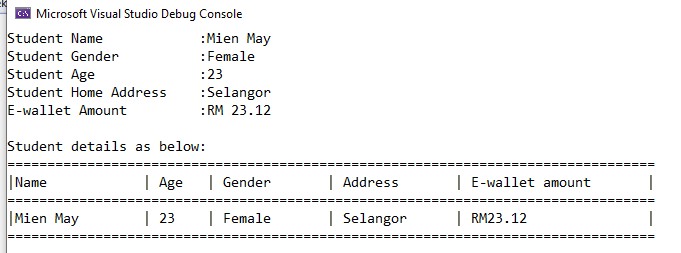
**Lab 1: Getting Started with C++**

**Part A**: Learn how to use the cin and cout in C++.

1. Write a program that to display the following output:



*[*

*Estimate Finish Time:*

*20*

*minutes]*

First, let’s define a struct for representing the Student.

struct Student {

    string name;

    bool gender; *// 0 for male, 1 for female*

    int age;

    string homeAdress;

    double eWalletAmount;

};

Then, let’s define a function to print the information in an organized manner.

void printStudentDetails(Student student) {

    cout << "Student details as below:\n";

    vector<string> colNames = {

        "Name", "Age", "Gender",

        "Address", "E-wallet amount"

    };

    vector<int> colWidths = {

        (int)colNames.at(0).length()\*3,

        (int)colNames.at(1).length()\*2,

        (int)colNames.at(2).length()\*2,

        (int)colNames.at(3).length()\*2,

        (int)(colNames.at(4).length()\*1.5),

    };

    int totalLength = 0;

    for(int i=0;i<colWidths.size();i++)

        totalLength+=colWidths.at(i)+1; *// +1 for the '|' char*

*// Print horizontal bar*

    for(int i=0;i<totalLength;i++)

        cout << "=";

    cout << "="; *// Extra char on top of the last '|' char*

    cout<<endl;

*// Print column names*

    for(int i=0;i<colNames.size();i++) {

        cout << "|" << colNames.at(i);

*// Add whitespace padding*

        for(int pad=0;

            pad<colWidths.at(i)-colNames.at(i).length();

            pad++) {

            cout<<" ";

        }

    }

    cout<<"|\n"; *// print finishing char*

*// Print horizontal bar*

    for(int i=0;i<totalLength;i++)

        cout << "=";

    cout << "="; *// Extra char on top of the last '|' char*

    cout<<endl;

*// Print student data*

    for(int i=0;i<colNames.size();i++) {

        string data;

        switch(i) {

            case 0: { data=student.name; break; } *// Name*

            case 1: { data= to\_string(student.age); break; } *// Age*

            case 2: { data=student.gender? "Male" : "Female"; break; } *// Gender*

            case 3: { data=student.homeAdress; break; } *// Home Address*

            case 4: { data=to\_string(student.eWalletAmount); break; } *// E-wallet Amount*

            default: break;

        }

        cout << "|" << data;

*// Add whitespace padding*

        for(int pad=0;

            pad<colWidths.at(i)-data.length();

            pad++) {

            cout<<" ";

        }

    }

    cout<<"|\n"; *// print finishing char*

*// Print horizontal bar*

    for(int i=0;i<totalLength;i++)

        cout << "=";

    cout << "="; *// Extra char on top of the last '|' char*

    cout<<endl;

}

Finally, we create a function to call from the main function, giving some dummy data for testing.

void partA\_1() {

    Student tststudent;

    tststudent.age=23;

    tststudent.eWalletAmount=45.00;

    tststudent.gender=0;

