1. **Problem Statement**

Program prints the total value of sales during a specific month by each department and in each store using 3-dimensional array. It also prints the total sales for both departments in each store.

1. **Requirements**
   1. **Assumptions**
      1. User will enter “y” or “n” to tell if they want to enter a new month to print.
      2. User will enter only integers for month selection
   2. **Specifications**
      1. Program will initialize 3-dimensional array.
      2. #define NUM\_DEPTS 2
      3. #define NUM\_STORES 2
      4. #define NUM\_MONTHS 12
      5. When user types the month, pass it in to *printMonthlySales* to print data
         1. Edit the correct value for month in main
      6. Ask user if they want to enter a new month to print by typing y for yes and n for no
         1. Loop the program until user says n for no.
2. **Decomposition Diagram**

|  |  |  |
| --- | --- | --- |
| **Main** | | |
| **Input** | **Process** | **Output** |
| “y” if user wants to enter a new month to print | Go back in the loop and call the function for that month | Print out the data for that month |
| “n” if user wants to quit the program |  | Print out Thank You message |
|  | Organize the data in the array in a readable format |  |
|  |  | Print out Welcome message |

1. **Test Strategy**
   1. Valid Data
   2. Invalid Data
2. **Test Plan Version 1**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Strategy | Test Number | Description | Input | Expected Output | Actual Output | Pass/Fail |
| Invalid | 1 | User enters number of month < 1 |  |  |  |  |
| Invalid | 2 | User enters number of month > 12 |  |  |  |  |
| Valid | 1 | User enters number of month between 1 and 12 inclusive |  |  |  |  |

1. **Initial Algorithm**
   1. Define global variables
      1. One for # of departments, assigning it to 2
      2. One for # of stores, assigning it to 2
      3. One for # of months, assigning it to 12
   2. In main function
      1. Initialize *storeMonthlySales* array to the given data in 3-dimensional format.
      2. Ask the user for the number of month to print its data
         1. Check that the month is between 1 and 12 inclusive
      3. Pass the array and month to the function *printMonthlySales* to call it
      4. Ask the user if they want to enter a new month to print using “y” for yes or “n” for no.
         1. Loop the program until the user types “n” for no.
   3. In *printMonthlySales* function
      1. Using a switch-case loop for the selected month, print out the values for that particular month from the array
      2. When printing out, format the output in readable format.
2. **Test Plan Version 2**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Strategy | Test Number | Description | Input | Expected Output | Actual Output | Pass/Fail |
| Invalid | 1 | User enters number of month < 1 | -5 | Error. Invalid Month. Try Again |  |  |
| Invalid | 2 | User enters number of month > 12 | 20 | Error. Invalid Month. Try Again |  |  |
| Valid | 1 | User enters number of month between 1 and 12 inclusive | 5 | Prints out the data for May month |  |  |

1. **Code**

/\*

Author: Nisarg Patel

CIS200 WINTER 2018

Creation Date: 1-31-2017

Modification Date: 2-01-2018

Purpose: Program prints the total value of sales during a specific month

by each department and in each store using 3-dimensional array.

It also prints the total sales for both departments in each store.

\*/

cout << "Greetings, user. Welcome to an amazing program that displays data of your choice of month" << endl << endl;

#include <iostream>

#include <string>

using namespace std;

#define NUM\_DEPTS 2 //Constant global variable for number of depertments

#define NUM\_STORES 2 //Constant global variable for number of stores

#define NUM\_MONTHS 12 //Constant global variable for number of months

void printMonthlySales(float storeMontlySales[NUM\_STORES][NUM\_MONTHS][NUM\_DEPTS], int numMonth); //Function to print data for a specific month

int main()

{

//Assign data to this 3-dimensional arrays

float storeMontlySales[NUM\_STORES][NUM\_MONTHS][NUM\_DEPTS] =

{

{{1.1, 1.2}, {1.3, 1.4}, {1.5, 1.6}, {1.7, 1.8}, {1.9, 2.0}, {2.1, 2.2},

{2.1, 2.2}, {2.3, 2.4}, {2.5, 2.6}, {2.7, 2.8}, {2.9, 3.0}, {3.1, 3.2}},

{{3.1, 3.2}, {3.3, 3.4}, {3.5, 3.6}, {3.7, 3.8}, {3.9, 4.0}, {4.1, 4.2},

{2.1, 2.2}, {2.3, 2.4}, {2.5, 2.6}, {2.7, 2.8}, {2.9, 3.0}, {3.1, 3.2}}

};

string newMonth; //User's response if they want to try different month: y for yes and n for no.

