CMIS 102 Hands-On Lab

Week 8

Overview

This hands-on lab allows you to follow and experiment with the critical steps of developing a program including the program description, analysis, test plan, and implementation with C code. The example provided uses sequential, repetition, selection statements, functions, strings and arrays.

Program Description

This program will input and store meteorological data into an array. The program will prompt the user to enter the average monthly rainfall for a specific region and then use a loop to cycle through the array and print out each value. The program should store up 5 years of meteorological data. Data is collected once per month. The program should provide the option to the user of not entering any data.

Analysis

I will use sequential, selection, and repetition programming statements and an array to store data.
I will define a 2-D array of Float number: Raindata[][] to store the Float values input by the user. To store up to 5 years of monthly data, the array size should be at least 5*12 = 60 elements. In a 2D array this will be RainData[5][12]. We can use #defines to set the number of years and months to eliminate hard-coding values.
A float number (rain) will also be needed to input the individual rain data.
A nested for loop can be used to iterate through the array to enter Raindata. A nested for loop can also be used to print the data in the array.

A array of strings can be used to store year and month names. This will allow a tabular display with labels for the printout.

Functions will be used to separate functionality into smaller work units. Functions for displaying the data and inputting the data will be used.

A selection statement will be used to determine if data should be entered.

Test Plan

To verify this program is working properly the input values could be used for testing:

Test Case	Input	Expected Output
1	Enter data? = y 1.2 2.2 3.3 2.2 10.2 12.2 2.3 0.4 0.2 1.1 2.1	year month rain 2011 Jan 1.20 2011 Feb 2.20 2011 Mar 3.30 2011 Apr 2.20 2011 May 10.20 2011 Jun 12.20 2011 Jul 2.30 2011 Aug 0.40 2011 Sep 0.20 2011 Oct 1.10 2011 Nov 2.10 2011 Dec 0.40

	0.4 1.1 2.2 3.3 2.2 10.2 12.2 2.3 0.4 0.2 1.1 2.1 0.4 1.1 2.2 3.3 2.2 10.2 12.2 2.3 0.4 0.2 1.1 2.1 0.4 1.1 2.2 3.3 2.2 10.2 12.2 2.3 0.4 0.2 1.1 2.1 0.4	2012 Jan 1.10 2012 Feb 2.20 2012 Mar 3.30 2012 Apr 2.20 2012 May 10.20 2012 Jun 12.20 2012 Jul 2.30 2012 Aug 0.40 2012 Sep 0.20 2012 Oct 1.10 2012 Nov 2.10 2012 Dec 0.40 2013 Jan 1.10 2013 Feb 2.20 2013 Mar 3.30 2013 Apr 2.20 2013 May 10.20 2013 Jun 12.20 2013 Jul 2.30 2013 Aug 0.40 2013 Sep 0.20 2013 Oct 1.10 2013 Nov 2.10 2013 Dec 0.40 2014 Jan 1.10 2014 Feb 2.20 2014 Mar 3.30 2014 Apr 2.20 2014 May 10.20 2014 Jun 12.20 2014 Jul 2.30 2014 Aug 0.40 2014 Sep 0.20 2014 Oct 1.10 2014 Nov 2.10 2014 Dec 0.40 2015 Jan 1.10 2015 Feb 2.20 2015 Mar 3.30 2015 Apr 2.20 2015 May 10.20 2015 Jun 12.20 2015 Jul 2.30 2015 Aug 0.40 2015 Sep 0.20 2015 Oct 1.10 2015 Nov 2.10 2015 Dec 0.40 Please try the Precipitation program again.
2	Enter data? = n	No data was input at this time.

		Please try the Precipitation program again.

C Code

The following is the C Code that will compile in execute in the online compilers.

