ENEE 140 Lab 13

Lab instructions

This handout includes instructions for the recitation sessions on Wednesday and Friday. **Follow these instructions** to review arrays and sorting operations, then **submit the homework** as indicated below.

1 Array removal and insertion

Given an integer array `a[8] = {2, 3, 5, 7, 11, 13, 17, 19}`, read three integers `k`, `m`, and `x` from the terminal. When `k` is a valid array index (i.e., between 0 and 7), remove `a[k]` from the array by shifting all the elements from `k+1` to the end of the array one place to the left (so that `a[k]` takes the value of the old `a[k+1]`, `a[k+1]` takes the value of the old `a[k+2]`, etc.). Then, if `m` is a valid array index, insert integer `x` on position `m` of the array and shift the elements from `m+1` to the end of the array one place to the right (so that `a[m+1]` takes the value of the old `a[m]`, `a[m+2]` takes the value of the old `a[m+1]`, etc.).

After these two operations, print out the elements in the array. The array should not have lost any elements other than the old `a[]`, and all the old elements should have retained their original order.

For example, if we have `k`=2, `m`=5, `x`=10 then the array will be changed to `{2, 3, 7, 11, 10, 13, 17, 19}`.

2 Insert a value in a sorted array

Given an integer array `a[9] = {2, 3, 5, 7, 11, 13, 17, 19}`, prompt the user for an integer `k` and insert `k` into the array such that the new array remains sorted. Note that the array a has size 9, but there are only 8 initial values. This ensures that no value will be lost after the insertion.

For example, when `k` = 12, the new array will become `{2, 3, 5, 7, 11, 12, 13, 17, 19}`; when `k` = -5, the new array will become `{-5, 2, 3, 5, 7, 11, 13, 17, 19}`; when `k` = 100, the new array will become `{2, 3, 5, 7, 11, 13, 17, 19, 100}`.

Homework

Due: May 1 at 11:59 pm.

Create three programs by following the instructions below. Submit them using the following commands:

```
submit 2016 spring enee 140 AAAA 13 selection_sort_clean.c
submit 2016 spring enee 140 AAAA 13 count_coins.c
submit 2016 spring enee 140 AAAA 13 insertion_sort.c
```

Note: you must replace `AAAA` with your own section number (0101, 0102, etc.)

1 Refactoring

Rewrite the C program `selection_sort_clean.c` from the class public folder on GRACE by implementing the functions invoked in the following main function. You can do this by moving the relevant code from `selection_sort_clean.c` to the body of these new functions (this operation is called "refactoring"). Your program should produce output identical to the original program.

```c
#include <stdio.h>
#define SIZE 9
int main(void)
{
  int a[SIZE];
  readArray (a, SIZE);
  printArray (a, SIZE);
  sortArray (a, SIZE);
  printf("After sorting:\n");
  printArray (a, SIZE);
  return 0;
}
```

2 Counting coins

Write a program, called `count_coins.c` that reads data from a text file called `coins_in_pocket.txt`. The file has 3 lines, each containing 4 integers. These integers correspond to the number of quarters, the number of dimes, the number of nickels, and the number of pennies in a person's pocket. Your program will then calculate the total amount of money for each set and put the three amounts in an array. Then have your program sort the three, from greatest to least amount of money and print them.

You will need to use at least one function to calculate the amount of total money from the sets of change, passing in the amount of quarters, dimes, nickels, and pennies as parameters, and returning a float to the main. You can also use the sortArray function you have implemented in the previous question to sort the final array. For example, if the input is

```
1 1 0 1
```

```
1 1 1 1
0 1 0 0
```

the output should be

```
$0.41
$0.36
$0.10
```

3 Insertion sort

Write a complete program, called `insertion_sort.c`, that sorts an array of integers. Insertion sort is similar to the selection sort algorithm in that, after the `k`-th iteration, the first `k` elements in the array are in sorted order. To sort the array `a[]` in ascending order (smallest to the largest), insertion sort works as follows:

- In iteration `k`, copy `a[k]` to a temporary variable, then insert this variable into the array region between `0` and `k-1` (which is sorted) on its correct position in the sorted order. You will have to shift the elements between that position and k one place to the right.
- Start from `k`=1, as the array region before k is trivially sorted (it has only one element).
- Before implementing the function, apply insertion sort manually to sort the following array, in order to visualize how this algorithm works.
  `{20, 5, 11, -3, 0, 34, -12, 9, 100, 4}`

4 Comparison of the sorting algorithms (optional)

Think about the two sorting algorithms we have learned by this point, selection sort and insertion sort. How many comparisons does each algorithm perform? You do not have to submit the answer to this question.

Reading assignment

Review the material for the final exam.

Weekly challenge

No challenge this week.