Jeremy Scheuerman

Dr. Wang

Cosc 120

Lab 7 Arrays

7.1

1.

```
D:\downloads\Lab_export(4)\Lab7\testscore.exe
Please input a grade from 1 to 100, (or -99 to stop)
98
78
67
101
0
87
98
89
98
77
66
78
56
65
45
88

The average of all the grades is 73.5333

The highest grade is 101
The Lowest grade is 0

Process returned 0 (0x0)   execution time : 17.213 s
Press any key to continue.
```

2.

```
D:\downloads\Lab_export(4)\Lab7\testscore.exe
Please input a grade from 1 to 100, (or -99 to stop)
90
45
73
62
-99

The average of all the grades is 8.7693e+008

The highest grade is 1878353108
The Lowest grade is -99

Process returned 0 (0x0)   execution time : 8.755 s
Press any key to continue.
```

3.

```
D:\downloads\Lab_export(4)\Lab7\testscore.exe
Please input a grade from 1 to 100, (or -99 to stop)
The grades from the file are
90
45
73
21
62

The average of all the grades is 58.2

The highest grade is 90
The Lowest grade is 21

Process returned 0 (0x0)   execution time : 0.112 s
Press any key to continue.
```

(note a misspelling caused me errors here)

Source Code

// 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>
#include <fstream>
#include <string>
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.
ifstream dataIn;
dataIn.open("gradefile.txt");
numberOfGrades = 5;        // Fill blank with appropriate identifier

// Read in the values into the array
pos = 0;
cout << "Please input a grade from 1 to 100, (or -99 to stop)" << endl;
cout<<"The grades from the file are"<<endl;
//cin >> grades[pos];

while (grades[pos] != -99)
{
   // Fill in the code to read the grades
   dataIn>>grades[pos];
   cout<<grades[pos]<<endl;
   pos++;
   if (pos==numberOfGrades)
   {
      break;
   }
}
```

```
    // call to the function to find average
    avgOfGrades = findAverage(grades, numberOfGrades);

    cout << endl << "The average of all the grades is " << avgOfGrades << endl;
    highestGrade= findHighest(grades,numberOfGrades);

    // 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
    lowestGrade= findLowest(grades,numberOfGrades);
    cout<<"The Lowest grade is "<<lowestGrade<<endl;

    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
}



int       findHighest(const GradeType array, int size)
{
   int highest=0;
   for(int i=0; i<size; i++)
```

```
    {

        if (array[i]>highest)
        {
            highest=array[i];
            //reassign
        }
        // Fill in the code for this function

    }
    return highest;
}



//*****************************************************************************
// 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)
{
   int lowest=100;
   for(int i=0; i<size; i++)
   {

      if (array[i]<lowest)
      {
         lowest=array[i];
         //reassign
      }
      // Fill in the code for this function

   }
   return lowest;
}
```

7.2

1.

```
D:\downloads\Lab_export(4)\Lab7\student.exe
Please input the last name of the student
Smith

Please input a grade
98

Please input a grade
78

Please input a grade
89

Please input a grade
99
Jeremy Smith has an average of 91.00 which gives them a letter grade of A


Please input a y if you want to input more students any other character will stop the input
y
Please input the first name of the student
John

Please input the last name of the student
Doe

Please input a grade
88

Please input a grade
90

Please input a grade
65

Please input a grade
45
John Doe has an average of 72.00 which gives them a letter grade of C


Please input a y if you want to input more students any other character will stop the input
```

Source Code

```
// This program will input an undetermined number of student names

// 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.

typedeffloat 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
    char letter;

    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;
      cin>>grades[count];

      // 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
        average=findGradeAvg(grades,numOfGrades);
        cout<<average;

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

        letter=findLetterGrade(average);
        cout<<" which gives them a letter grade of "<<letter;

        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;
}

//*********************************************************************
// 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)
{
   int total_grades=0;
   float avg_grades=0;
   for (int i=0; i<numGrades; i++)
   {
      total_grades+=array[i];
      //add all to total
   }
   avg_grades=total_grades/numGrades;
   return avg_grades;
   // 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)
{
   char letterGrade;
   if (numGrade>=90)
   {
      letterGrade='A';
   }
   else if((numGrade>=80)&&(numGrade<90))
   {
      letterGrade='B';

   }
```

```
    else if((numGrade>=70)&&(numGrade<80))
    {
        letterGrade='C';


    }
    else if((numGrade>=60)&&(numGrade<70))
    {
        letterGrade='D';


    }
    else if(numGrade<60)
    {
        letterGrade='F';
    }
    else
    {
        letterGrade='W';
    }
// Fill in the code for this function
    return letterGrade;
}
```

7.3

1.

