6 Arrays

Instructions: For each of the Lab problems work through the exercises. When asked for a screenshot be sure to take one, if a screenshot is not specifically stated then you should be recording the results by typing them out or answering the questions being asked. DO NOT include a screenshot for these two later items. I have included "Notes" to the left side that you should be sure to review.

LAB 6.1 Working with One-Dimensional Arrays

Retrieve program `testscore.cpp` from Blackboard. The code is as follows:

```
// This program will read in a group of test scores (positive integers from 1 to 100)
// from the keyboard and then calculate and output the average score
// as well as the highest and lowest score. There will be a maximum of 100 scores.

// PLACE YOUR NAME HERE

#include <iostream>
using namespace std;

typedef int GradeType[100];               // declares a new data type:
                                          // an integer array of 100 elements


float findAverage (const GradeType, int);  // finds average of all grades
int   findHighest (const GradeType, int);  // finds highest of all grades
int   findLowest  (const GradeType, int);  // finds lowest of all grades

int main()

{
      GradeType grades;                     // the array holding the grades.
      int numberOfGrades;                   // the number of grades read.
      int pos;                              // index to the array.

      float avgOfGrades;                    // contains the average of the grades.
      int highestGrade;                     // contains the highest grade.
      int lowestGrade;                      // contains the lowest grade.

      // Read in the values into the array

      pos = 0;
      cout << "Please input a grade from 1 to 100, (or -99 to stop)" << endl;
```

continues

```
      cin  >> grades[pos];

      while (grades[pos] != -99)
      {

        // Fill in the code to read the grades

      }

      numberOfGrades = _________;  // Fill blank with appropriate identifier

      // call to the function to find average

      avgOfGrades = findAverage(grades, numberOfGrades);

      cout << endl << "The average of all the grades is " << avgOfGrades << endl;


      // Fill in the call to the function that calculates highest grade


      cout << endl << "The highest grade is " << highestGrade << endl;

      // Fill in the call to the function that calculates lowest grade
      // Fill in code to write the lowest to the screen

      return 0;
}

//******************************************************************************
//                                 findAverage
//
// task:          This function receives an array of integers and its size.
//                It finds and returns the average of the numbers in the array
// data in:       array of floating point numbers
// data returned: average of the numbers in the array
//
//******************************************************************************

float findAverage (const GradeType  array, int size)

{

    float sum = 0;   // holds the sum of all the numbers

    for (int pos = 0; pos < size; pos++)

      sum = sum + array[pos];

    return (sum / size);  //returns the average

}
```

```
//****************************************************************************
//                                 findHighest
//
// task:          This function receives an array of integers and its size.
//                It finds and returns the highest value of the numbers in the array
// data in:       array of floating point numbers
// data returned: highest value of the numbers in the array
//
//****************************************************************************

int   findHighest (const GradeType array, int size)

{

      / Fill in the code for this function

}


//****************************************************************************
//                                 findLowest
//
// task:          This function receives an array of integers and its size.
//                It finds and returns the lowest value of the numbers in the array
// data in:       array of floating point numbers
// data returned: lowest value of the numbers in the array
//
//****************************************************************************

int   findLowest  (const GradeType array, int size)

{
      // Fill in the code for this function

}
```

Exercise 1: Complete this program as directed.

Exercise 2: Run the program with the following data: 90 45 73 62 -99 and **record the output here:**

____

____

____

Exercise 3: Modify your program from Exercise 1 so that it reads the information from the `gradfile.txt` file, reading until the end of file is encountered. Run the program and **take a screenshot.**

