Github Repo Link - github.com/Standby-Coder/DevOpsTraining

1.

```
PS F:\DevOpsTraining\Python and Bash\scripts> python 1.py
Enter your score: 100
Your score is: A
PS F:\DevOpsTraining\Python and Bash\scripts> python 1.py
Enter your score: 91
Your score is: A
PS F:\DevOpsTraining\Python and Bash\scripts> python 1.py
Enter your score: 81
Your score is: B
PS F:\DevOpsTraining\Python and Bash\scripts> python 1.py
Enter your score: 90
Your score is: A
PS F:\DevOpsTraining\Python and Bash\scripts> python 1.py
Enter your score: abc
Error: invalid literal for int() with base 10: 'abc'
```

```python
def getScore(score : int) -> str:
    if score < 0 or score > 100:
        raise ValueError("Score must be between 0 and 100")
    elif score >= 90:
        return "A"
    elif score >= 80:
        return "B"
    elif score >= 70:
        return "C"
    elif score >= 60:
        return "D"
    else:
        return "F"

if __name__ == "__main__":
    x = input( "Enter your score: ")
    try:
        score = getScore(int(x))
        print("Your score is: " + score)
    except ValueError as e:
        print("Error: " + str(e))
```

2.

```python
def add():
    try:
        name = input("Enter student's name: ")
        grade = float(input("Enter student's grade: "))
        if name in gradeList:
            raise ValueError("Student already exists.")
        if not (0 <= grade <= 100):
            raise ValueError("Grade must be between 0 and 100.")
        gradeList[name] = grade
        print(f"Added {name} with grade {grade}.")
    except ValueError as e:
        print("Error: " + str(e))

def updateGrade():
    try:
        name = input("Enter student's name to update grade: ")
        if name not in gradeList:
            raise ValueError("Student does not exist.")
        new_grade = float(input("Enter new grade: "))
        if not (0 <= new_grade <= 100):
            raise ValueError("Grade must be between 0 and 100.")
        gradeList[name] = new_grade
        print(f"Updated {name}'s grade to {new_grade}.")
    except ValueError as e:
        print("Error: " + str(e))

def viewAll():
    if not gradeList:
        print("No students found.")
    else:
        print("Students and their grades:")
        for name, grade in gradeList.items():
            print(f"{name}: {grade}")
```

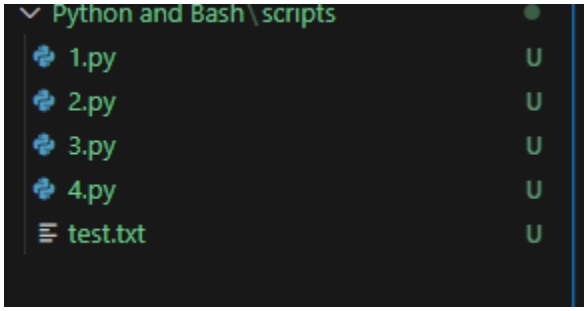
```python
if __name__ == "__main__":
    gradeList = {}
    while True:
        try:
            print("-" * 20)
            print("Enter a choice:")
            print("1. Add new student and grade")
            print("2. Update student's grade")
            print("3. View all students and grades")
            print("4. Exit")
            choice = int(input("Choice: "))
            if choice == 1:
                add()
            elif choice == 2:
                updateGrade()
            elif choice == 3:
                viewAll()
            elif choice == 4:
                print("Exiting the program.")
                break
            else:
                print("Invalid choice. Please try again.")
        except ValueError as e:
            print("Error: " + str(e))
```

```
PS F:\DevOpsTraining\Python and Bash\scripts> python 2.py
--------------------
Enter a choice:
1. Add new student and grade
2. Update student's grade
3. View all students and grades
4. Exit
Choice: 1
Enter student's name: Daniel
Enter student's grade: 92
Added Daniel with grade 92.0.
--------------------
Enter a choice:
1. Add new student and grade
2. Update student's grade
3. View all students and grades
4. Exit
Choice: 1
Enter student's name: Kaley
Enter student's grade: 81
Added Kaley with grade 81.0.
--------------------
Enter a choice:
1. Add new student and grade
2. Update student's grade
3. View all students and grades
4. Exit
Choice: 2
Enter student's name to update grade: Kaley
Enter new grade: 92
Updated Kaley's grade to 92.0.
--------------------
Enter a choice:
1. Add new student and grade
2. Update student's grade
3. View all students and grades
4. Exit
Choice: 3
Students and their grades:
Daniel: 92.0
Kaley: 92.0
--------------------
Enter a choice:
1. Add new student and grade
2. Update student's grade
3. View all students and grades
4. Exit
Choice: 4
Exiting the program.
```

3.

```python
def writeToFile():
    with open("test.txt", "w") as f:
        f.write(input("Enter text to write to file: "))

if __name__ == "__main__":
    try:
        writeToFile()
        print("Text written to file successfully.")
    except Exception as e:
        print("An error occurred: " + str(e))
```




```
PS F:\DevOpsTraining\Python and Bash\scripts> python 3.py
Enter text to write to file: This is the answer to the 3rd assignment uestion
Text written to file successfully.
```

4.

```python
def readFromFile(filename):
    try:
        with open(filename, 'r') as file:
            content = file.read()
        return content
    except FileNotFoundError:
        return "File not found."
    except IOError:
        return "Error reading file."

if __name__ == "__main__":
    try:
        filepath = input("Enter the file path to read from: ")
        content = readFromFile(filepath)
        print("Content read from file: " + content)
    except Exception as e:
        print("An error occurred: " + str(e))
```

```
PS F:\DevOpsTraining\Python and Bash\scripts> python 4.py
Enter the file path to read from: ./test.txt
Content read from file: This is the answer to the 3rd assignment uestion
```