```
D:\downloads\Lab_export(4)\Lab7\price.exe
Please input the number of rows from 1 to 10
3
Please input the number of columns from 1 to 10
3
Input the price of an item with 2 decimal places
34.4
Input the price of an item with 2 decimal places
4.35
Input the price of an item with 2 decimal places
4.65
Input the price of an item with 2 decimal places
6.76
Input the price of an item with 2 decimal places
6.56
Input the price of an item with 2 decimal places
4.32
Input the price of an item with 2 decimal places
9.76
Input the price of an item with 2 decimal places
6.54
Input the price of an item with 2 decimal places
4.56

34.40 4.35 4.65
6.76 6.56 4.32
9.76 6.54 4.56
Process returned 0 (0x0)   execution time : 28.908
Press any key to continue.
```

2.

The getPrices array values are pass by reference because of the pointer.

The printPrices are pass by value because they do not use a pointer.

3-4

```
D:\downloads\Lab_export(4)\Lab7\price.exe
Please input the number of rows from 1 to 10
2
Please input the number of columns from 1 to 10
3
Input the price of an item with 2 decimal places
23.45
Input the price of an item with 2 decimal places
1.90
Input the price of an item with 2 decimal places
109.87
Input the price of an item with 2 decimal places
87.56
Input the price of an item with 2 decimal places
4.32
Input the price of an item with 2 decimal places
24.65

23.45 1.90 109.87
87.56 4.32 24.65
The highest price is 109.87
The lowest price is 1.90

Process returned 0 (0x0)   execution time : 18.365 s
Press any key to continue.
```

5.

```
Please input the number of rows from 1 to 10
2
Please input the number of columns from 1 to 10
3
Input the price of an item with 2 decimal places
1.45
Input the price of an item with 2 decimal places
2.56
Input the price of an item with 2 decimal places
12.98
Input the price of an item with 2 decimal places
87.86
Input the price of an item with 2 decimal places
102.34
Input the price of an item with 2 decimal places
67.89

1.45 2.56 12.98
87.86 102.34 67.89
The highest price is 102.34
The lowest price is 1.45

Process returned 0 (0x0)   execution time : 25.168 s
Press any key to continue.
```

Source Code 1

// 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 int MAXROWS = 10;
const int 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
float findHighestPrice(PriceType table, int numOfRows, int numOfCols);
float findLowestPrice(PriceType table, int numOfRows, int numOfCols);
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
    cout<<"The highest price is "<<findHighestPrice( priceTable,rowsUsed,colsUsed)<<endl;
    cout<<"The lowest price is "<<findLowestPrice( priceTable,rowsUsed,colsUsed)<<endl;

    return 0;
}
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;
}
float findLowestPrice(PriceType table, int numOfRows, int numOfCols)
// This function returns the highest price in the array
{
    float lowestPrice;
```

```
   lowestPrice = table[0][0]; // make first element the highest price
   for (int row = 0; row < numOfRows; row++)
      for (int col = 0; col < numOfCols; col++)
         if (lowestPrice > table[row][col] )
            lowestPrice = table[row][col];
   return lowestPrice;
}

//*****************************************************************************
***
//         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++)
        {
            cout<<"Input the price of an item with 2 decimal places"<<endl;
            cin>>table[row][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++)
    {
        cout<<endl;
        for (int col = 0; col < numOfCols; col++)
        {
            cout<<table[row][col]<<" ";
            // Fill in the code to print the table
        }
    }
    cout<<endl;


}
```

6.

```
D:\downloads\Lab_export(4)\Lab7\quartsal.exe
Please input Year 7 quarter 4
46
Please input Year 8 quarter 1
4
Please input Year 8 quarter 2
56
Please input Year 8 quarter 3
4
Please input Year 8 quarter 4
65
Please input Year 9 quarter 1
546
Please input Year 9 quarter 2
46
Please input Year 9 quarter 3
54
Please input Year 9 quarter 4
56
Please input Year 10 quarter 1
4
Please input Year 10 quarter 2
4
Please input Year 10 quarter 3
5
Please input Year 10 quarter 4
6
        YEARLY QUARTERLY SALES


      YEAR Quarter 1 Quarter 2 Quarter 3 Quarter 4
      2000        45       654        34         5
      2001         4      4434        23        65
      2002        45        23        54        34
      2003        65        76        87        67
      2004        56        45        34        34
      2005        23        54        67       546
      2006       456         4        56        46
      2007         4        56         4        65
      2008       546        46        54        56
      2009         4         4         5         6

Process returned 0 (0x0)   execution time : 45.331 s
Press any key to continue.
```