Lab 6.2 Strings as Arrays of Characters

Retrieve program `student.cpp` Blackboard.

```
// This program will input an undetermined number of student names
// and a number of grades for each student. The number of grades is
// given by the user. The grades are stored in an array.
// Two functions are called for each student.
// One function will give the numeric average of their grades.
// The other function will give a letter grade to that average.
// Grades are assigned on a 10 point spread.
// 90-100 A   80-89 B  70-79 C   60-69 D   Below 60 F

// PLACE YOUR NAME HERE


#include <iostream>
#include <iomanip>
using namespace std;

const  int MAXGRADE = 25;              // maximum number of grades per student
const  int MAXCHAR = 30;               // maximum characters used in a name

typedef char StringType30[MAXCHAR + 1];// character array data type for names
                                       // having 30 characters or less.
typedef  float GradeType[MAXGRADE];    // one dimensional integer array data type

float findGradeAvg(GradeType, int);    // finds grade average by taking array of
                                       // grades and number of grades as parameters

char  findLetterGrade(float);          // finds letter grade from average given
                                       // to it as a parameter

int main()

{
      StringType30 firstname, lastname;// two arrays of characters defined
      int numOfGrades;                 // holds the number of grades
      GradeType  grades;               // grades defined as a one dimensional array
      float average;                   // holds the average of a student's grade
      char moreInput;                  // determines if there is more input

      cout << setprecision(2) << fixed << showpoint;

    // Input the number of grades for each student

      cout << "Please input the number of grades each student will receive." << endl
           << "This must be a number between 1 and " << MAXGRADE << " inclusive”
           << endl;

      cin >> numOfGrades;
      while (numOfGrades > MAXGRADE || numOfGrades < 1)
      {
            cout << "Please input the number of grades for each student." << endl
                 << "This must be a number between 1 and " << MAXGRADE
                 << " inclusive\n";

            cin >> numOfGrades;

      }

    // Input names and grades for each student

      cout << "Please input a y if you want to input more students"
           << " any other character will stop the input" << endl;
      cin >> moreInput;

      while (moreInput == 'y' || moreInput == 'Y')

      {
           cout << "Please input the first name of the student" << endl;
           cin >> firstname;
           cout << endl << "Please input the last name of the student" << endl;
           cin >> lastname;

           for (int count = 0; count < numOfGrades; count++)

           {

                 cout << endl << "Please input a grade" << endl;

                  // Fill in the input statement to place grade in the array

           }

            cout << firstname << " " << lastname << " has an average of ";

             // Fill in code to get and print average of student to screen
             // Fill in call to get and print letter grade of student to screen

            cout << endl << endl << endl;
            cout << "Please input a y if you want to input more students"
                 << " any other character will stop the input" << endl;
            cin >> moreInput;

      }

      return 0;
}
```

continues

```
//*********************************************************************
//                              findGradeAvg
//
//  task:           This function finds the average of the
//                  numbers stored in an array.
//
//  data in:        an array of integer numbers
//  data returned:  the average of all numbers in the array
//
//*********************************************************************

float findGradeAvg(GradeType array, int numGrades)

{
    // Fill in the code for this function
}

//*********************************************************************
//                              findLetterGrade
//
//  task:           This function finds the letter grade for the number
//                  passed to it by the calling function
//
//  data in:        a floating point number
//  data returned:  the grade (based on a 10 point spread) based on the number
//                  passed to the function
//
//*********************************************************************

char  findLetterGrade(float numGrade)

{
  // Fill in the code for this function

}
```

Exercise 1: Complete the program by filling in the code. (Areas in bold)

Run the program with 3 grades per student using the sample data below.

Mary Brown	100	90	90
George Smith	90	30	50
Dale Barnes	80	78	82
Sally Dolittle	70	65	80
Conrad Bailer	60	58	71

You should get the following results. **Take a screenshot.**

Mary Brown has an average of 93.33 which gives the letter grade of A
George Smith has an average of 56.67 which gives the letter grade of F
Dale Barnes has an average of 80.00 which gives the letter grade of B
Sally Dolittle has an average of 71.67 which gives the letter grade of C
Conrad Bailer has an average of 63.00 which gives the letter grade of D

LAB 6.3 Working with Two-Dimensional Arrays

Look at the following table containing prices of certain items:

12.78	23.78	45.67	12.67
7.83	4.89	5.99	56.84
13.67	34.84	16.71	50.89

These numbers can be read into a two-dimensional array.

Retrieve `price.cpp` from Blackboard. The code is as follows:

```
// This program will read in prices and store them into a two-dimensional array.
// It will print those prices in a table form.

// PLACE YOUR NAME HERE

#include <iostream>
#include <iomanip>
using namespace std;


const  MAXROWS = 10;
const  MAXCOLS = 10;

typedef float PriceType[MAXROWS][MAXCOLS];      // creates a new data type
                                                // of a 2D array of floats

void   getPrices(PriceType, int&, int&);        // gets the prices into the array
void   printPrices(PriceType, int, int);        // prints data as a table


int main()

{
    int rowsUsed;                               // holds the number of rows used
    int colsUsed;                               // holds the number of columns used
    PriceType priceTable;                       // a 2D array holding the prices

    getPrices(priceTable, rowsUsed, colsUsed);  // calls getPrices to fill the array
    printPrices(priceTable, rowsUsed, colsUsed);// calls printPrices to display array

    return 0;
}
```

