CS 11 Data Structures and Algorithms

Assignment 1: Pointers

Return to Course Homepage

Assignment 1.1

This solution does not include documentation.

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

void noNegatives(int *x);
void swap(int* x, int* y);


int main(){

    int x, y; // 1.
    int* p1;  // 2.

    p1 = &x;  // 3.
    *p1 = 99; // 4.
    cout << "x contains: " << x << endl;   // 5.
    cout << "x contains: " << *p1 << endl; // 6.

    p1 = &y;    // 7.
    *p1 = -300; // 8.

    int temp;   // 9.
    int* p2;
    p2 = &x;

    temp = *p1; // 10.
    *p1 = *p2;
    *p2 = temp;

    // 12.
    cout << "After calling noNegatives()," << endl;
    noNegatives(&x);
    noNegatives(&y);
    cout << "x is: " << *p2 << endl; // 13.
    p2 = &y;
    cout << "y is: " << *p2 << endl;

    int a[2];  // 14.
    p2 = a;
    p2[0] = x; // 15.
    p2[1] = y; // 16.
    cout << "The address of the first  element is " << &p2[0] << endl; // 17.
    cout << "The address of the second element is " << &p2[1] << endl; // 18.

    p1 = &a[0]; // 19.
    p2 = &a[1];
    temp = *p1;
    *p1 = *p2;
    *p2 = temp;
    cout << "The first  element in a[] is " << *p1 << endl; // 20.
    cout << "The second element in a[] is " << *p2 << endl;

    // 22.
    swap(&x, &y);
    cout << "Now, x contains: " << x << " and y contains: " << y << endl;

    // 23.
    swap(&a[0], &a[1]);
    cout << "Now, a[0] contains: " << a[0] << " and a[1] contains: " << a[1] << endl;

}
```

```
// 11.
void noNegatives(int *x){
    if(*x < 0){
        *x = 0;
    }
}



//21.
void swap(int* x, int* y){
    int temp;
    temp = *x;
    *x = *y;
    *y = temp;
}
```

Assignment 1.2

```
/*
    Dave Harden
    CS 11
    Assignment 1.3

    This program receives scores and the names of people they belong to and then
    sorts them in descending order by score amount. It uses two dynamic arrays, an int
    array, scores, and a string array, names. First, the program prompts the user for
    the number of scores to be answered. The user's response is stored in "size."
    Both arrays are declared to be of length size. The program will prompt the user
    to enter a name and then prompt them to enter a score until both arrays are full.
    The name and score pairs are maintained by keeping the name and its score at the same
    index number for both arrays. Then, the program sorts them in descending order by
    score, keeping the original name and score pairs together. Finally, it prints
    the sorted scores in two columns, the names on the left and the scores on the right.
    Each row contains a name and score entered together by the user.
*/

#include <iostream>
using namespace std;

void getArraySize(int& size);
void initializeArrays(string names [], int scores[], int size);
void sortData(string names[], int scores[], int size);
int indexOfLargest(const int list[], int startingIndex, int size);
void displayData(const string names[], const int scores[], int size);

int main()
{

    string* names;
    int* scores;
    int size;

    getArraySize(size);

    names = new string[size];
    scores = new int[size];

    initializeArrays(names, scores, size);
    sortData(names, scores, size);
    displayData(names, scores, size);

    delete [] names;
    delete [] scores;
}


/*
   This function prompts the user for the number of score and name pairs they want
   to enter and stores the user's response in the int size. size is pass by
   reference making the number entered by the user available to the calling function.

   pre: none
   post: passes by reference through size the number of scores to be entered.
*/

void getArraySize(int& size){
    cout << "How many scores will you enter?: ";
    cin >> size;
```

```
}



void initializeArrays(string names[], int scores[], int size)
{
    for(int index = 0; index < size; index++)
    {
        cout << "Enter the name  for score #" << (index + 1) << ": ";
        cin >> names[index];
        cout << "Enter the score for score #" << (index + 1) << ": ";
        cin >> scores[index];
    }
    cout << endl;
}




void sortData(string names[], int scores[], int size) {
    int largestIndex;
    for (int count = 0; count < size - 1; count++){
        largestIndex = indexOfLargest(scores, count, size);
        swap(names[largestIndex],  names[count]);
        swap(scores[largestIndex], scores[count]);
    }

}




int indexOfLargest(const int list[], int startingIndex, int size){
    int targetIndex = startingIndex;

    for (int count = startingIndex + 1; count < size; count++){
        if (list[count] > list[targetIndex]){
            targetIndex = count;
        }
    }

    return targetIndex;
}



void displayData(const string names [], const int scores [], int size)
{
    cout << "Top Scorers: " << endl;
    for(int index = 0; index < size; index++)
    {
        cout << names[index] << ": " << scores[index] << endl;
    }
}
```