int numMonth; //User's choice of month to be be passed in as parameter

do { //Takes user's month number and passes the array and month number in functuon

do { //Loops until user enters month between 1 and 12

cout << "Give me a number of month (Between 1 and 12): " << endl;

cin >> numMonth;

} while (numMonth > 12 || numMonth < 1);

printMonthlySales(storeMontlySales, numMonth - 1); //Calling function to print data in organized format

cout << "Do you want to enter a new month? Type y for yes and n for no: " << endl;

cin >> newMonth;

} while (newMonth == "y"); //Runs until user enter n for no.

system("pause");

return 0;

}

void printMonthlySales(float storeMontlySales[NUM\_STORES][NUM\_MONTHS][NUM\_DEPTS], int numMonth) //Function to print data for a specific month

{

// Organizes and prints the data according to given month and array

cout << "\t\tDept 1\t\tDept 2\t\tStore Total" << endl;

cout << "Store 1\t\t" << storeMontlySales[0][numMonth][0] << "\t\t" << storeMontlySales[0][numMonth][1] << "\t\t" << storeMontlySales[0][numMonth][0] + storeMontlySales[0][numMonth][1] << endl;

cout << "Store 1\t\t" << storeMontlySales[1][numMonth][0] << "\t\t" << storeMontlySales[1][numMonth][1] << "\t\t" << storeMontlySales[1][numMonth][0] + storeMontlySales[1][numMonth][1] << endl;

cout << "Dept Total\t" << storeMontlySales[0][numMonth][0] + storeMontlySales[1][numMonth][0] << "\t\t" << storeMontlySales[0][numMonth][1] + storeMontlySales[1][numMonth][1] << "\t\t" << storeMontlySales[0][numMonth][0] + storeMontlySales[0][numMonth][1] + storeMontlySales[1][numMonth][0] + storeMontlySales[1][numMonth][1] << endl;

}

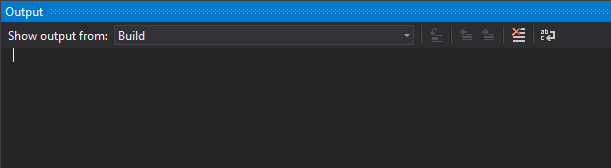
cout << "Thank you for using this program." << endl;

1. **Updated Algorithm**
   1. Greetings
   2. Define global variables
      1. One for # of departments, assigning it to 2
      2. One for # of stores, assigning it to 2
      3. One for # of months, assigning it to 12
   3. In main function
      1. Initialize *storeMonthlySales* array to the given data in 3-dimensional format.
      2. Ask the user for the number of month to print its data
         1. Check that the month is between 1 and 12 inclusive
         2. Loop until corrent month value is given
      3. Pass the array and month-1 to the function *printMonthlySales* to call it
      4. Ask the user if they want to enter a new month to print using “y” for yes or “n” for no.
         1. Loop the program until the user types “n” for no.
      5. Thank you message
   4. In *printMonthlySales* function
      1. ~~Using a switch-case loop for the selected month, print out the values for that particular month from the array~~
      2. Print out each data separately for the given month for every store and department
      3. When printing out, format the output in readable format.
2. **Test Plan Version 3**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Strategy | Test Number | Description | Input | Expected Output | Actual Output | Pass/Fail |
| Invalid | 1 | User enters number of month < 1 | -5, 0 | Error. Invalid Month. Try Again | Error. Invalid Month. Try Again. Keeps looping | Pass |
| Invalid | 2 | User enters number of month > 12 | 20, 40 | Error. Invalid Month. Try Again | Error. Invalid Month. Try Again. Keeps looping | Pass |
| Valid | 1 | User enters number of month between 1 and 12 inclusive | 5 | Prints out the data for May month | Prints out the data for May month | Pass |

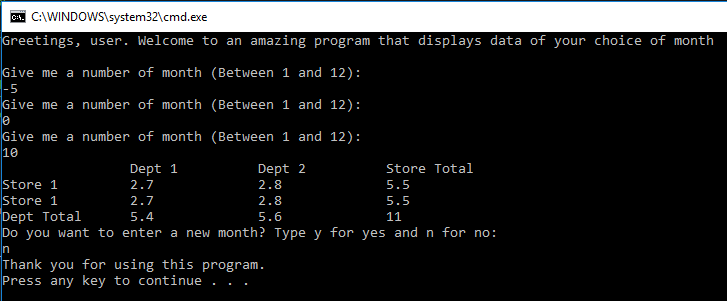
1. **Screenshots**

Build (Windows):

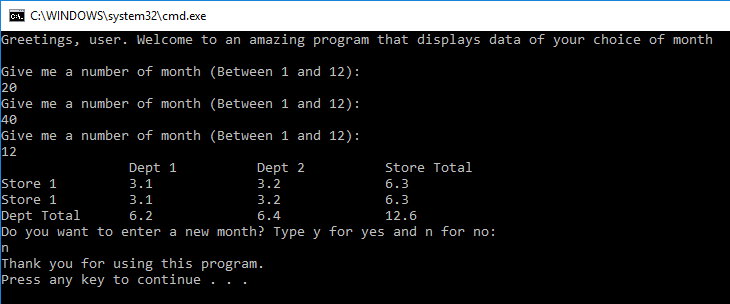


I was confused why there was nothing showing up in this build windows even though I was compiling and running it just like the other program. This is what I got for this program.