continues

```
//*************************************************************************
//                           getPrices
//
//  task:     This procedure asks the user to input the number of rows and
//            columns. It then asks the user to input (rows * columns) number of
//            prices.  The data is placed in the array.
//  data in:  none
//  data out: an array filled with numbers and the number of rows
//            and columns used.
//
//*************************************************************************

void   getPrices(PriceType table, int& numOfRows, int& numOfCols)
{

      cout << "Please input the number of rows from 1 to "<< MAXROWS << endl;
      cin >> numOfRows;

      cout << "Please input the number of columns from 1 to "<< MAXCOLS << endl;
      cin >> numOfCols;


      for (int row = 0;  row < numOfRows;  row++)
      {
         for (int col = 0; col < numOfCols;  col++)

             // Fill in the code to read and store the next value in the array
      }
}

//*********************************************************************
//                           printPrices
//
//  task:     This procedure prints the table of prices
//  data in:  an array of floating point numbers and the number of rows
//            and columns used.
//  data out: none
//
//*********************************************************************

void   printPrices(PriceType table, int numOfRows, int numOfCols)
{

      cout << fixed << showpoint << setprecision(2);

      for (int row = 0;  row < numOfRows; row++)
      {
            for (int col = 0;  col < numOfCols; col++)

                   // Fill in the code to print the table
      }
}
```

Exercise 1: Fill in the code to complete both functions `getPrices` and `printPrices`, then run the program with the following data. **Take a screenshot.**

```
Please input the number of rows from 1 to 10
2

Please input the number of columns from 1 to 10
3

Please input the price of an item with 2 decimal places
1.45

Please input the price of an item with 2 decimal places
2.56

Please input the price of an item with 2 decimal places
12.98

Please input the price of an item with 2 decimal places
37.86

Please input the price of an item with 2 decimal places
102.34

Please input the price of an item with 2 decimal places
67.89

        1.45    2.56        12.98
        37.86   102.34      67.89
```

Exercise 2: **Why does `getPrices` have the parameters `numOfRows` and `numOfCols` passed by reference whereas `printPrices` has those parameters passed by value?**

Exercise 3: The following code is a function that returns the highest price in the array. After studying it very carefully, place the function in the above program and have the program print out the highest value.

```
float  findHighestPrice(PriceType table, int numOfRows, int numOfCols)
// This function returns the highest price in the array
{

      float highestPrice;

      highestPrice = table[0][0];  // make first element the highest price

      for (int row = 0;  row < numOfRows;  row++)
            for (int col = 0;  col < numOfCols; col++)
                  if ( highestPrice < table[row][col] )

                        highestPrice = table[row][col];

      return highestPrice;
}
```

continues

Video may state Exercise 6 is optional. It is NOT. You should complete this exercise and submit the required screenshot.

NOTE: This is a value returning function. Be sure to include its prototype in the global section.

Exercise 4: Create another value returning function that finds the lowest price in the array and have the program print that value.

Exercise 5: After completing all the exercises above, run the program again with the values from Exercise 1 and **record your results.**

Exercise 6: Look at the following table that contains quarterly sales transactions for three years of a small company. Each of the quarterly transactions are integers (number of sales) and the year is also an integer.

YEAR	Quarter 1	Quarter 2	Quarter 3	Quarter 4
2000	72	80	60	100
2001	82	90	43	98
2002	64	78	58	84

We could use a two-dimensional array consisting of 3 rows and 5 columns. Even though there are only four quarters we need 5 columns (the first column holds the year).

