

GHANA COMMUNICATION

TECHNOLOGY UNIVERSITY

INSTITUTE OF CONTINUING

AND DISTANCE EDUCATION (ICDE)

|  |  |
| --- | --- |
| COURSE CODE | CICS 112 |
| COURSE TITLE | Programming with C++ |
| DATE | 10th September, 2025 |
| GROUP NAME | GROUP 13 |

|  |  |
| --- | --- |
| **INDEX NUMBER** | **NAME** |
| 2425140073 | Francklina Delali Mensah |
| 2425140016 | RUNCIE FRIMPONG |
| 2425140023 | AGBENYO DELATOR ROGERS |
| 2425140099 | Darglo Isaac Mawuli |
| 2425140125 | Emmanuel Koomson |

**Objective 1**

Write a C++ program that calculates the average of marks obtained in 5 subjects and assigns a grade based on the average. The program must include input validation to ensure that each mark entered is between 0 and 100.

**Solution**

#include <iostream>

using namespace std;

int main()

{float marks[5];

float sum = 0.0;

float average;

string grade;

for (int i=0; i<5; i++){

do {

cout << "Enter mark for subject" << (i+1) <<":";

cin >> marks[i];

if (marks[i]<0|| marks[i]> 100){

cout <<"invalid mark. please enter a mark between 0 and 100."<<endl;

}

} while (marks[i],0 || marks[i]> 100);

sum += marks[i];

}

average = sum/5;

if (average >= 91 && average <= 100){

grade = "A1";

} else if (average >= 81 && average <= 90){

grade = "A2";

} else if (average >= 71 && average <= 80) {

grade = "B1";

} else if (average >= 61 && average <= 70){

grade = "B2";

} else if (average >= 51 && average <= 60){

grade = "C1";

} else if (average >= 41 && average <= 50){

grade = "C2";

} else if (average >= 33 && average <= 40){

grade = "D";

} else if (average >= 21 && average <= 32){

grade = "E1";

} else if (average >= 0 && average <= 20){

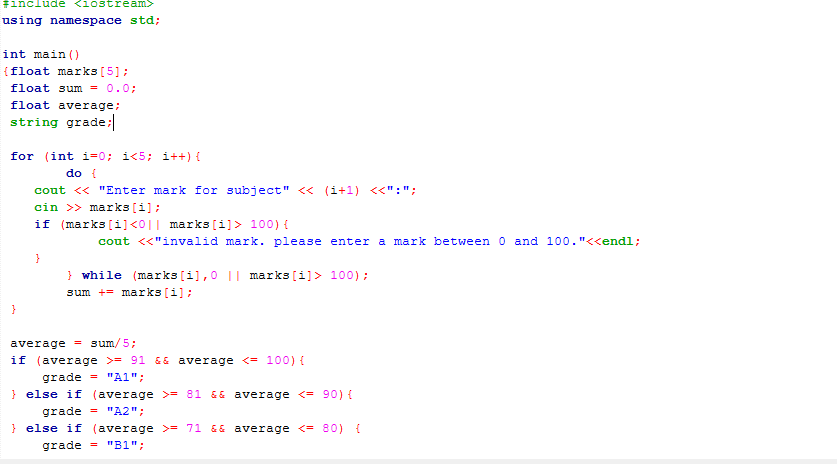
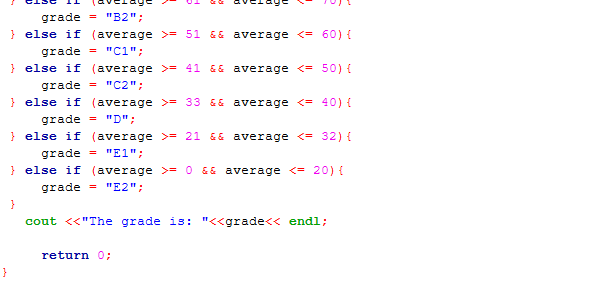
grade = "E2";

