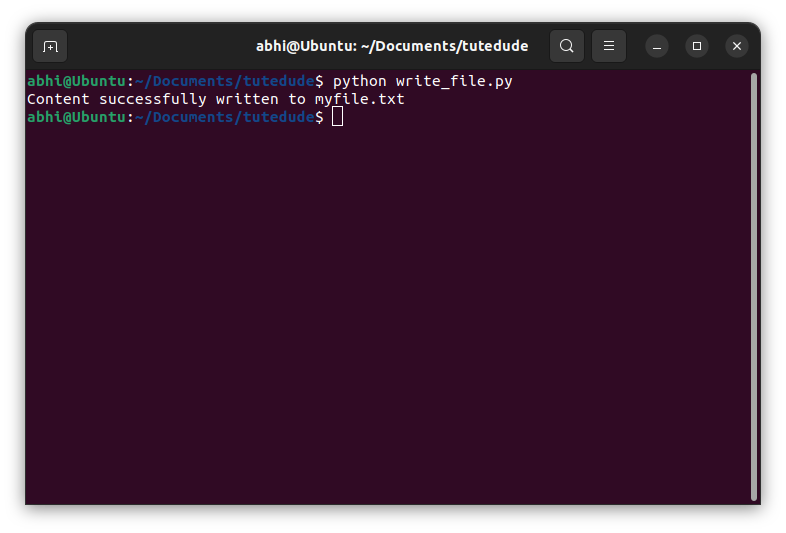
with open(filename, "w") as file:

file.write(content)

print("Content successfully written to", filename)

**Explanation**

* Opens myfile.txt in **write mode**.
* Writes the specified text.
* with context auto-closes the file.



Using file functions like write and open.

4. Read from a File

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

# read\_file.py

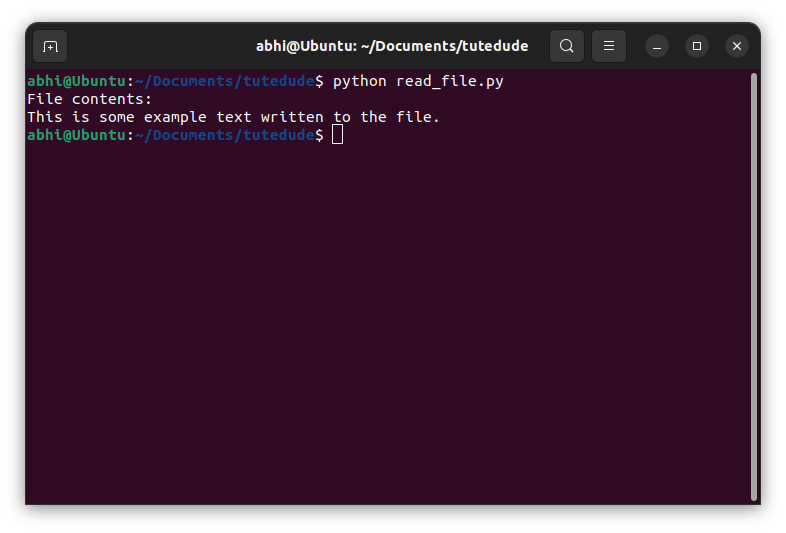
filename = "myfile.txt"

with open(filename, "r") as file:

content = file.read()

print("File contents:")

print(content)



**Explanation**

* Opens myfile.txt in **read mode**.
* Reads the content and prints it to the terminal.

[HAbhishekBhat/TuteDude---Devops: This repository presents the comprehensive solutions for the DevOps course assessment provided by Tutedude. Each answer has been solved practically and verified, accompanied by a relevant screenshot and explanation in accordance with the guidelines provided during the course.](https://github.com/HAbhishekBhat/TuteDude---Devops)