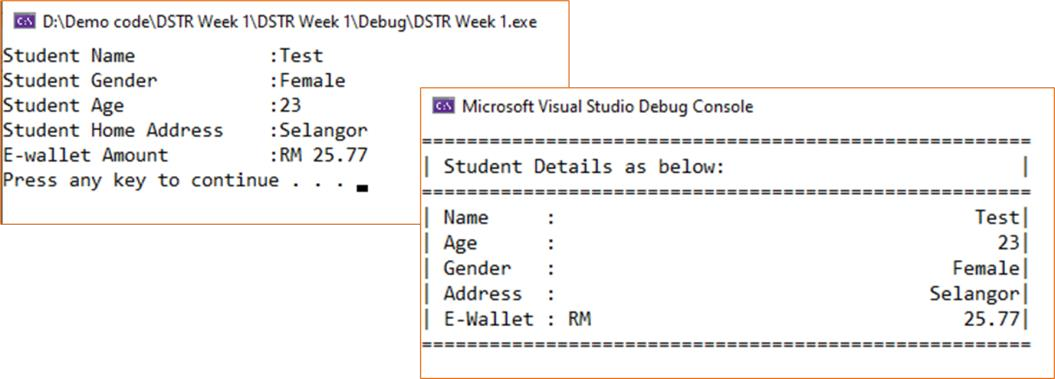
    tststudent.homeAdress="Selangor";

    tststudent.name="Mohammed";

    printStudentDetails(tststudent);

}

1. Modify the program in Question 1 to get another output as below:

 *[Estimate Finish Time: 15 minutes]*

First, we will add another function to input the data into a Student struct from the command line:

Student getStudentDetails() {

    int inputPos = 50; *// offset where the ':' sign is displayed*

    vector<string> inputNames = {

        "Student Name", "Student Age",

        "Student Gender", "Student Home Address",

        "E-wallet Amount"

    };

    Student output;

    bool receivingInput = true;

    bool invalidInput = true;

    int inputIndex = 0;

    while(receivingInput) {

        if(invalidInput)

            cout << "Invalid input, try again!\n";

        invalidInput=false;

        cout << inputNames.at(inputIndex);

*// Print padding*

        for(int i=0;

        i<inputPos-inputNames.at(inputIndex).length();

        i++) cout << " ";

        cout<<":";

        fflush(stdin);

        switch(inputIndex) {

*// Name*

            case 0: {cin>>output.name; break;}

*// Age*

            case 1: {cin>>output.age; break;}

*// Gender*

            case 2: {

                string input;

                cin>>input;

                output.gender = ("Male"==input||"male"==input);

                output.gender = !("Female"==input||"female"==input);

*// If the gender is not recognized, it should automatically*

*// set to "Male" (0)*

                break;

            }

*// Home Address*

            case 3: {

                string input;

                cin >> input;

                output.homeAdress = input;

                break;

            }

*// E-wallet amount*

            case 4: {

                cin>>output.eWalletAmount;

                break;

            }

            default: {

                invalidInput=true;

                break;

            }

        }

        if(!invalidInput) {

            if(4==inputIndex) receivingInput=false;

            else inputIndex++;

        }

    }

    return output;

}

Then, we create a function for getting and printing the data:

void partA\_2() {

    Student tstStudent;

    tstStudent = getStudentDetails();

    printStudentDetails(tstStudent);