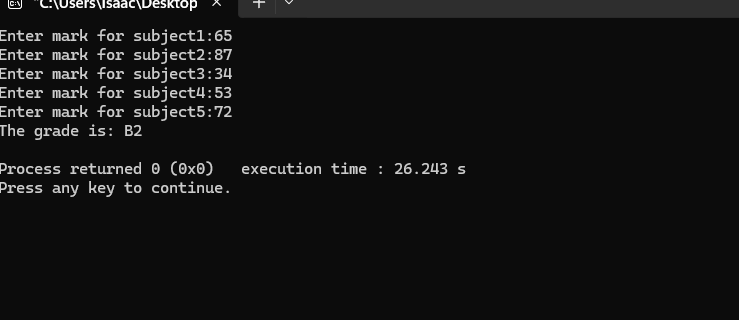
}

cout <<"The grade is: "<<grade<< endl;

return 0;

}



**Objective 2:**

1. What is the purpose of input validation in programming?

Input validation serves to ensure that data entered into a program meets specific criteria before processing. It prevents errors, security vulnerabilities, and unexpected program behavior by checking that inputs are in the correct format, within acceptable ranges, and safe to use. This protects against malicious attacks like SQL injection and buffer overflows while improving user experience through clear error messages.

2. Why is a do-while loop suitable for this task?

A do-while loop is particularly suitable for input validation scenarios because it guarantees that the code block executes at least once before checking the condition. This means the program will always prompt the user for input initially, then continue prompting until valid input is received. The loop structure naturally handles the “ask-validate-repeat if invalid” pattern common in user input scenarios.

3. How could you improve this program to handle more complex grading systems?

To handle more complex grading systems, you could implement several enhancements: use arrays or data structures to store multiple grade boundaries and corresponding letter grades; implement weighted categories for different assignment types; add support for plus/minus grades with more granular score ranges; create functions to modularize the grading logic; add file I/O capabilities to read grading scales from configuration files; implement exception handling for edge cases; and create a class-based structure to manage different grading policies for various courses or institutions.​​​​​​​​​​​

Objective 3:

C++ Programming Task: Simple Interest Calculator Objective: Write a C++ program that calculates simple interest using a separate function. Program Requirements: Your program should:

1. Prompt the user to enter: a. Principal amount b. Time (in years) c. Rate of interest.

2. Use a separate function named calculateInterest to compute the simple interest using the formula: Interest=P×T×R100Interest=100P×T×R 3. Display the result to the user.

**Solution**

#include <iostream>

#include <iomanip>

using std::cin;

using std::cout;

using std::fixed;

using std::setprecision;

double calcInterest(double principal, double duration, double rate)

{

return (principal \* duration \* rate) / 100;

}

int main()

{

double principal, duration, rate;

cout << "\n\n=======================================================\n";

cout << " SIMPLE INTEREST RATE CALCULATOR \n";

cout << "=======================================================\n\n";

do

{

cout << "Enter Principal Amount (positive value): ";

cin >> principal;

if (principal <= 0)

{

cout << "Error: Principal must be a positive value. Try again.\n";

}

} while (principal <= 0);

do

{

cout << "Enter Duration Amount (in years, positive value): ";

cin >> duration;

if (duration <= 0)

{

cout << "Error: Duration must be a positive value. Try again.\n";

}

} while (duration <= 0);

do

{

cout << "Enter Rate Amount (positive value): ";

cin >> rate;

if (rate <= 0)

{

cout << "Error: Rate must be a positive value. Try again.\n";

}

} while (rate <= 0);

double interest = calcInterest(principal, duration, rate);

double totalAmount = principal + interest;

cout << "\n\n=======================================================\n\n";

cout << fixed << setprecision(2);

cout << "Principal Amount: GHc " << principal << "\n";

cout << "Duration: " << duration << " years\n";

cout << "Interest Rate: " << rate << "%\n";

cout << "Interest: GHc " << interest << "\n";

cout << "Total Amount: GHc " << totalAmount << "\n\n";

cout << "=======================================================\n\n";

return 0;

}

