CISC 105 Spring 2007 Project 2

Due Monday, April 16th at midnight

on MyCourses, paper your TA the next day.

This project is about using parallel one-dimensional arrays and functions to extract information from a data set, and reading data from a file.

Check this directory periodically for updates. When I get questions from students that I think are of interest to the whole class I will put them in an FAQ file in this project's directory.

The identifiers shown in this assignment are not required, but be sure that the identifiers you choose are very clear and follow the style rules we have discussed in class.

Do you see an error or logical flaw in this assignment? Be the first to report it and get a point added to your project grade. Check to be sure you have the latest copy first! No student can get more than three points this way.

Inventory Management

Our simple inventory management system will keep track of a product ID number, the price, the number of items on hand, and the item weight.

ID number	price	supply	lbs	oz
21	3.00	15	1	2
312	3451.12	2	825	2
2	12.05	23	6	15
8970	0.99	45	0	3
45	2.00	980	0	13
17	5.98	6	2	1
38	0.01	21000	0	1

You may assume the following:

- The product ID number is a positive integer;
- The price is a double, and may be negative;
- Supply, or items on hand, is an integer;
- Weight in pounds is an integer, and ounces is also an integer.

1. (20 pts) Write a main function with five arrays of size 50. The first array will hold ID numbers, the second will hold price numbers, etc. All of the information for a product will be in the same index position of every array. In the example data above, productID[0] would contain 21, price[0] would hold 3.00, weightLBS[0] would contain 1, and so on.

   Initialize each array to the example values shown (so the product ID number array would contain 21,312,2,8970,45,17,38. How can you tell if your initialization is working correctly?) Then use assignment to place the defined constant sentinel -1 in the ID number array after the last valid datum.

2. (10 pts) Write code in main to traverse your data and print the array index associated with a given product ID number. Use a sentinel loop that looks for both the ID number and the sentinel that marks the end of the data. If the item is not found before the end of the valid product IDs, return a defined constant value that means "not found" in your program.

   Once your code is working, copy the file (to preserve a working file!). Now move your code to a function that will return the index when you give it an ID "key" to look for in the array of ID numbers:

   ```
   int getIndexFromID( int productID[], int keyID);
   ```

   Like most functions in the project, this function will not print anything; the printing will be done in the calling function.

3. (10 pts) Write a function that will return the price when given a *valid* index. To do this, the function must have access to the price array:

   ```
   double getPriceFromIndex( double price[], int index);
   ```

   Similarly, write three more functions that will get supply on hand, weight in pounds, and weight in ounces when given an index. Use these four functions in your program whenever you need this kind of information. This practice ensures that if you change the way your data is organized later, you will only need to change these functions to keep your program working correctly.

4. (10 pts) Add a menu with a switch statement to your program. Place code in your menu to get an index from an ID, show the price from an ID, and show the minimum price item. As you add each function below to your program, also add it to your menu.

5. (15 pts) Alter main() so that it will read the data from a file into the arrays declared in main. Have the user enter the number of data lines expected in the file. Read that many lines using fscanf and put each data field in the appropriate array. Then update the sentinel in the product ID array.

   Once this works, create a function that is called from your menu so that the user can have the program read a different inventory file at any time. If a new file is read, it should replace any previous data. The function will take all of the arrays and the expected data size as parameters.

6. (10 pts)Write a function that will print the whole inventory list with appropriate headers and nicely aligned columns of numbers.

7. (5 pts) Write a function to traverse your data and find the index of the product with the minimum price. (What other function should this function call?) Have the *calling* function print:

   ```
   The lowest priced product has ID number 38 and price $0.01.
   ```

8. (10 pts) Create a function for ordering products. Ask a user for product ID and quantity repeatedly until user enters a sentinel for product ID. Keeps a running total of the amount user spends, and returns the total. If the user orders more of a product than is in supply, a nice error message is printed and the user is asked for ID and quantity again.

For an A:

A project that has all of the specification working up to this point is eligible for 90 points out of 100. Once you have *all* of the above code working, you may work on the following. No points will be given for the following functions unless the previous sections are complete.

9. (3 pts) Write a function that, given a product ID and quantity, will calculate the total weight and report it in pounds and ounces.

10. (3 pts) Make another order function so that the user can order products by ID number and total weight.Ask a user for product ID and weight repeatedly until user enters a sentinel.

11. (4 pts) Write a function that allows the user to add a new product. The function will prompt the user for all product data and put the data into the arrays after the existing data.

Submission

Design and test your program carefully using data that you make up. Check the calculations! Monday before the project is due you will be given test data and a project submission sheet. You must show that your program performs correctly for all test data, and you must complete the **submission sheet** and turn it in with your project's **paper copy** (when?). Of course, code for your program must be submitted to **MyCourses** along with your script file.