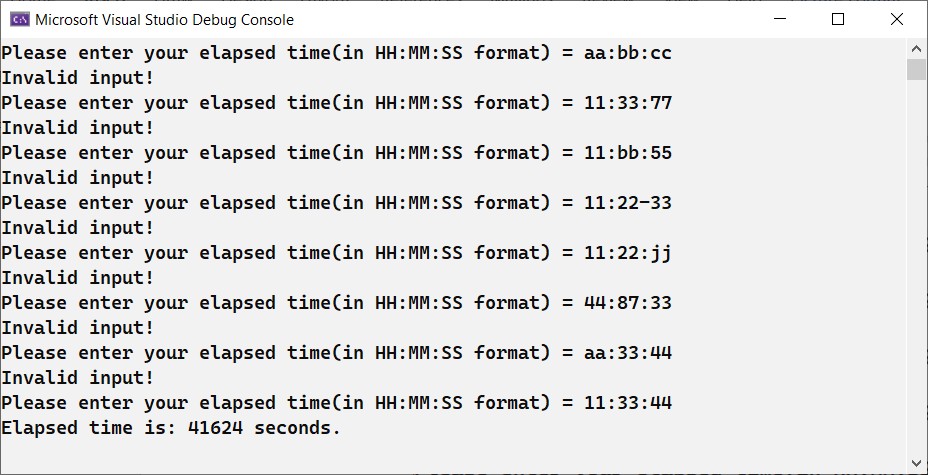
}

**Part B**: Learn how to use the Control statements in C++.

1. A program is required to prompt for and accept a time and compute the number of seconds elapsed since midnight. The time should be entered in the format HH:MM:SS. Include some input validations in your program.

**Output sample:**

Please enter your elapsed time (in HH:MM:SS format) = 00:02:44 Elapsed time in seconds = 164 seconds



*[Estimate Finish Time: 30 minutes]*

void partB\_1() {

    int elapsedSeconds;

    bool takingInput = true;

    bool invalidInput = false;

    while(takingInput) {

        fflush(stdin); *// clear the input buffer*

        if(invalidInput)

            std::cout << "Invalid Input!\n";

*// Get elapsed time in HH:MM:SS*

        string input;

        cout << "Please enter yout elapsed time (in HH:SS:MM format) = ";

        cin >> input;

*// Validate input*

        invalidInput =

            (input.length() != 8);

*// debug*

        cout << "input length = " << input.length() << endl;

        for(int i=0; i<input.length(); i++) {

*// Check the ':' signs*

            if(i==2 || i==5) {

                if(':'!=input.at(i)) {

                    std::cout<< "Not a valid separator at i = " << i << endl;

                    invalidInput = true;

                }

            }

*// Make sure that the digits are decimal*

            else {

                if(!isdigit(input.at(i))) {

                    std::cout<< "Not a valid digit at i = " << i << endl;

                    invalidInput = true;

                }

            }

        }

*// Go to next iteration if invalid input*

        if(invalidInput) continue;

*// Parse input*

        int hours, minutes, seconds;

        stringstream sstr;

*// Parse Hours*

        sstr << input[0]; sstr << input[1];

        hours = stoi(sstr.str());

        cout << "Hours = " << hours << " ; ";

        sstr.str("");

        sstr.clear();

*// Parse Minutes*

        sstr << input[3]; sstr << input[4];

        minutes = stoi(sstr.str());

        cout << "Minutes = " << minutes << " ; ";

        sstr.str("");

        sstr.clear();

*// Parse Seconds*

        sstr << input[6]; sstr << input[7];

        seconds = stoi(sstr.str());

        cout << "Seconds = " << seconds << endl;

        sstr.str("");

        sstr.clear();

        elapsedSeconds = (3600\*hours) + (60\*minutes) + seconds;

*// No errors, finish input loop*

        if(!invalidInput) takingInput=false;

