

CSE 331 COMPUTER ORGANIZATION

#HW2

Atakan ALTIN

1801042668

Pseudocode:

for : int i=0 to size

for: int j=i+1 to size

subSize=0 // Each Subsequent we need to set size 0.

subArr[subSize++]=arr[i] // If the size is 0 next element add unconditional.

if arr[j] >= subArr[subSize-1] // Checking if the next element is bigger than subsequent's last element

subArr[subSize++]=arr[j] // adding new element

for: int k=j+1 to size

if arr[k]>=subArr[subSize-1] //Checking if the next element is bigger than subsequent's last element

subArr[subSize++]=arr[k] // adding new element

printSubSequences(subArr,subSize)

if subSize>=max //If the new subsequent's size bigger than last then it's the current longest increasing sequence

max=subSize

for: int m=0 to subSize

resultArr[m]=subArr[m]

print("Longest Subsequence Is :")

printResult(resultArr,max)

Time complexity:

```
16- for (int i = 0; i < size; i++) {
17-     for (j=i+1; j<size; j++){
18-         subSize=0;
19-         subArr[subSize++]=arr[i];
20-         if (arr[j]>=subArr[subSize-1]){
21-             subArr[subSize++]=arr[j];
22-             for (int k=j+1; k<size; k++){
23-                 if(arr[k]>=subArr[subSize-1]){
24-                     subArr[subSize++]=arr[k];
25-                 }
26-             }
27-             printArr(subArr,subSize);
28-             printf("Size: %d\n",subSize);
29-             if(subSize>=max){
30-                 max=subSize;
31-                 for(int k=0; k<subSize; k++){
32-                     resultArr[k]=subArr[k];
33-                 }
34-             }
35-         }
36-     }
37- }
38- }
39- }
40- printf("\nMax sized array is: ");
41- printArr(resultArr,max);
```

$O(n-2)$

$O(n-2) * O(n-1) = O(n^2)$

Space Complexity:

-Since the auxillary space that is used here has a strict upper bound that is independent on the input. So, space complexity is $O(1)$.

Test Cases:

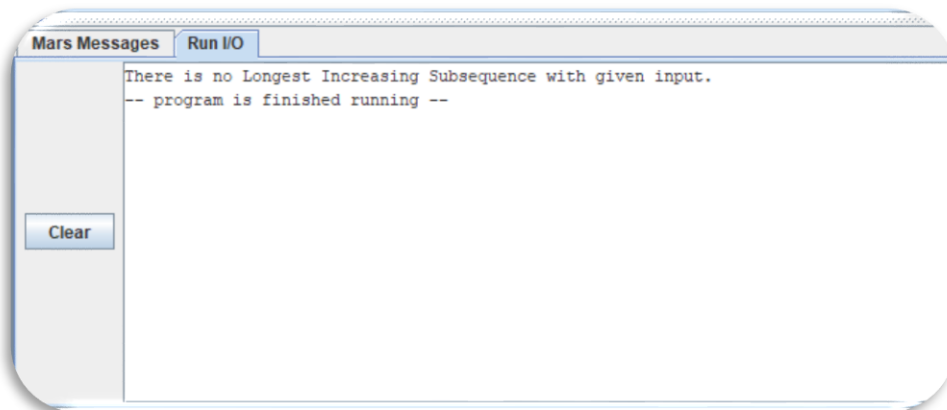
Array = {3,10,7,9,4,11}

```
-- program is finished running --  
  
3,10,11,Size: 3  
3,7,9,11,Size: 4  
3,9,11,Size: 3  
3,4,11,Size: 3  
3,11,Size: 2  
10,11,Size: 2  
7,9,11,Size: 3  
7,11,Size: 2  
9,11,Size: 2  
4,11,Size: 2  
  
Longest Increasing Subsequence Is: 3,7,9,11,Size: 4  
  
-- program is finished running --
```

