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

// Michael Steele

#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, int, int);
float findLowestPrice(PriceType, int, int);
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 << "Highest value: " << findHighestPrice(priceTable, rowsUsed, colsUsed) << endl;
   cout << "Lowest value: " << findLowestPrice(priceTable, rowsUsed, colsUsed) << endl;

        return 0;
}

//**************************************************************************
//      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++)
                {
                        int price=0;
                        // Fill in the code to read and store the next value in the array
                        cout << "Enter a value to store with two decimal places " << endl;
                        cin >> price;
                        table[row][col] = price;
                }
        }
}

//*****************************************************************************
//        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++)
                {
```

```
                    cout << "    " <<table[row][col];

                }
                cout << endl;
            }


}
float findHighestPrice(PriceType table, int numOfRows, int numOfCols)
{
   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)
{
   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;
}
```