**Exercise 7**

**Aim**

To develop a student management system using inheritance to manage student details, marks, and grades.

**Algorithm**

|  |  |  |
| --- | --- | --- |
| Step 1 | **:** | Start the Program. |
| Step 2 | **:** | Define the Base Class. |
| Step 3 | **:** | Define Derived Classes. |
| Step 4 | **:** | Create Student Instances. |
| Step 5 | **:** | Input Marks for Undergraduate Student. |
| Step 6 | **:** | Input Marks for Graduate Student. |
| Step 7 | **:** | Display Student Academic Information. |
| Step 8 | **:** | Stop the Program. |

**Program:**

# Class to store and handle personal details of a student

class StudentDetails:

def \_\_init\_\_(self, name, student\_id, programme):

self.name = name

self.student\_id = student\_id

self.programme = programme # Store the programme info

def student\_info(self):

return f"Student ID: {self.student\_id}\nName: {self.name}\nProgramme: {self.programme}"

# Class to handle academic details like marks and grades

class Student:

def \_\_init\_\_(self, student\_details):

self.student\_details = student\_details # Composition: Student has a StudentDetails object

self.marks = {}

def add\_marks(self, subject, marks):

self.marks[subject] = marks

# Removed the print statement here

def calculate\_grade(self):

total\_marks = sum(self.marks.values())

num\_subjects = len(self.marks)

if num\_subjects == 0:

return "No marks available"

percentage = total\_marks / num\_subjects

if percentage >= 75:

return "A"

elif percentage >= 60:

return "B"

elif percentage >= 50:

return "C"

elif percentage >= 40:

return "D"

else:

return "F"

def student\_info(self):

# Get student's personal details from the StudentDetails object

personal\_info = self.student\_details.student\_info()

# Add academic details like marks and grade

return f"{personal\_info}\nMarks: {self.marks}\nGrade: {self.calculate\_grade()}"

# Main function to gather input and create student instances

def main():

# Collect Student Details

print("Enter Student Details:")

name = input("Enter Name: ")

student\_id = input("Enter Student ID: ")

programme = input("Enter Programme (UG/PG): ") # Ask user for programme (UG or PG)

student\_details = StudentDetails(name, student\_id, programme)

# Create Student instance for academic details

student = Student(student\_details)

# Get marks for the student

num\_courses = int(input("How many courses' marks should be entered?: "))

for i in range(1, num\_courses + 1):

marks = float(input(f"Enter the course {i} mark: ")) # Prompt for each course mark

subject = f"Course {i}" # Automatically name the subject as "Course X"

student.add\_marks(subject, marks)

# Display student information

print("\nStudent Academic Information")

print("----------------------------")

print(student.student\_info()) # Display complete student info

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Output**

**Result:**

The student management system was successfully developed using inheritance, which displayed student information and computed grades