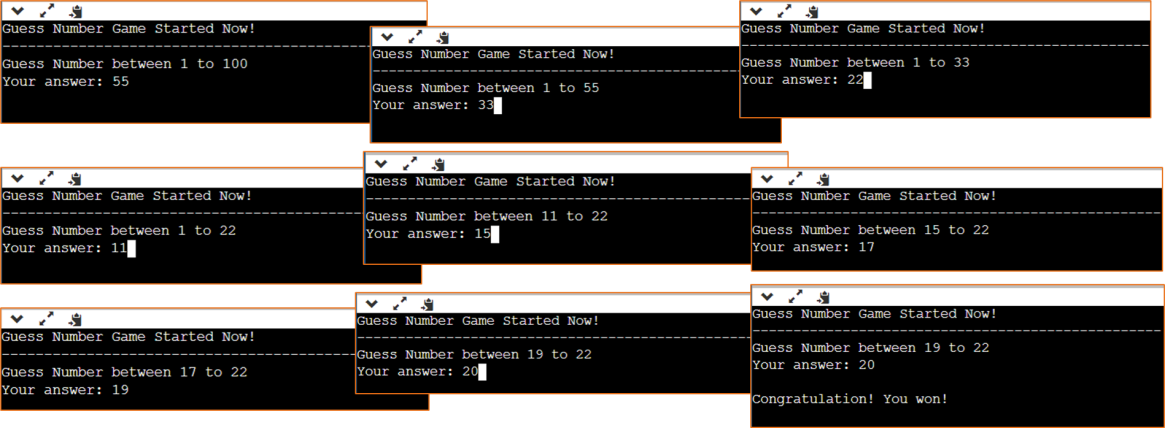
    }

*// Output elapsed time in Seconds*

    std::cout << "Elapsed time is: " << elapsedSeconds << " second(s).\n";

}

1. Create a simple game in C++ that can let the user guess a random number which secretly selected from the system. The output should be similar as below:



*[Estimate Finish Time: 30 minutes]*

void partB\_2() {

    int randomNumber, inputNumber;

    srand((unsigned)time(NULL)); *// set random seed*

    randomNumber = ( ((float)rand()) / ((float)RAND\_MAX) ) \* 100;

*// Print the title*

    string title = "Guess Number Game Started Now!";

    cout << title <<endl;

*// Print the title bar*

    for(int i=0; i<title.size();i++) cout << "-"; cout << endl;

*// Print the prompt and get input*

    cout << "Guess the number between 1 and 100"<<endl<<"Your Answer:";

    cin>>inputNumber;

*// Print result*

    cout <<

    ((inputNumber==randomNumber)?

       "Congratulations! You Won!" :

       "You lost..."

    )

    << "\nThe number was " << randomNumber

    << endl;

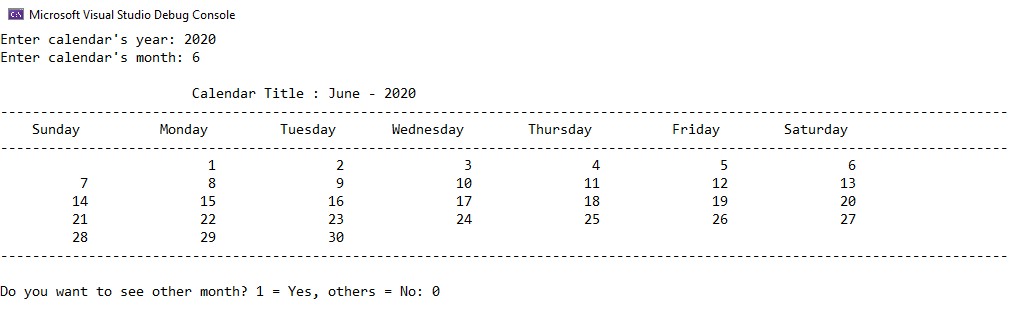
}

**Part C: Practice Yourself with More Questions.**

Submit your answer (*in doc / pdf*) to Moodle before 17 November 2023. Your answer should include your code and your program screenshot.

