Software Design and Development

Assessment Report

**Task 1 Evidence:**

**Add your algorithm below:**

1.0 **Input a Student’s Coursework Mark:**

1.1 Declare variable courseworkMark and assign to -1.

1.2 While courseworkMark is less than 0 or greater than 60:

1.3 Prompt the user to enter coursework mark.

1.4 Convert input to floating point number and assign to courseworkMark.

2.0 **Input Prelim Mark:**

2.1 Declare variable prelimMark and assign to -1.

2.2 While prelimMark is less than 0 or greater than 90:

2.3 Prompt the user to enter prelim mark.

2.4 Convert input to floating point number and assign to prelimMark.

3.0 **Calculate Mark Percentage:**

3.1 Calculate the mark percentage as the integer value of the sum of courseworkMark and prelimMark multiplied by 100 divided by 150.

4.0 **Determine Grade:**

4.1 If mark percentage is greater than or equal to 70:

4.2 Display student achieved grade A.

4.3 Else if mark percentage is between 60 and 69:

4.4 Display student achieved grade B.

4.5 Else if mark percentage is between 50 and 59:

4.6 Display student achieved grade C.

4.7 Else if mark percentage is between 45 and 49:

4.8 Display student achieved grade D.

4.9 Else no grade awarded.

START

1.1 INITIALISE courseworkMark to -1

1.2 WHILE 0 > courseworkMark > 60  
1.3 DISPLAY "Enter coursework mark: "  
1.4 INPUT courseworkMark

2.1 INITIALISE prelimMark to -1

2.2 WHILE 0 > prelimMark > 90

2.3 DISPLAY "Enter prelim mark: "

2.4 INPUT prelimMark

3.1 SET markPercentage to integer value of (courseworkMark + prelimMark) \* 100 / 150.

4.1 IF markPercentage >= 70:

4.2 DISPLAY "Student achieved grade A with total mark of”, markPercentage

4.3 ELSE IF markPercentage >= 60:

4.4 DISPLAY "Student achieved grade B with total mark of", markPercentage

4.5 ELSE IF markPercentage >= 50:

4.6 DISPLAY "Student achieved grade C with total mark of", markPercentage

4.7 ELSE IF markPercentage >= 45:

4.8 DISPLAY "Student achieved grade D with total mark of", markPercentage

4.9 ELSE:

4.10 DISPLAY "Student failed to achieve a grade with total mark of", markPercentage

**Task 2 Evidence:**

**Using the table below complete the test plan for the program:**

**Note: You may need to add additional rows in order to show a complete set of tests.**

|  |  |  |  |
| --- | --- | --- | --- |
| Function Name | Reason for test (valid/invalid – Normal, exceptional, extreme etc.) | Test Data | Expected Output |
| coursework | Normal | 40 | Data accepted |
| coursework | Extreme | 0 | Data accepted |
| coursework | Extreme | 60 | Data accepted |
| coursework | Exceptional | 65 | Data rejected |
| coursework | Exceptional | Sixteen | Data rejected |
| prelim | Normal | 50 | Data accepted |
| prelim | Extreme | 0 | Data accepted |
| prelim | Extreme | 90 | Data accepted |
| prelim | Exceptional | 100 | Data rejected |
| prelim | Exceptional | Seventy | Data rejected |
| calculate\_percentage | Normal | 40, 80 | Data accepted |
| calculate\_percentage | Extreme | 0, 0 | Data accepted |
| calculate\_percentage | Extreme | 60, 90 | Data accepted |
| Calculate\_percentage | Exceptional | Twenty, Sixty | Data rejected |
| display\_grade | Normal | 20, 30 | Data accepted |
| display\_grade | Extreme | 0, 0 | Data accepted |
| display\_grade | Extreme | 60, 90 | Data accepted |
| display\_grade | Exceptional | Thirty, 200 | Data rejected |

**Task 3 and 4 Evidence:**

**Copy/ paste your source code and add below: Make sure to add internal commentary asked for.**

# Author: Gary Siu  
# Date: 5th Feb 2024  
# Description: Calculate student grade from coursework and prelim marks  
  
# Function to return coursework marks  
def coursework():  
 # Initialise variable  
 mark = -1  
 # Loop until coursework mark within limits  
 while mark < 0 or mark > 60:  
 # Catch ValueErrors such as strings  
 try:  
 # Input student coursework mark  
 mark = float(input("Enter coursework mark: "))  
 except ValueError:  
 print("Please use numbers.")  
 return mark  
  
  
# Function to return prelim marks  
def prelim():  
 # Initialise variable  
 mark = -1  
 # Loop until prelim mark within limits  
 while mark < 0 or mark > 90:  
 # Catch ValueErrors such as strings  
 try:  
 # Input student prelim mark  
 mark = float(input("Enter prelim mark: "))  
 except ValueError:  
 print("Please use numbers.")  
 return mark  
  
  
# Function to calculate mark percentage  
def calculate\_percentage(coursework\_mark, prelim\_mark):  
 # Use coursework and prelim marks  
 return int(((coursework\_mark + prelim\_mark) \* 100) / 150)  
  
  
# Function to return student grade  
def display\_grade(coursework, prelim):  
 # Pass in marks to calculate\_percentage function  
 mark\_percentage = calculate\_percentage(coursework, prelim)  
 # IF conditionals to determine grade achieved  
 if mark\_percentage >= 70:  
 return f"Student achieved grade A with total mark of **{**mark\_percentage**}**"  
 elif mark\_percentage >= 60:  
 return f"Student achieved grade B with total mark of **{**mark\_percentage**}**"  
 elif mark\_percentage >= 50:  
 return f"Student achieved grade C with total mark of **{**mark\_percentage**}**"  
 elif mark\_percentage >= 45:  
 return f"Student achieved grade D with total mark of **{**mark\_percentage**}**"  
 else:  
 return (f"Student failed to achieve a grade with total mark of "  
 f"**{**mark\_percentage**}**")  
  
  
def main():  
 # Call functions within main  
 coursework\_mark = coursework()  
 prelim\_mark = prelim()  
 # Pass values returned from coursework and prelim functions into  
 # display\_grade() and display the result returned  
 print(display\_grade(coursework\_mark, prelim\_mark))  
  
  
# Call main to start the program  
main()

**Task 5 Evidence:**

**Use the table below to complete the testing outlined in above test plan.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function Name | Reason for test (valid/invalid – Normal, exceptional, extreme etc.) | Test Data | Expected Output | Actual Output | Comments |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**If any of the above tests failed, then outline how you fixed it and rerun the test to show that it now works correctly (include screenshot below to show this).**

**Task 6 Evidence:**

**Copy/ paste your source code (read from file included) and add below: Make sure to add internal commentary asked for.**

**Task 7 Evidence:**

**Copy/ paste your source code (count occurences / max algorithm now included) and add below: Make sure to add internal commentary asked for.**

**Task 8 Evidence**

**Write your description of the fetch execute cycle below: You may include a diagram as well.**