CS 115 Midterm 2A Solutions

November 13, 2008

Rules

- You must briefly explain your answers to receive partial credit.
- When a snippet of code is given to you, you can assume that the code is enclosed within some function, even if no function definition is shown. You can also assume that the `main` function is properly defined and that the `iostream`, `fstream`, `iomanip`, `string`, and `cmath` libraries have been included at the beginning of the program.
- When you are asked to write *a snippet* of code, you may also assume that it is enclosed within some function that any necessary libraries have been included.
- When you are asked to write *a complete program*, you must write the `#include` statements, the `int main()`, etc. in your solution to receive full credit.
- A line consisting solely of "..." represents one or more unspecified C++ statements, some of which may change the values of program variables.
- You are encouraged to use the backs of these pages for scratch paper. If you want answers written there to be graded, they must be very clearly labeled and also noted on the main test, e.g. "See the back of page 1 for 3a."

Problem 1: 15 points.

What is the output of each of the following snippets of code?

(a) (6 points)

```
int a = 5;
int *ptr = &a;
*ptr += 5;
cout << *ptr << endl;
cout << a << endl;
```

10
10

(b) (3 points)

```
int a[6] = {5, 8, 7, 9, 13, -1};
cout << a[4] << endl;
```

13

(c) (6 points)

If the following function is defined somewhere in the program and prototyped above main….

```
int square(int& x) {
    return x*x;
}
```

…what does the following code print?

```
int x = 5;
int y = square(x);
cout << x << endl;
cout << y << endl;
```

5
25

Problem 2: 10 points.

(a) (5 points)

For the snippet of code…

```
float f[50];
```

…what is the datatype of `f[15]`?

float

(b) (5 points)

Assume that the following declaration appears above the main program:

```
struct inventory {
     string flavor;
     int num_scoops;
     float price;
     float revenue;
};
```

For the snippet of code…

```
inventory inv[5];
```

…what is the datatype of `inv`?

array of inventory

Problem 3: 25 points.

The snippets of code in this problem do not successfully accomplish the task described in their accompanying comment. Correct the code so that it performs the task described in the comment. The code may have more than one error. **Make your corrections clear and unambiguous.**

(a) (10 points)

```
/* Function that finds and returns the smallest
element of an integer array.  Inputs are the array and
its size */
int FindMin(int[] array, int size) {
     int min = 0;
     for (int i=0; i <= size; i++) {
          if (i < min) {
               min = array[i];
          }
     }

}
```

5 corrections (underlined):

```
int FindMin(int array[], int size) {
        int min = array[0];
        // Note: we can now start at i=1 but don't have to
        for (int i=0; i < size; i++) {
            if (array[i] < min) {
                min = array[i];
            }
        }
        return min;
}
```

(b) (8 points)

For this problem, you may assume that the `struct inventory` from Problem 2b has been declared above the main program.

```
/* Update the ice cream inventory and revenue when the
user buys scoops_bought scoops.  Assume that
scoops_bought is an integer greater than or equal to
0.   */
inventory my_store;
int scoops_bought;
…
inventory[num_scoops] -= scoops_bought;
inventory[revenue] += price;
```

6 corrections (underlined):

inventory my_store;

int scoops_bought;

…

my_store.num_scoops -= scoops_bought;

my_store.revenue += my_store.price * scoops_bought;

(c) (7 points)

```
/* Function looks up the string search_string in the
array arr.   Returns the position of search_string in
the array, or -1 if the string is not found. */

int findStringPos(string search_string, string arr[],
int size) {
     for (int i=0; i<= size; i++) {
          if (search_string = arr[i]) {
               return i;
          }
          else {
               return -1;
          }
     }
}
```

3 corrections (underlined):

```
int findStringPos(string search_string, string arr[], int size) {
      for (int i=0; i< size; i++) {
            if (search_string == arr[i]) {
                  return i;
            }
            else {
                  return -1;
            }
      }
      return -1;
}
```

Problem 4: 25 points.

Write short snippets of code to accomplish the following tasks:

(a) (7 points)

For an array that has been declared as

```
float floatArr[5][8];
```

write a snippet of code that computes and prints the average of all the elements in the **entire array**.

```
float sum = 0;
for (int i=0; i< 5; i++) {
  for (int j=0; j<8; j++) {
        sum += floatArr[i][j];
  }
}
cout << sum/40;
```

(b) (6 points)

For this problem, you may assume that the `struct inventory` from Problem 2b has been declared above the main program. Write a snippet of code that declares an `inventory` array of size 50 and initializes the `revenue` to zero for each element of the array.

```
inventory invArr[50];
for (int i=0; i<50; i++) {
  invArr[i].revenue = 0;
 }
```

(c) (12 points)

Write a snippet of code that repeatedly asks for the user's input as a string. If the user's input is `update`, your code should call the function `Update()`, which has no inputs. If the string is `print`, your code should call the function `Print()`, which has no inputs. If the string is `quit`, you should print a goodbye message and stop printing the menu. If the string is anything else, you should print an error message and reprint the menu.

```
string user_input;
bool quit = false;

do {
  cout << "Enter your input: ";
  cin >> user_input;
  if (user_input == "update") {
    Update();
  }
  else if (user_input == "print") {
    Print();
  }
  else if (user_input == "quit") {
   cout << "Goodbye! " << endl;
  }
  else {
   cout << "Error!" << endl;
  }
} while (user_input != "quit");
```

Problem 5: 25 points.

For this problem, you must write a **complete program** that contains the following:

- The definition of a `struct` called `MenuItem` that has two fields. You should decide the appropriate data types for these fields:
  - `itemName`, the name of a menu item
  - `price`, its price in dollars

- A function called `ComputeAvg` that computes the average price of an array of `MenuItem`. Its return value is the average price, as a float. Its inputs are:
  - `inputArr`, an array of `MenuItem`
  - `size`, the size of the array

- Prototype for `ComputeAvg`

- A `main` function that does the following:
  - Declares **an array** of 3 `MenuItems`
  - Initializes the menu items as follows:
    - The first item is `coffee`, for `1.25`
    - The second item is `bagel`, for `2.25`
    - The third item is `smoothie`, for `3.75`
  - Calls `ComputeAvg` to compute the average price of an item in the menu
  - Prints the average price computed by `ComputeAvg`.

See next page.

```
#include <iostream>
#include <string>
using namespace std;

struct MenuItem {
  string itemName;
  float price;
};

float ComputeAvg(MenuItem inputArr[], int size);

int main() {
  MenuItem items[3];
  items[0].itemName = "coffee";
  items[0].price = 1.25;
  items[1].itemName = "bagel";
  items[1].price = 2.25;
  items[2].itemName = "smoothie";
  items[2].price = 3.75;
  cout << ComputeAvg(items, 3);
  return 0;
}

float ComputeAvg(MenuItem inputArr[], int size) {
  float sum = 0;
  for (int i=0; i<size; i++) {
    sum += inputArr[i].price;
  }
  return sum/size;
}
```