CSE 331 – Computer Organization Homework #2 Report

Barış Ayyıldız
1901042252

C code:

```c
#include <stdio.h>
#include <stdlib.h>

int longest[100];
int longestSize = 0;

void printArr(int *arr, int n){
  if(n == 0)
    return;
  printf("candidate sequence : [");
  for(int i=0; i<n; i++){
    printf("%d,", arr[i]);
  }
  printf("\b], size=%d\n", n);
}

void changeLongest(int *temp, int tempSize){
  for(int i=0; i<tempSize; i++){
    longest[i] = temp[i];
  }
  longestSize = tempSize;
}

void algo(int *arr, int arrSize, int *temp, int tempSize, int index){
  if(index >= arrSize){
    if(tempSize > longestSize){
      changeLongest(temp, tempSize);
    }
    printArr(temp, tempSize);
    return;
  }
  if(tempSize == 0 || *arr > temp[tempSize-1]){
    temp[tempSize] = *arr;
    algo(arr+1, arrSize, temp, tempSize+1, index+1);
  }
  algo(arr+1, arrSize, temp, tempSize, index+1);
}

int main(){
  int arr[] = {3,10,7,9,4,11};
  int temp[6];
  algo(arr, 6, temp, 0, 0);
  printf("----------------------------------------\n");
  printArr(longest, longestSize);

  return 0;
}
```

Pseudo code:

```
// Global Variables
int longest[];
int longestSize = 0;

function printArr(int[] arr, int n):
  if n == 0
    return
  print "candidate sequence : ["
  for(i=0; i<n; i++)
    print arr[i] + ","

function changeLongest(int[] temp, int tempSize):
  for(i=0; i<tempSize; i++)
    longest[i] = temp[i]
  longestSize = tempSize

function algo(int[] arr, int arrSize, int[] temp, int tempSize, int index):
  if(index >= arrSize):
    if(tempSize > longestSize):
      changeLongest(temp, tempSize)
    printArr(temp, tempSize)
    return
  if(tempSize == 0 or arr[index] > temp[tempSize-1]):
    temp[tempSize] = arr[index];
    algo(arr+1, arrSize, temp, tempSize+1, index+1);
  algo(arr+1, arrSize, temp, tempSize, index+1);

// Driver Code
int[] arr = [3,10,7,9,4,11]
int[] temp
algo(arr, 6, temp, 0, 0)
print("--------------------")
printArr(longest, longestSize)
```

Explanation of the pseudo code:

My algorithm, the algo function, works recursively. It takes an array, temporary array, size of the array, size of the temporary array and the current index as inputs.

If the current index is greater than the size of the array, it just checks if the temporary array's size is bigger than longest array's size. If that's the case, it changes longest array to temporary array and longest array's size to temporary array's size. And then prints the candidate sequence.

If the current index is not greater than the size of the array, it compares the value at the array's index with the temporary array's last element. If the first value is bigger than the temporary array's last element it appends that value to the end of the temporary array

Missing Parts

I could not implement the file operations. If a user wants to test the code, he/she needs to change the array and array size on the assembly file

Improvements

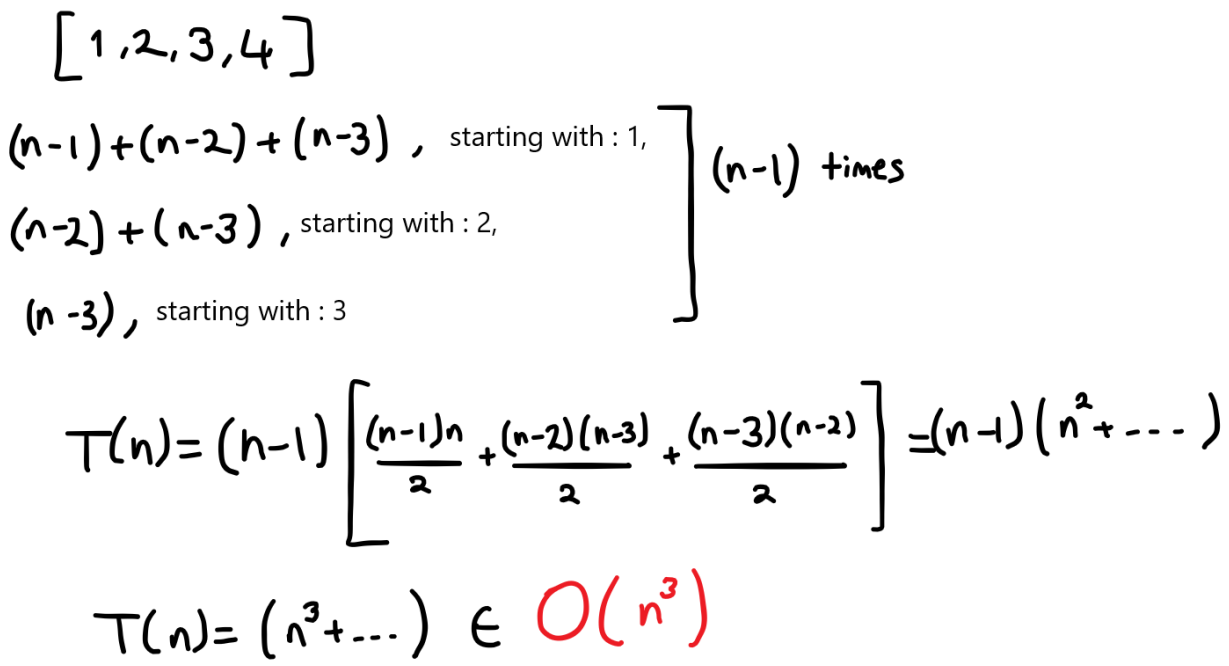
The assembly code is able to print all the candidate sequences of the increasing subsets.