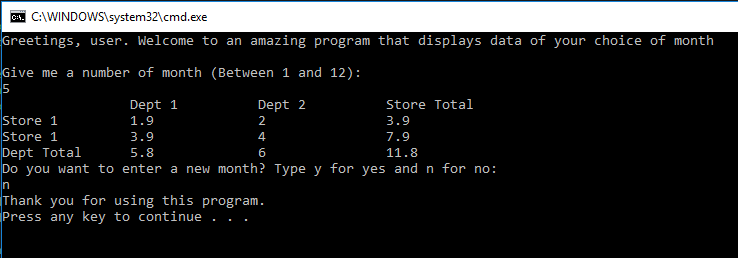
Invalid Test Case 1 (Windows):



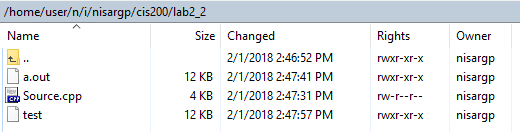
Invalid Test Case 2 (Windows):



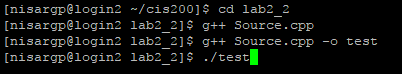
Valid Test Case 1 (Windows):



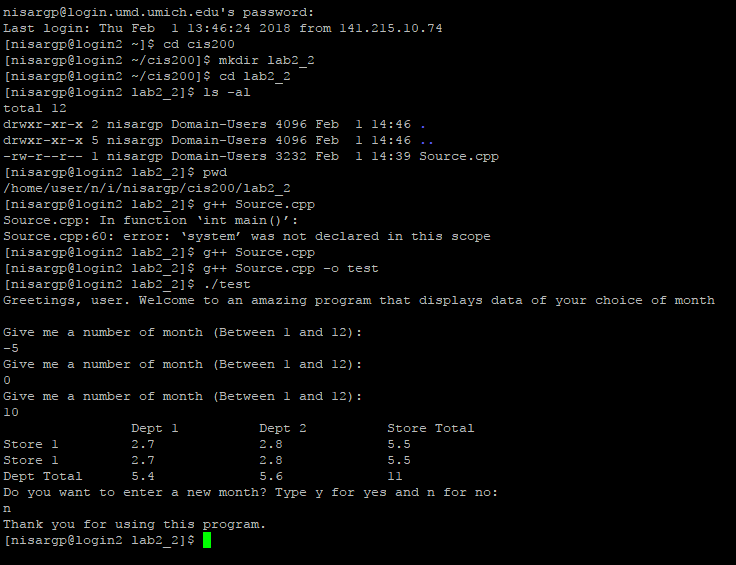
Directory (Linux):



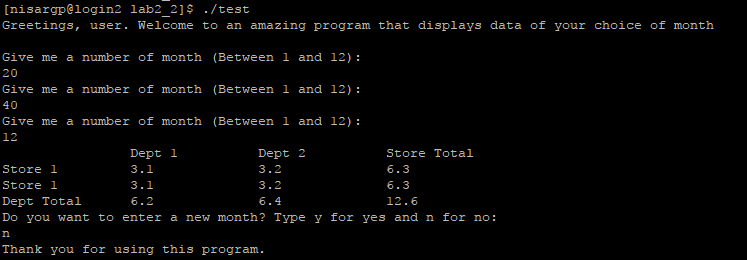
Compile using *g++* (Linux):



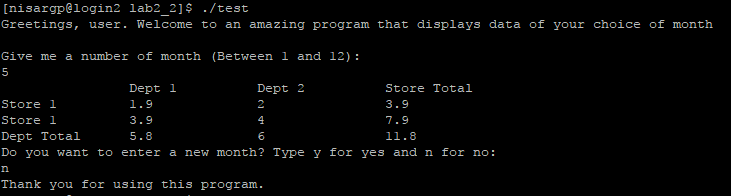
Invalid Test Case 1 (Linux):



Invalid Test Case 2 (Linux):



Valid Test Case 1 (Linux):



1. **Status**

It works with assumptions in mind.