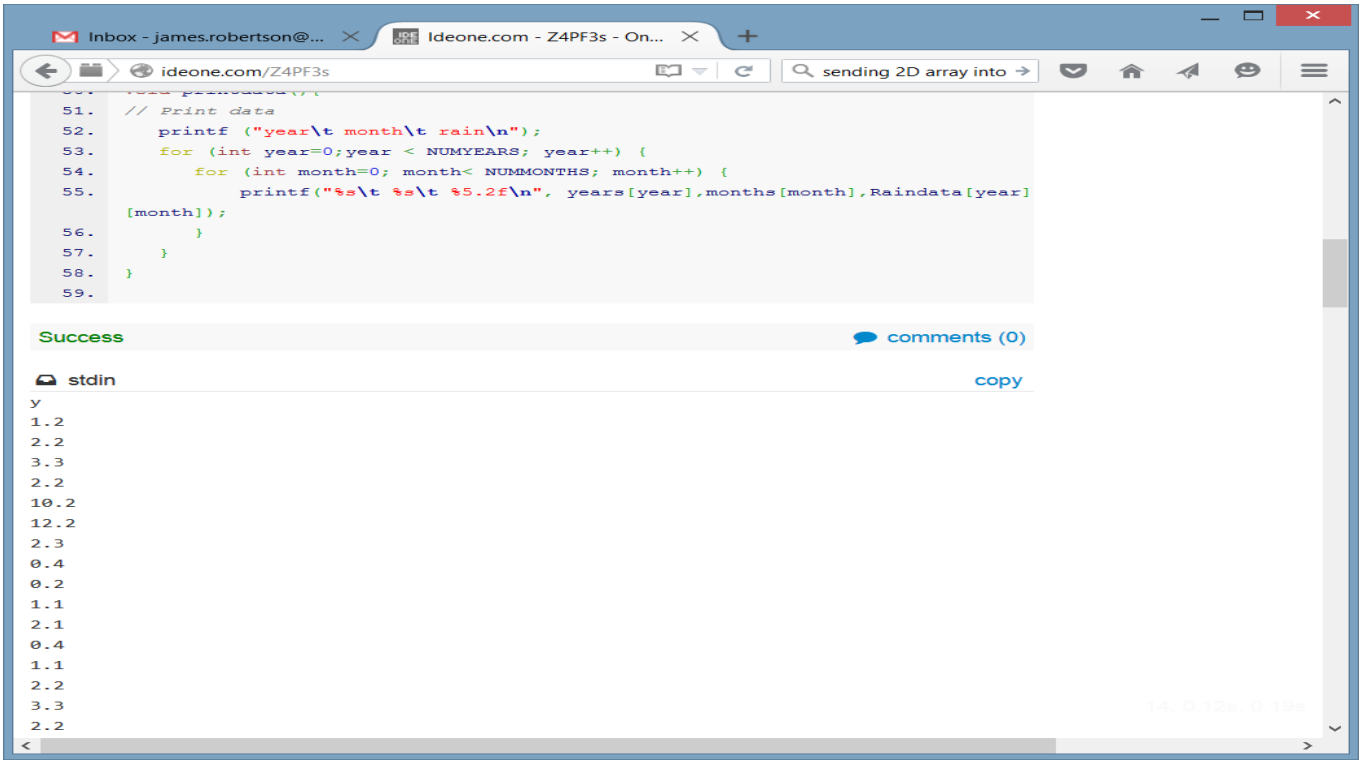
```
// C code
// This program will input and store meteorological data into an array.
// Developer: Faculty CMIS102
// Date: Jan 31, XXXX
#define NUMMONTHS 12
#define NUMYEARS 5
#include <stdio.h>

// function prototypes
void inputdata();
void printdata();

// Global variables
// These are available to all functions
float Raindata[NUMYEARS][NUMMONTHS];
char years[NUMYEARS][5] = {"2011","2012","2013","2014","2015"};
 char months[NUMMONTHS][12]
={"Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"};
int main ()
{
  char enterData = 'y';
  printf("Do you want to input Precipatation data? (y for yes)\n");
  scanf("%c",&enterData);
  if (enterData == 'y') {
    // Call Function to Input data
    inputdata();

     // Call Function to display data
     printdata();
   }
   else {
     printf("No data was input at this time\n");
   }
   printf("Please try the Precipitation program again. \n");
   return 0;
}
// function to inputdata
void inputdata() {
  /* variable definition: */
  float Rain=1.0;
   // Input Data
   for (int year=0;year < NUMYEARS; year++) {
      for (int month=0; month< NUMMONTHS; month++) {
          printf("Enter rain for %d, %d:\n", year+1, month+1);
          scanf("%f",&Rain);
          Raindata[year][month]=Rain;
```