Space Complexity

The algorithm requires a ‘n’ element sized temporary and longest arrays.
T(n) = 2*n ∈ O(n)

Time Complexity

The worst case is when the array is already increasing. Because it has to find all the combinations



Test Cases

#1

```
arr: .word 3, 10, 7, 9, 4, 11
arraySize: .word 6
```

```
candidate sequence : [3,10,11,] ,size = 3
candidate sequence : [3,10,] ,size = 2
candidate sequence : [3,7,9,11,] ,size = 4
candidate sequence : [3,7,9,] ,size = 3
candidate sequence : [3,7,11,] ,size = 3
candidate sequence : [3,7,] ,size = 2
candidate sequence : [3,9,11,] ,size = 3
candidate sequence : [3,9,] ,size = 2
candidate sequence : [3,4,11,] ,size = 3
candidate sequence : [3,4,] ,size = 2
candidate sequence : [3,11,] ,size = 2
candidate sequence : [3,] ,size = 1
candidate sequence : [10,11,] ,size = 2
candidate sequence : [10,] ,size = 1
candidate sequence : [7,9,11,] ,size = 3
candidate sequence : [7,9,] ,size = 2
candidate sequence : [7,11,] ,size = 2
candidate sequence : [7,] ,size = 1
candidate sequence : [9,11,] ,size = 2
candidate sequence : [9,] ,size = 1
candidate sequence : [4,11,] ,size = 2
candidate sequence : [4,] ,size = 1
candidate sequence : [11,] ,size = 1
candidate sequence : [] ,size = 0
-------------------------
candidate sequence : [3,7,9,11,] ,size = 4

-- program is finished running --
```

#2

```
arr: .word 5, 4, 3, 2, 10
arraySize: .word 5
```

```
candidate sequence : [5,10,] ,size = 2
candidate sequence : [5,] ,size = 1
candidate sequence : [4,10,] ,size = 2
candidate sequence : [4,] ,size = 1
candidate sequence : [3,10,] ,size = 2
candidate sequence : [3,] ,size = 1
candidate sequence : [2,10,] ,size = 2
candidate sequence : [2,] ,size = 1
candidate sequence : [10,] ,size = 1
candidate sequence : [] ,size = 0
-------------------------
candidate sequence : [5,10,] ,size = 2

-- program is finished running --
```

#3

```
arr: .word 6, 12, 8, 21, 45, 13
arraySize: .word 6
```

```
candidate sequence : [6,12,21,45,] ,size = 4
candidate sequence : [6,12,21,] ,size = 3
candidate sequence : [6,12,45,] ,size = 3
candidate sequence : [6,12,13,] ,size = 3
candidate sequence : [6,12,] ,size = 2
candidate sequence : [6,8,21,45,] ,size = 4
candidate sequence : [6,8,21,] ,size = 3
candidate sequence : [6,8,45,] ,size = 3
candidate sequence : [6,8,13,] ,size = 3
candidate sequence : [6,8,] ,size = 2
candidate sequence : [6,21,45,] ,size = 3
candidate sequence : [6,21,] ,size = 2
candidate sequence : [6,45,] ,size = 2
candidate sequence : [6,13,] ,size = 2
candidate sequence : [6,] ,size = 1
candidate sequence : [12,21,45,] ,size = 3
candidate sequence : [12,21,] ,size = 2
candidate sequence : [12,45,] ,size = 2
candidate sequence : [12,13,] ,size = 2
candidate sequence : [12,] ,size = 1
candidate sequence : [8,21,45,] ,size = 3
candidate sequence : [8,21,] ,size = 2
candidate sequence : [8,45,] ,size = 2
candidate sequence : [8,13,] ,size = 2
candidate sequence : [8,] ,size = 1
candidate sequence : [21,45,] ,size = 2
candidate sequence : [21,] ,size = 1
candidate sequence : [45,] ,size = 1
candidate sequence : [13,] ,size = 1
candidate sequence : [] ,size = 0
-------------------------
candidate sequence : [6,12,21,45,] ,size = 4

-- program is finished running --
```

