CPE 150 Laboratory 10: Arrays II

Department of Computer Engineering
Yarmouk University

Summer 2017

1 Objectives

- To declare arrays, initialize arrays and refer to individual array elements.
- To be able to pass arrays to functions.
- To understand basic sorting techinques.
- To understand basic searching techinques.

2 Lab Exercise 1 - Array Operations

Write a program that creates a one-dimentional array initialized with test data. The program should have the following functions:

- `void readArray(int array[], const int size)`: This function reads the elements of the array from a user.
- `void printArray(const int array[], const int size)`: This function prints the elements of the array.
- `int searchArray(const int array[], const int size)`: This function sequentially searches the array for a number and returns the first index (or -1 if not found)
- `void sortDesc(int array[], const int size)`: This function uses bubble sort to sort the array in a descending order.

3 Lab Exercise 2 - Selection Sort

Write a program to implement a selection sort algorithm. As the name implies, in the selection sort algorithm, we rearrange the list by selecting an element in the list and moving it to its proper position. This algorithm finds the location of the smallest element in the unsorted portion of the list and moves it to the top of the unsorted portion of the list. The first time, we locate the smallest item in the entire list. The second time, we locate the smallest item in the list starting from the second element in the list, and so on.

4 Lab Exercise 3 - Binary Search

The binary search is a clever algorithm that is much more efficient than the linear search. Its only requirement is that the values in the array be sorted in order. Instead of testing the array's first element, this algorithm starts with the element in the middle. If that element happens to contain the desired value, then the search is over. Otherwise, the value in the middle element is either greater than or less than the value being searched for. If it is greater, then the desired value (if it is in the list) will be found somewhere in the first half of the array. If it is less, then the desired value (again, if it is in the list) will be found somewhere in the last half of the array. In either case, half of the array's elements have been eliminated from further searching.

If the desired value wasn't found in the middle element, the procedure is repeated for the half of the array that potentially contains the value. For instance, if the last half of the array is to be searched, the algorithm immediately tests its middle element. If the desired value isn't found there, the search is narrowed to the quarter of the array that resides before or after that element. This process continues until either the value being searched for is found or there are no more elements to test.

Here is the pseudocode for a function that performs a binary search on an array:

```
Set first index to 0.
Set last index to the last subscript in the array.
Set found to false.
Set position to -1.
While found is not true and first is less than or equal to last
    Set middle to the subscript halfway between array[first] and array[last].
    If array[middle] equals the desired value
        Set found to true.
        Set position to middle.
    Else If array[middle] is greater than the desired value
        Set last to middle - 1.
    Else
        Set first to middle + 1.
    End If.
End While.
Return position.
```

Write a program to implement the binary search algorithm. To sort the array before calling the binary search algorithm, you can use the selection sort algorithm you developed earlier or the bubble sort algorithm in the textbook.

5 Postlab Exercise

Write a program that will performs rotating operations on single-dimensional array. The operations allowed are rotate left and rotate right.

```
Enter 10 elements to be stored in an array: 0 1 2 3 4 5 6 7 8 9

Enter 1 to rotate the array to the left, or 2 to rotate to the right (-1 to end): 1
Left Rotation is Selection.
Your rotated array is: 1 2 3 4 5 6 7 8 9 0
```

```
Enter 1 to rotate the array to the left, or 2 to rotate to the right (-1 to end): 2
Right Rotation is Selection.
Your rotated array is: 0 1 2 3 4 5 6 7 8 9

Enter 1 to rotate the array to the left, or 2 to rotate to the right (-1 to end): -1
```