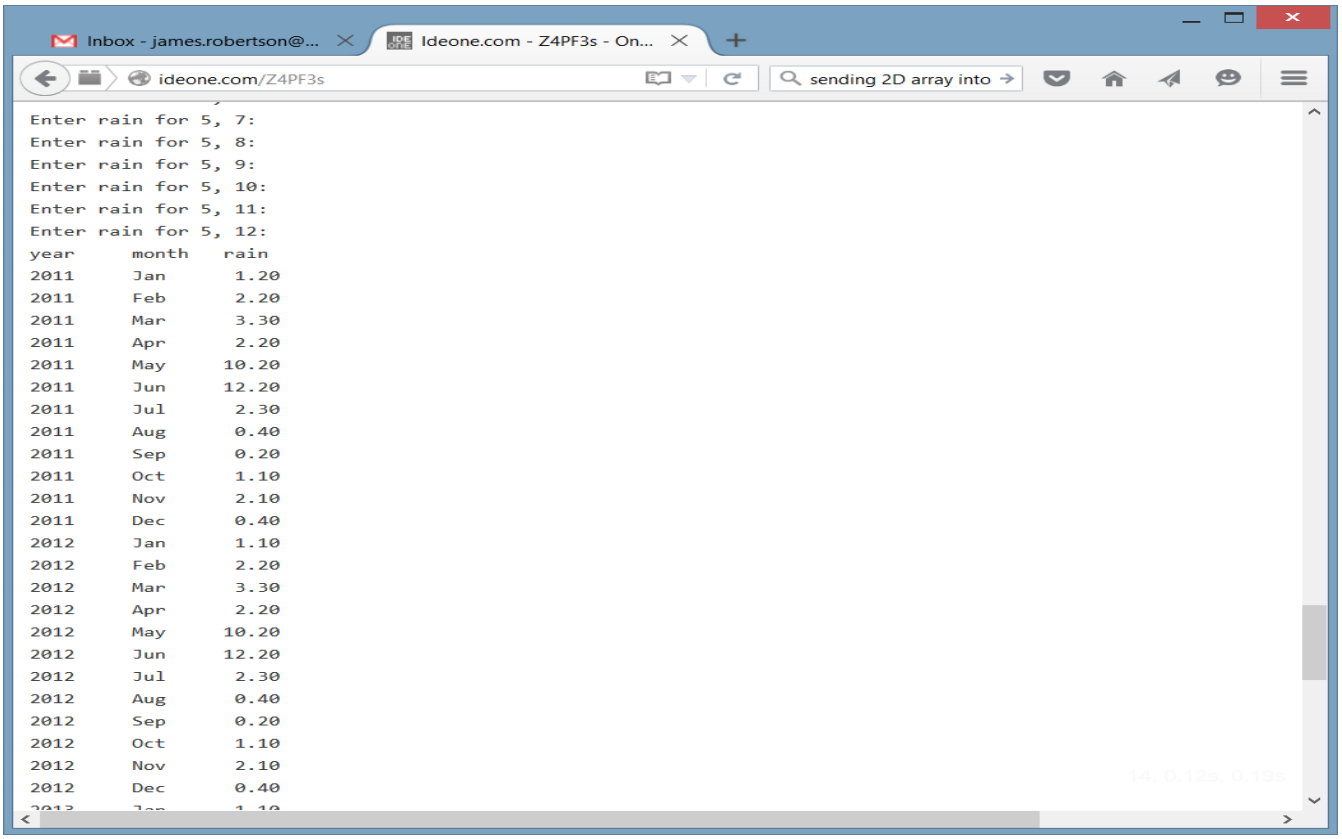
```
      }
   }
}
// Function to printdata
void printdata(){
// Print data
   printf ("year\t month\t rain\n");
   for (int year=0;year < NUMYEARS; year++) {
      for (int month=0; month< NUMMONTHS; month++) {
          printf("%s\t %s\t %5.2f\n",
years[year],months[month],Raindata[year][month]);
      }
   }
}
```

Setting up the code and the input parameters in ideone.com:

You can change these values to any valid integer values to match your test cases.

Results from running the programming at ideone.com





Learning Exercises for you to complete

1. Modify the program to add a function to sum the rainfall for each year. (Hint: you need to sum for each year. You can do this using a looping structure). Support your experimentation with screen captures of executing the new code.

2. Enhance the program to allow the user to enter another meteorological element such as windspeed (e.g. 2.4 mph). Note, the user should be able to enter both rainfall and windspeed in your new implementation. Support your experimentation with screen captures of executing the new code.

3. Prepare a new test table with at least 2 distinct test cases listing input and expected output for the code you created after step 2.

4. What happens if you change the NUMMONTHS and NUMYEARS definitions to other values? Be sure to use both lower and higher values. Describe and implement fixes for any issues if errors results. Support your experimentation with screen captures of executing the new code.

Grading guidelines

Submission	**Points**
Successfully demonstrates execution of this lab with online compiler. Includes a screen capture.	2
Modifies the code to add a function to sum the rainfall for each year. Support your experimentation with screen captures of executing the new code	2
Enhances the program to allow the user to enter another meteorological element such as windspeed (e.g. 2.4 mph). Support your experimentation with screen captures of executing the new code.	2
Provides a new test table with at least 2 distinct test cases listing input and expected output for the code you created after step 2.	1
Describes what would happen if you change the NUMMONTHS and NUMYEARS definitions to other values? Applies both lower and higher values. Describes and implements fixes for any issues if errors results. Support your experimentation with screen captures of executing the new code.	2
Document is well-organized, and contains minimal spelling and grammatical errors.	1
Total	**10**