7.

```
D:\downloads\Lab_export(4)\Lab7\quartsal.exe
Please input the number of years (1-10)
3
Please input Year 1 quarter 1
72
Please input Year 1 quarter 2
80
Please input Year 1 quarter 3
60
Please input Year 1 quarter 4
100
Please input Year 2 quarter 1
82
Please input Year 2 quarter 2
90
Please input Year 2 quarter 3
43
Please input Year 2 quarter 4
98
Please input Year 3 quarter 1
64
Please input Year 3 quarter 2
78
Please input Year 3 quarter 3
58
Please input Year 3 quarter 4
84
        YEARLY QUARTERLY SALES


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

Process returned 0 (0x0)   execution time : 36.379 s
Press any key to continue.
```

Source Code

// 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 int MAXYEAR = 10;
const int 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
```

```
    int row=0;
    int col=0;
    for (row=0; row<numOfYears; row++)
    {
        for(col=1; col<5; col++)
        {
            cout<<"Please input Year "<<row+1<<" quarter "<< col<<endl;
            cin>>table[row] [col];
            //populate years on left hand side
        }
    }

}

//*************************************************************************
*
//          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
   int j=0;
   //years counter
   int year=2000;

   for (int row=0; row<numOfYears; row++)
   {
      for(int col=0; col<5; col++)
      {
         if (col==0){
         table[row][col]=2000+j;
         cout<<setw(10)<<table[row][col];
         //populate years on left hand side
         }else{
         cout<<setw(10)<<table[row] [col];
         }
      }
      j++;
```

```
        cout<<endl;
    }


}
```

7.4

Option 1

```
"D:\Documents\School\Year 3 Semester 1\COSC 120\Lab_7\lab_7.4.exe"
Please input an age from 1 to 100, put -99 stop
3
Please input an age from 1 to 100, put -99 stop
3
Please input an age from 1 to 100, put -99 stop
65
Please input an age from 1 to 100, put -99 stop
45
Please input an age from 1 to 100, put -99 stop
4
Please input an age from 1 to 100, put -99 stop
5
Please input an age from 1 to 100, put -99 stop
3
Please input an age from 1 to 100, put -99 stop
3
Please input an age from 1 to 100, put -99 stop
-99
The number of people 3 years old  is 4
The number of people 4 years old  is 1
The number of people 5 years old  is 1
The number of people 45 years old  is 1
The number of people 65 years old  is 1

Process returned 0 (0x0)   execution time : 34.979
Press any key to continue.
```

Source Code

```
#include <iostream>


using namespace std;
int main()
{
    int arr[100];


    for (int i=1; i<100; i++)
    {
        arr[i]=0;
//populate at 0
    }


int age=0;
    while (age!=-99)
{
    cout<<"Please input an age from 1 to 100, put -99 stop"<<endl;
    cin>>age;
    arr[age]+=1;


    }
    for (int i=1; i<100; i++)
```

```
    {
//populate at 0


        if (arr[i]!=0)
    {
        cout<<"The number of people "<<i<<" years old "<<" is "<<arr[i]<<endl;
            //print out amount of people
        }
    }
}
```

Option 2

```
#include <iostream>
using namespace std;
const int MAX_AMT=50;
typedef float temp[MAX_AMT];


int main()
{
    int num=0;
    double lowest=1000;
    double highest=0;
    double avg=0;
    while (num>50)
```

```
{

    cout<<"Please input the number of temperatures to be read"<<endl;
    cin>>num;
}
temp temperature  [num];
for (int i=0; i<num-1; i++)
{
    cout<<"Input temperature "<<i<<":"<<endl;
    cin>>temperature[i];
    if (temperature[i]<lowest)
    {
        lowest=temperature[i];
        //get lowest
    }
    if (temperature[i]>highest)
    {
        highest=temperature[i];
        //get highest
    }
    avg+=temperature[i];
}
avg=avg/50;
```