|  |
| --- |
|  |

1. Create a calendar application by using C++. The output should be similar as below:



*[*

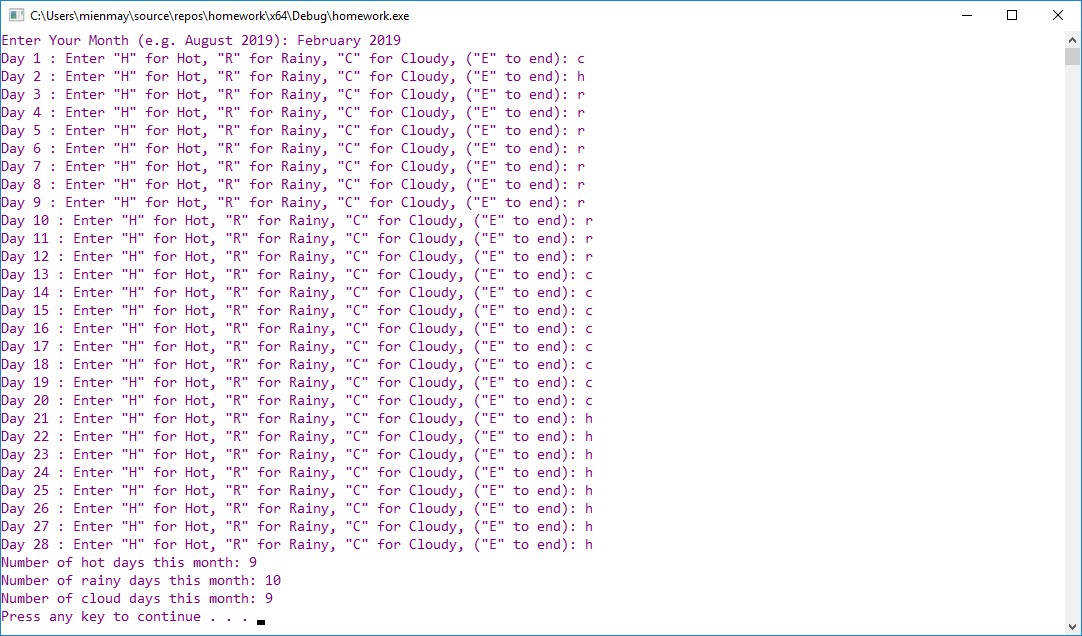
*Estimate Finish Time*

*:*

*45*

*minutes]*

1. Develop an interactive program that will keep track of the weather forecast in a month. On any given day, the weather forecast may be hot, rainy, or cloudy. Your program should input the weather forecast for each day in the month and should display the number of hot, rainy, and cloudy days in a month. You should use a loop and a conditional structure to develop this program. Array is not allowed to use in this question.



*[Estimate Finish Time: 45 minutes]*

Firstly, we will use the following utility functions:

*// Naive linear search*

*// (use a template for making it versatile)*

template<typename T>

bool isInVec(T elem, vector<T> vec) {

    bool in = false;

    for(auto i=vec.begin(); i!=vec.end();++i)

        if(elem==\*i) {in=true;break;}

    return in;

}

*// Modifies the string, removing*

*// the whitespace on the left*

void trimLeft(string str) {

    for(auto i=str.begin();i!=str.end();++i)

        if(\*i==' ') str.erase(i);

        else break;

}

*// Modifies the string, removing*

*// the whitespace on the right*

void trimRight(string str) {

    for(auto i=str.end();i!=str.begin();--i)

        if(\*i==' ') str.erase(i);

        else break;

}

Then, we proceed to implement the actual solution for the problem:

*// Weather forecast*

void partC\_2() {

    string monthYearInput;

    cout << "Enter your Month (e.g. August 2019): ";

    cin >> monthYearInput;

    if(0==monthYearInput.size()) {

        cout << "Invalid month..." << endl;

        return;

    }

    std::vector<string> validMonths = {

        "January", "February", "March", "April", "May",

        "June", "July", "August", "September", "October",

        "November", "December"

    };

*// Parse month only (only the first token*

*// before the whitespace will be taken)*

    int nMaxDays;

    trimLeft(monthYearInput);

    trimRight(monthYearInput);

    if(!isInVec<string>(monthYearInput, validMonths)) {

        cout << "Invalid month..." << endl;

        return;

    }

    if("February"==monthYearInput) {

        nMaxDays=28;