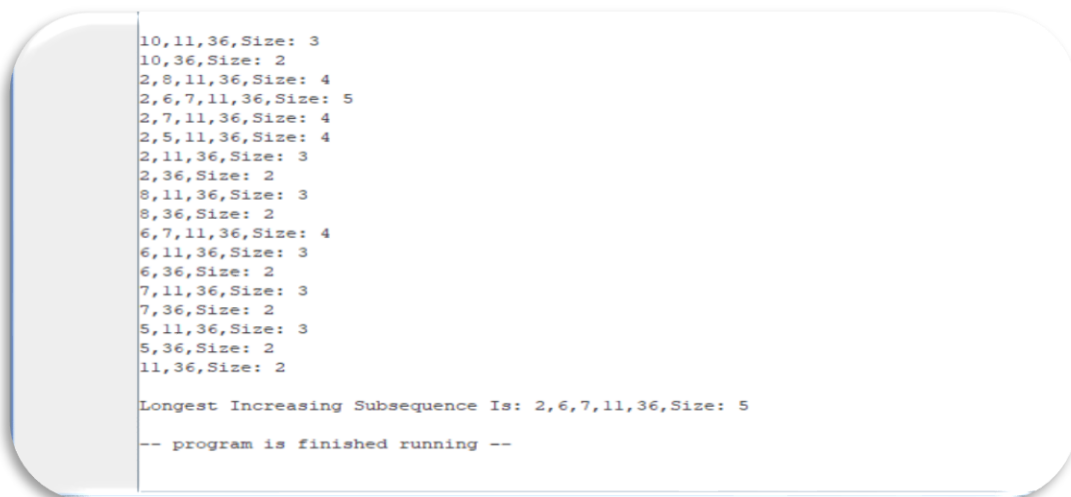
Array={5,7,1,4,9,22}

```
5,7,9,22,Size: 4  
5,9,22,Size: 3  
5,22,Size: 2  
7,9,22,Size: 3  
7,22,Size: 2  
1,4,9,22,Size: 4  
1,9,22,Size: 3  
1,22,Size: 2  
4,9,22,Size: 3  
4,22,Size: 2  
9,22,Size: 2  
  
Longest Increasing Subsequence Is: 5,7,9,22,Size: 4  
  
-- program is finished running --
```

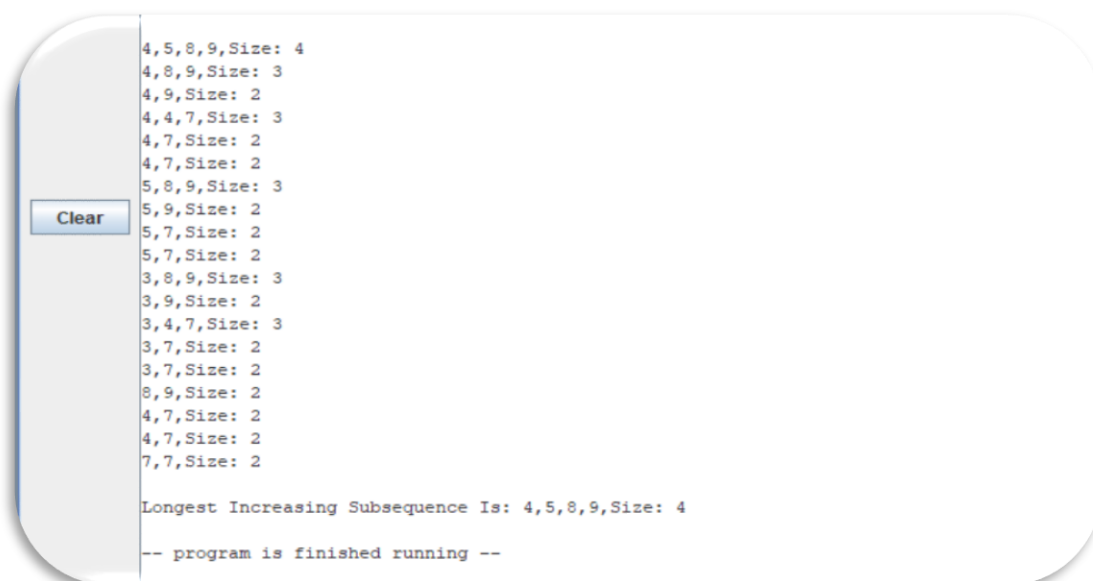
Array={9,8,7,5,4,3,2,1}



Array={10,2,8,6,7,5,11,36}



Array={4,5,3,8,9,4,7,7}



Array={163,241,55,1,21,9,3,83}

```
163,241,Size: 2
55,83,Size: 2
1,21,83,Size: 3
1,9,83,Size: 3
1,3,83,Size: 3
1,83,Size: 2
21,83,Size: 2
9,83,Size: 2
3,83,Size: 2

Longest Increasing Subsequence Is: 1,21,83,Size: 3

-- program is finished running --
```

- File reading & writing are missing.
- Inner results are provided.
- Commentation of code provided in pseudocode
- I have implemented the input array manually. All test cases performed manually. Array and array size can be changed on the top of the asm file.

```
.data
arr: .word 163,241,55,1,21,9,3,83
tempArr: .space 400
resultArr: .space 400
newline: .asciiz "\n"
comma: .asciiz ","
size : .asciiz "Size: "
result: .asciiz "Longest Increasing Subsequence Is: "
specialCaseText: .asciiz "There is no Longest Increasing Subsequence with given input"

.text

.globl main
main:
    li $s0,32 # size of array
```