```
    //get average

    cout<<"The average temperature is "<<avg;

    cout<<"The highest temperature is "<<highest;

    cout<<"The lowest temperature is "<<lowest;

    //print


}


return 0;

}
```

Option 2

```
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 70.8
The highest temperature is 91.00
The lowest temperature is 36.00

Process returned 0 (0x0)   execution time : 12.549 s
Press any key to continue.
```

```
"D:\Documents\School\Year 3 Semester 1\COSC 120\Lab_7\Lab_7.4_op_2.exe"
Please input the number of temperatures to be read
6
Input temperature 1:
56
Input temperature 2:
44
Input temperature 3:
32
Input temperature 4:
65
Input temperature 5:
87
Input temperature 6:
5
The average temperature is 48.17
The highest temperature is 87.00
The lowest temperature is 5.00

Process returned 0 (0x0)   execution time : 7.229 s
Press any key to continue.
```

Source Code

```
#include <iostream>

#include <iomanip>

using namespace std;

const int MAX_AMT=50;

typedef float temp[MAX_AMT];

float get_average(temp temperature,int num);

float get_highest(temp temperature,int num);

float get_lowest(temp temperature,int num)

{

    float lowest=1000;
```

```
    for (int i=0; i<num; i++)
    {
        if (temperature[i]<lowest)
        {
            lowest=temperature[i];
            //get lowest
        }
    }
    return lowest;
}
float get_highest(temp temperature,int num)
{
    float highest=0;
    for (int i=0; i<num; i++)
    {
        if (temperature[i]>highest)
        {
            highest=temperature[i];
            //get highest
        }
    }
    return highest;
}
```

```
float get_average(temp temperature,int num)
{
    float avg=0;
    for (int i=0; i<num; i++)
    {
        avg+=temperature[i];
    }
    avg=avg/num;
    return avg;
}
int main()
{
    int num=51;
    float avg=0;
    float highest=0;
    float lowest=0;
    while (num>50)
    {

        cout<<"Please input the number of temperatures to be read"<<endl;
        cin>>num;
    }
    temp temperature;
```

```
    for (int i=0; i<num; i++)
    {
        cout<<"Input temperature "<<i+1<<":"<<endl;
        cin>>temperature[i];
        //assign
    }
    avg=get_average(temperature,num);
    //get average
    highest=get_highest(temperature,num);
    //get highest
    lowest=get_lowest(temperature,num);
    //get lowest
    cout<<"The average temperature is "<<fixed<<setprecision(2)<<avg<<endl;;
    cout<<"The highest temperature is "<<fixed<<setprecision(2)<<highest<<endl;
    cout<<"The lowest temperature is "<<fixed<<setprecision(2)<<lowest<<endl;
    //print

    return 0;
}
```

Option 3

```
"D:\Documents\School\Year 3 Semester 1\COSC 120\Lab_7\lab_7.4_op3.exe"
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

Process returned 0 (0x0)   execution time : 10.548 s
Press any key to continue.
```

Source Code

```
#include <iostream>



using namespace std;

int number_grades(char grade_arr[],char letter,int num);

int number_grades(char grade_arr[],char letter,int num)

{

    int amnt =0;

    //return letter
```

```
    for (int i=0; i<num; i++)
    {
        if (grade_arr[i]==letter)
        {
            amnt++;
            //tally letters
        }


    }
    return amnt;
}


int main()
{
    int num=0;
    int amnt_A=0;
    int amnt_B=0;
    int amnt_C=0;
    int amnt_D=0;
    int amnt_F=0;
    //declare amounts
    cout<<"Please input the number of grades to be read in.(1-50)"<<endl;
    cin>>num;
```

```
    char grades[num];
    cout<<"All grades must be upper case A B C D or F"<<endl;
    for (int i=0; i<num; i++)
    {
        cout<<"Input a grade"<<endl;
        cin>>grades[i];


    }
    amnt_A=number_grades(grades,'A',num);
    amnt_B=number_grades(grades,'B',num);
    amnt_C=number_grades(grades,'C',num);
    amnt_D=number_grades(grades,'D',num);
    amnt_F=number_grades(grades,'F',num);
    //get amnts
    cout<<endl<<"Number of A="
        <<amnt_A<<endl;
    cout<<"Number of B="<<amnt_B<<endl;
    cout<<"Number of C="<<amnt_C<<endl;
    cout<<"Number of D="<<amnt_D<<endl;
    cout<<"Number of F="<<amnt_F<<endl;
    //output
    return 0;
}
```