    }

    bool recordingDay = true;

    bool invalidInput = false;

    int dayCounter = 1;

    string promptString =

    "Day %d: Enter \"H\" for Hot, \"R\" for Rainy,\n\"C\" for Cloudy, (\"E\" to end): ";

    vector<char> validInputs = {'h', 'H', 'r', 'R', 'c', 'C', 'e', 'E'};

    vector<char> inputs;

    int nHot=0, nRainy=0, nCloudy=0;

    while(recordingDay && (dayCounter<=nMaxDays)) {

        if(invalidInput)

            cout << "\033[31mInvalid Input, try again!\033[m" << endl;

        invalidInput = false;

        string ninput;

        printf(promptString.c\_str(), dayCounter); *// use printf for formatting only*

        cin >> ninput;

*// input should have something...*

        if(0==ninput.size()) {invalidInput=true; continue;}

*// Only read the first character*

*// (Consider both lower and upper case)*

        if(!isInVec<char>(ninput[0], validInputs)) {

            invalidInput=true; continue;

        }

        if('e'==ninput[0]||'E'==ninput[0]) {

            recordingDay=false; break;

        } else {

            switch(ninput[0]) {

                case 'h': case 'H': nHot++; break;

                case 'r': case 'R': nRainy++; break;

                case 'c': case 'C': nCloudy++; break;

                default: break;

            }

        }

        dayCounter++;

    }

*// Final output*

    cout

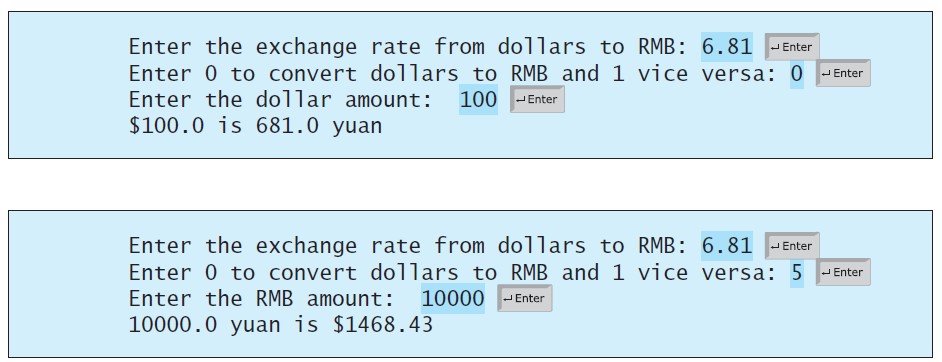
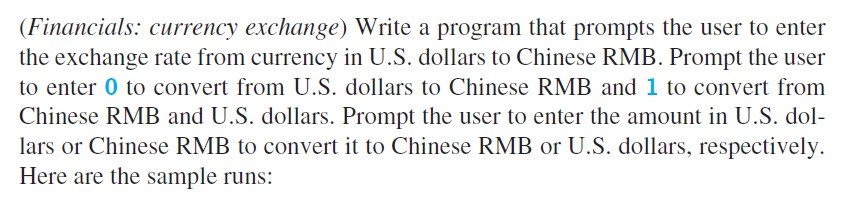
    << "Number of hot days this month: " << nHot << endl

    << "Number of rainy days this month: " << nRainy << endl

    << "Number of cloudy days this month: " << nCloudy << endl;

}

3.



*[*

*Estimate Finish Time:*

*15*

*minutes]*

*// Exchange rate program*

void partC\_3() {

*// Prompt user to input exchange rate from usd to rmb*

    cout << "Enter the exchange rate from dollars to RMB: ";

    double usdToRmbExchange;

    cin >> usdToRmbExchange;

*// Prompt 'direction' of conversion*

*// USD -> RMB = 0*

*// RMB -> USD = 1*

    cout << "Choose one of the following options:\n"

    << "0. Convert USD to RMB\n"

    << "1. Convert RMB to USD\n"

    << "Your choice: ";

    int conversionChoice;

    cin >> conversionChoice;

    if(0!=conversionChoice&&1!=conversionChoice) {

        cout << "Invalid choice, exiting..." << endl;

        return;

    }

    double originalAmount, convertedAmount;

*// If valid choice, then proceed to the amount input*

    cout <<

    "Enter the " << (conversionChoice? "RMB" : "USD")

    << " amount: ";

    cin >> originalAmount;

    convertedAmount = originalAmount

    \* (conversionChoice?