Retrieve `quartsal.cpp` from the Lab 7 folder. The code is as follows:

```
// This program will read in the quarterly sales transactions for a given number
// of years. It will print the year and transactions in a table format.
// It will calculate year and quarter total transactions.

// PLACE YOUR NAME HERE

#include <iostream>
#include <iomanip>
using namespace std;

const  MAXYEAR = 10;
const  MAXCOL = 5;

typedef int SalesType[MAXYEAR][MAXCOL];   // creates a new 2D integer data type

void   getSales(SalesType, int&);         // places sales figures into the array
void   printSales(SalesType, int);        // prints data as a table
void   printTableHeading();               // prints table heading


int main()

{
      int yearsUsed;                      // holds the number of years used

      SalesType sales;                    // 2D array holding
                                          // the sales transactions


      getSales(sales, yearsUsed);         // calls getSales to put data in array
      printTableHeading();                // calls procedure to print the heading
      printSales(sales, yearsUsed);       // calls printSales to display table

      return 0;
}


//***********************************************************************
//                           printTableHeading
//  task:     This procedure prints the table heading
//  data in:  none
//  data out: none
//
//***********************************************************************


void printTableHeading()

{
      cout << setw(30) << "YEARLY QUARTERLY SALES" << endl << endl << endl;

      cout << setw(10) << "YEAR" << setw(10) << "Quarter 1"
           << setw(10) << "Quarter 2" << setw(10) << "Quarter 3"
           << setw(10) << "Quarter 4" << endl;
}



//***********************************************************************
//                           getSales
//
//  task:     This procedure asks the user to input the number of years.
//            For each of those years it asks the user to input the year
//            (e.g. 2004), followed by the sales figures for each of the
//            4 quarters of that year.  That data is placed in a 2D array
//  data in:  a 2D array of integers
//  data out: the total number of years
//
//***********************************************************************

void   getSales(SalesType  table, int&  numOfYears)
{

      cout << "Please input the number of years (1-" << MAXYEAR << ')' << endl;
      cin >> numOfYears;


      // Fill in the code to read and store the next value
```

continues

```
}


//***********************************************************************
//                           printSales
//
//  task:     This procedure prints out the information in the array
//  data in:  an array containing sales information
//  data out: none
//
//***********************************************************************



void   printSales(SalesType table, int numOfYears)
{
      // Fill in the code to print the table
}
```

Fill in the code for both `getSales` and `printSales`.

This is similar to the `price.cpp` program in Exercise 1; however, the code will be different. This is a table that contains something other than sales in column one.

Exercise 7: Run the program so that the chart from Exercise 6 is printed. **Take a screenshot.**

LAB 6.4 Student Generated Code Assignments (Optional; required exercises follows).

Option 1: Write the complete age population program given in the Lab Reading Assignment.

Statement of the problem:

Given a list of ages (1 to 100) from the keyboard, the program will tally how many people are in each age group.

Sample Run:

```
Please input an age from one to 100, put -99 to stop
5
Please input an age from one to 100, put -99 to stop
10
Please input an age from one to 100, put -99 to stop
100
Please input an age from one to 100, put -99 to stop
20
Please input an age from one to 100, put -99 to stop
5
Please input an age from one to 100, put -99 to stop
8
Please input an age from one to 100, put -99 to stop
20
```

```
Please input an age from one to 100, put -99 to stop
5
Please input an age from one to 100, put -99 to stop
9
Please input an age from one to 100, put -99 to stop
17
Please input an age from one to 100, put -99 to stop
-99

The number of people 5 years old is 3
The number of people 8 years old is 1
The number of people 9 years old is 1
The number of people 10 years old is 1
The number of people 17 years old is 1
The number of people 20 years old is 2
The number of people 100 years old is 1
```

Option 2: Write a program that will input temperatures for consecutive days. The program will store these values into an array and call a function that will return the average of the temperatures. It will also call a function that will return the highest temperature and a function that will return the lowest temperature. The user will input the number of temperatures to be read. There will be no more than 50 temperatures. Use `typedef` to declare the array type. The average should be displayed to two decimal places.

Sample Run:

```
Please input the number of temperatures to be read
5
Input temperature 1:
68
Input temperature 2:
75
Input temperature 3:
36
Input temperature 4:
91
Input temperature 5:
84

The average temperature is 70.80
The highest temperature is 91.00
The lowest temperature is 36.00
```

Option 3: Write a program that will input letter grades (A, B, C, D, F), the number of which is input by the user (a maximum of 50 grades). The grades will be read into an array. A function will be called five times (once for each letter grade) and will return the total number of grades in that category. The input to the function will include the array, number of elements in the array and the letter category (A, B, C, D or F). The program will print the number of grades that are A, B, etc.

Sample Run:

```
Please input the number of grades to be read in. (1-50)
6
All grades must be upper case A B C D or F
Input a grade
A
Input a grade
C
Input a grade
A
Input a grade
B
Input a grade
B
Input a grade
D

Number of A=2
Number of B=2
Number of C=1
Number of D=1
Number of F=0
```