#4

```
arr: .word 5, 4, 3, 12, 11, 10, 86, 1
arraySize: .word 8
```

```
candidate sequence : [5,12,86,] ,size = 3
candidate sequence : [5,12,] ,size = 2
candidate sequence : [5,11,86,] ,size = 3
candidate sequence : [5,11,] ,size = 2
candidate sequence : [5,10,86,] ,size = 3
candidate sequence : [5,10,] ,size = 2
candidate sequence : [5,86,] ,size = 2
candidate sequence : [5,] ,size = 1
candidate sequence : [4,12,86,] ,size = 3
candidate sequence : [4,12,] ,size = 2
candidate sequence : [4,11,86,] ,size = 3
candidate sequence : [4,11,] ,size = 2
candidate sequence : [4,10,86,] ,size = 3
candidate sequence : [4,10,] ,size = 2
candidate sequence : [4,86,] ,size = 2
candidate sequence : [4,] ,size = 1
candidate sequence : [3,12,86,] ,size = 3
candidate sequence : [3,12,] ,size = 2
candidate sequence : [3,11,86,] ,size = 3
candidate sequence : [3,11,] ,size = 2
candidate sequence : [3,10,86,] ,size = 3
candidate sequence : [3,10,] ,size = 2
candidate sequence : [3,86,] ,size = 2
candidate sequence : [3,] ,size = 1
candidate sequence : [12,86,] ,size = 2
candidate sequence : [12,] ,size = 1
candidate sequence : [11,86,] ,size = 2
candidate sequence : [11,] ,size = 1
candidate sequence : [10,86,] ,size = 2
candidate sequence : [10,] ,size = 1
candidate sequence : [86,] ,size = 1
candidate sequence : [1,] ,size = 1
candidate sequence : [] ,size = 0
-------------------------
candidate sequence : [5,12,86,] ,size = 3

-- program is finished running --
```

#5

```
arr: .word 1, 2, 3, 4, 5
arraySize: .word 5
```

```
candidate sequence : [1,2,3,4,5,] ,size = 5
candidate sequence : [1,2,3,4,] ,size = 4
candidate sequence : [1,2,3,5,] ,size = 4
candidate sequence : [1,2,3,] ,size = 3
candidate sequence : [1,2,4,5,] ,size = 4
candidate sequence : [1,2,4,] ,size = 3
candidate sequence : [1,2,5,] ,size = 3
candidate sequence : [1,2,] ,size = 2
candidate sequence : [1,3,4,5,] ,size = 4
candidate sequence : [1,3,4,] ,size = 3
candidate sequence : [1,3,5,] ,size = 3
candidate sequence : [1,3,] ,size = 2
candidate sequence : [1,4,5,] ,size = 3
candidate sequence : [1,4,] ,size = 2
candidate sequence : [1,5,] ,size = 2
candidate sequence : [1,] ,size = 1
candidate sequence : [2,3,4,5,] ,size = 4
candidate sequence : [2,3,4,] ,size = 3
candidate sequence : [2,3,5,] ,size = 3
candidate sequence : [2,3,] ,size = 2
candidate sequence : [2,4,5,] ,size = 3
candidate sequence : [2,4,] ,size = 2
candidate sequence : [2,5,] ,size = 2
candidate sequence : [2,] ,size = 1
candidate sequence : [3,4,5,] ,size = 3
candidate sequence : [3,4,] ,size = 2
candidate sequence : [3,5,] ,size = 2
candidate sequence : [3,] ,size = 1
candidate sequence : [4,5,] ,size = 2
candidate sequence : [4,] ,size = 1
candidate sequence : [5,] ,size = 1
candidate sequence : [] ,size = 0
-------------------------
candidate sequence : [1,2,3,4,5,] ,size = 5

-- program is finished running --
```

#6

```
arr: .word 45, 32, 67, 41, 675, 12, 78, 8, 2, 4
arraySize: .word 10
```

```
candidate sequence : [45,67,675,] ,size = 3
candidate sequence : [45,67,78,] ,size = 3
candidate sequence : [45,67,] ,size = 2
candidate sequence : [45,675,] ,size = 2
candidate sequence : [45,78,] ,size = 2
candidate sequence : [45,] ,size = 1
candidate sequence : [32,67,675,] ,size = 3
candidate sequence : [32,67,78,] ,size = 3
candidate sequence : [32,67,] ,size = 2
candidate sequence : [32,41,675,] ,size = 3
candidate sequence : [32,41,78,] ,size = 3
candidate sequence : [32,41,] ,size = 2
candidate sequence : [32,675,] ,size = 2
candidate sequence : [32,78,] ,size = 2
candidate sequence : [32,] ,size = 1
candidate sequence : [67,675,] ,size = 2
candidate sequence : [67,78,] ,size = 2
candidate sequence : [67,] ,size = 1
candidate sequence : [41,675,] ,size = 2
candidate sequence : [41,78,] ,size = 2
candidate sequence : [41,] ,size = 1
candidate sequence : [675,] ,size = 1
candidate sequence : [12,78,] ,size = 2
candidate sequence : [12,] ,size = 1
candidate sequence : [78,] ,size = 1
candidate sequence : [8,] ,size = 1
candidate sequence : [2,4,] ,size = 2
candidate sequence : [2,] ,size = 1
candidate sequence : [4,] ,size = 1
candidate sequence : [] ,size = 0
-------------------------
candidate sequence : [45,67,675,] ,size = 3

-- program is finished running --
```