        (double)(1/usdToRmbExchange) : *// Invert the conversion rate*

        usdToRmbExchange); *// Keep the rate as-is*

    cout << originalAmount

    << (conversionChoice? " yuan" : " dollars")

    << " is "

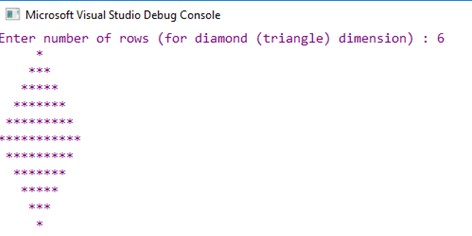
    << convertedAmount

    << (conversionChoice? " dollars" : " yuan")

    << endl;

}

1. Write a C++ program ask to the user to enter number of rows for diamond dimension to print the diamond pattern, then display the result on the screen. Note: Must use loops to print the diamond.



*[*

*Estimate Finish Time:*

*30*

*minutes]*

*// Diamond*

void partC\_4() {

*// Get the number of rows first*

    cout << "Enter the number of rows for the diamond (triangle) dimension: ";

    int nHalfRows;

    cin >> nHalfRows;

*// Validate input*

    if(nHalfRows<3) {

        cout << "Invalid number of rows..." << endl;

        return;

    }

    int maxRowLength = 1 + (2\*(nHalfRows));

    int lengthCounter = 1;

    for(int i=0;i<=nHalfRows\*2;i++) {

*// Print the left offset*

        for(int j=0; j<(maxRowLength-(lengthCounter/2)); j++) cout << " ";

*// Print the diamond row*

        for(int j=0; j<lengthCounter; j++) cout << "\*";

*// -> For the upper part of the diamond, increase the*

*// number of '\*' by 2*

*// -> Else, decrease it by 2*

        lengthCounter += (2 \* (i>=(nHalfRows)? -1 : 1 ));

*// After each row, go to next line*

        cout << endl;

    }

}

1. Write a program that can print the even numbers that in the between of 1 to 50.

**Output sample:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2 | 4 | 6 | 8 | 10 |
| 12 | 14 | 16 | 18 | 20 |
| 22 | 24 | 26 | 28 | 30 |
| 32 | 34 | 36 | 38 | 40 |
| 42 | 44 | 46 | 48 | 50 |

*[Estimate Finish Time: 10 minutes]*

*// Even numbers*

void partC\_5() {

    for(int i=1;i<=50;i++) {

        if (0==i%2) {

            cout << i; *// Print if even*

            if(i<10) cout << " "; *// Alignment thingy*

        }

        cout << " "; *// Print space in between*

        if (0==i%10) cout << endl;*// Go to next line, "wrapping"*

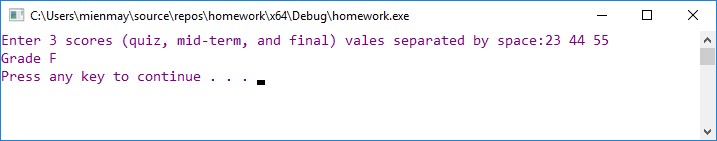
    }

}

1. Write a program that determines a student’s grade. The program will read three types of scores in percentage (quiz, mid-term, and final scores) and determine the grade based on the following rules:

*if the average score =90% =>grade=A if the average score >= 70% and <90% => grade=B if the average score>=50% and <70% =>grade=C if the average score<50% =>grade=F*

