Name: Minh Anh Bui

**CS 37 - Exam #2**

**20 Points Possible**

(7) 1. Write a function template that uses the **return** statement to return the smallest value in an array. The array size is also passed to the function template. Please only write the function template, no main is needed.

template <class T>

T smallest(T\*arr, int size)

{

T small, temp;

int i, j;

for (i = 0; i < size; i++)

{

for (j = 0; j < size; j++)

{

if (arr[j+1] < arr[j])

{

temp = arr[j];

arr[j] = arr[j+1];

arr[j+1] = arr[j];

}

}

}

small = arr[0];

return small;

}

(7) 2. A file of stock names, number of shares, buy prices, and current prices has already been saved to a text file called STOCKS.DAT. The stock name is on the first line of the file and the second line for each stock contains their number of shares, buy price, and current price.

Write a program fragment (**NOT A COMPLETE PROGRAM, NO DECLARATIONS, NO FUNCTIONS**) which will create a text file called PROFITS.DAT which will contain the stock's name on the first line of the file and the stock’s total cost (shares \* buy price), current value (shares \* current price), and profit (current value – total cost) on the second line of the file.

A picture of the files is as follows:

|  |  |  |
| --- | --- | --- |
| Joe’s Technology  150 69.34 77.55  Bummer Inc  50 25.00 15.00  .  .  . |  | Joe’s Technology  10401.00 11632.50 1231.500  Bummer Inc  1250.000 750.0000 -500.0000  .  .  . |

STOCKS.DAT PROFITS.DAT

ifstream infile ("c:\\stocks.dat", ios::in);

ofstream outfile ("c:\\profits.dat", ios::out);

while (infile.getline(name, 50))

{

infile >> shares >> buyPrice >> curPrice;

totCost = shares\*buyPrice;

curValue = shares\*curPrice;

profit = curValue - totCost;

outfile << name << endl << totCost << ' ' << curValue << ' ' << profit << endl;

infile.ignore(50, '\n');

}

infile.close();

outfile.close();

(6) 3. Short Answer Section.

a. What is an inline function?

Inline functions substitute a very small amount of code onto the call function. For example, an inline function called “doubling” may have the return statement “return num\*2;”, hence doubling the number. What happens when the program calls the inline function in its main body is that it literally copies the segment of code “num\*2” and replace it with the main’s local variable that was called into the function. Hence, cout << doubling(number); will literally turn into cout << number\*2; The computer will also store the return address of the next instruction to do in this process, so that it will know where to go after having taken return num\*2; from the function. Note that this only works for small functions because running elaborate code in an inline function implies that the supposed piece of code is suitable for substituting where the call function is.

b. What does the scope resolution operator do?

It allows access to variables and methods that are outside of the current scope, and it’s represented as ::

For example, if int someNumber; is declared outside of main (as a global variable), cout << ::someNumber; will access that number outside the scope of your current function.

c. What are overloaded functions?

Overloaded functions are a group of functions with the same name, but with different parameters. When the function is called, the particular function that the program will actually use will depend on what parameters is inputted.

d. Why are overloaded functions used?

Overloaded functions are used because it’s convenient for the programmer. The programmer doesn’t need to remember too many different function names with the same purpose, just because they have different parameters. The type of parameter passed through is taken into consideration, and the programmer doesn’t have to worry about the type of input the user might put in so long as the overloaded functions cover them.

e. (True/False) Only the rightmost arguments can be used as default arguments in a parameter list. True

f. (True/False) Function pointers can work for functions with different signatures. False