**Output sample:**



*[Estimate Finish Time: 15 minutes]*

*// Student grade*

void partC\_6() {

*// Get input*

    cout << "Enter 3 scores (quiz, mid-term, and final) separated by spaces: ";

    double quizScore, midtermScore, finalScore;

    cin >> quizScore >> midtermScore >> finalScore;

*// Validate input*

    if(

        quizScore>100       ||quizScore<0

        ||midtermScore>100  ||midtermScore<0

        ||finalScore>100    ||finalScore<0

    ) {

        cout << "Invalid input... make sure that values are between 0-100"

        << endl;

        return;

    }

*// Calculate final numerical grade*

    double average = (quizScore+midtermScore+finalScore)/3;

*// Ternary operator sorcery*

    char grade =

    (average>=90)? 'A'

        :((average>=70)? 'B'

            :((average>=50)? 'C'

                :'F')

        );

    cout << "Grade: " << grade << endl;

}

1. The area of a rectangle is the rectangle’s length times its width. Write a program that asks for the length and width of two rectangles. The program should tell the user which rectangle has the greater area, or if the areas are the same.

**Output sample:**

Rectangle A:

Width = 2

Height = 4

Rectangle B:

Width = 3

Height = 10

Area in rectangle B is bigger rectangle A.

*[Estimate Finish Time: 15 minutes]*

*// Rectangle areas*

void partC\_7() {

    int wA, hA, wB, hB;

*// Get Rectangle A*

    cout << "Rectangle A:" << endl;

    cout << "Width = ";

    cin >> wA;

    cout << "Height = ";

    cin >> hA;

    if(wA<=0||hA<=0) {

        cout << "Dimensions should be greater than 0..." << endl;

        return;

    }

*// Get Rectangle B*

    cout << "Rectangle B:" << endl;

    cout << "Width = ";

    cin >> wB;

    cout << "Height = ";

    cin >> hB;

    if(wB<=0||hB<=0) {

        cout << "Dimensions should be greater than 0..." << endl;

        return;

    }

*// Calculate areas*

    int

    areaA = wA\*hA,

    areaB = wB\*hB;

*// Output*

    if(areaA!=areaB){

        cout << "Area in Rectangle "

        << ((areaA>areaB)? "A" : "B")

        << " is bigger than area of rectangle "

        << ((areaA>areaB)? "B" : "A")

        << endl;

    } else

        cout << "Areas are equal!" << endl;

}

1. Compute the total cost of a meal inclusive of GST (Goods and Service Tax) and tipping. Assume that the GST is fixed at 6% but the amount to tip will depend on the customer. Your program should input the cost of the meal (before GST and tipping) and the tip amount (to be input as a percentage).

Your program should display:

*The total cost of the meal BEFORE GST and tipping*

*The total cost of the meal AFTER GST*

*The total cost of the meal AFTER GST and tipping*

*// Goods and Service Tax*

void partC\_8() {

    double cost, tip;

*// Get the original meal cost*

    cout << "Original cost of the meal >> ";

    cin >> cost;

*// Convert to positive, maybe the previous*

*// input was given as a negative cost*

    if(cost<0) cost \*= -1;

*// Get the tax as percentage*

    cout << "Tip (percentage, 0-100) >> ";

    cin >> tip;

*// Ensure its a percentage*

    if(0>tip||100<tip) {

        cout << "Make sure its a 0-100 percentage..." << endl;

        return;

    }

    const double GST = 0.06;

*// Convert tip from 0-100 to 0-1 rate*

*// (For convenience)*

    tip /= 100;

    cout << "--------------" << endl

    << "INVOICE" << endl

    << "--------------" << endl

    << "Original cost: " << cost << endl

    << "Cost with \%6 Goods and \nServices Tax (GST): "

    << cost + (GST\*cost) << endl

    << "--------------" << endl

    << "Cost plus GST and Tip: "

    << cost + (GST\*cost) + (tip\*cost) << endl;

}