

# GEBZE TECHNICAL UNIVERSITY

## CSE 331/503 Computer Organization Homework # 2

Ahmet Yazıcı  
1801042639

## 1.Pseudocode

```
for(i=0;i<size;++i){
    temp_counter = 0;
    temp[temp_counter++] = arr[i];
    for(j=i+1;j<size;++j){
        for(k=j+1;k<size;k++){
            if(temp[temp_counter-1] < arr[k])
                temp[temp_counter++] = arr[k];
        }

        if(max < temp_counter){
            max = temp_counter;
            for(k=0;k<temp_counter;k++)
                output[k] = temp[k];
        }
    }
}
```

In the first loop, the algorithm iterates all the elements of the array and puts the current element in the first index of the temp array.

In the second loop, the loop starts with i+1 index because there is no need to scan for the first index again.

In the third loop, the algorithm starts with  $j+1$  index because again there is no need to scan for the first element. Then in this loop, compares with previous temp array content with the current index. If the current index larger than previous content of the temp array (increasing order), saves the content to the temp array. After the third loop, the algorithm compares the size of the temp array and the previously finding largest size (which is initially 0). If the size of temp array is larger, all the contents of the temp array store in output array. Output array contains the largest increasing sequence.

The time complexity of algorithm is  $n * (n-1) * (n-2) = O(n^3)$

The space complexity of algorithm is:  $3n$

## 2.Functions

```
# function for performs the desired algorithm
# parameters of function ---- a0 -> address of array a1 -> size of array
# return statement ----- v0 -> size of maximum length

# c form of function ----> int algorithm(int arr[], int size)
algorithmStart:

    addi $sp,$sp, -12      # make room on stack for 5 registers
    sw $ra,8($sp)          # save $ra on stack
    sw $s1,4($sp)          # save $s1 on stack
    sw $s0,0($sp)          # save $s2 on stack

    move $s0,$a0
    move $s1,$a1

    li $t0,0 # loop counter == i
    li $t1,0 # loop counter == j
    li $t2,0 # loop counter == k
    li $t3,0 # tempArr counter
    li $t4,0 # temp variable for accessing the array elements
    li $t5,0 # temp variable for accessing the array elements
    li $t6,0 # temp variable for maximum length

firstLoop:
    # for(i=0;i<size;i++)
    bge $t0,$s1,firstLoopEnd      # if (i < size)
    li $t3,0                       # tempArr counter
    sub $t4,$s0,$t0               # calculating index
    lw $t4,0($t4)                 # arr[i]
    sw $t4,tempArr($t3)           # tempArr[tempArrCounter] = arr[i]
    addi $t3,$t3,4                # tempArrCounter = 1;

middleLoop:
    # for(j=i+1;j<size;j++)
    addi $t1,$t0,4
middleLoopCondition:
    bge $t1,$s1,middleLoopEnd      # if (j < size)

innerLoop:
    # for(k=j+1;k<size;k++)
    addi $t2,$t1,4
innerLoopCondition:
    bge $t2,$s1,innerLoopEnd        # if (k < size)

    sub $t4,$s0,$t2                # calculating index
    lw $t4,0($t4)                  # arr[k]
```

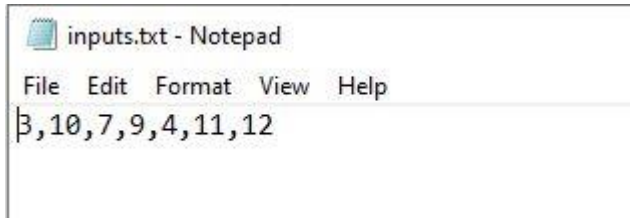
In this function, the desired algorithm performs and the largest sequence stores in outputArr.

Inputs -> Array and Size of Array

Outputs -> The size of largest sequence

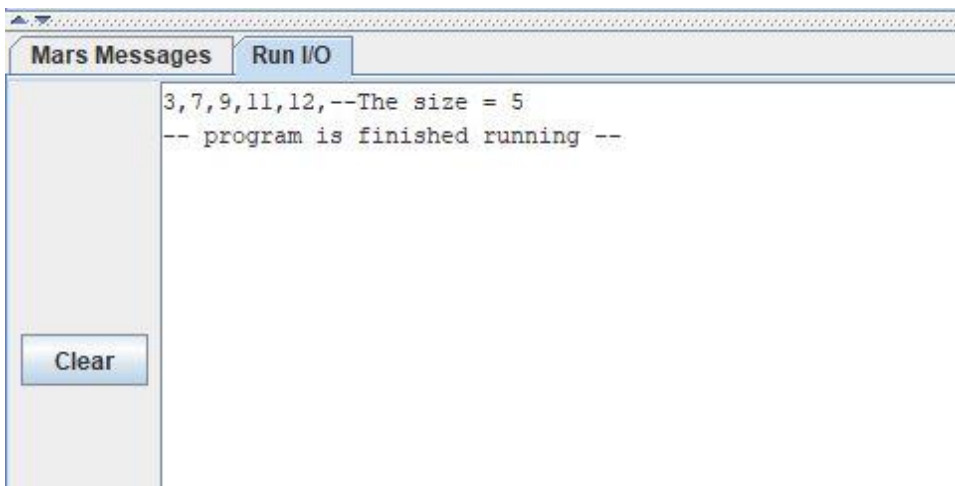
### 3. Results

For input:



```
inputs.txt - Notepad
File Edit Format View Help
3,10,7,9,4,11,12
```

Result:



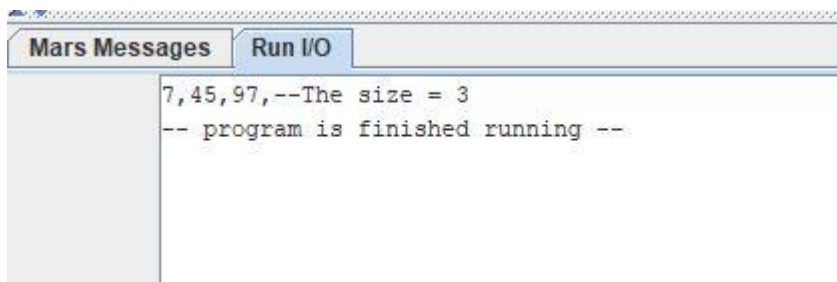
```
Mars Messages Run I/O
3,7,9,11,12,--The size = 5
-- program is finished running --
Clear
```

For input:



```
inputs.txt - Notepad
File Edit Format View Help
50,80,7,9,45,97,2
```

Output:



```
Mars Messages Run I/O
7,45,97,--The size = 3
-- program is finished running --
```

## 4. Comments

I read the file and converted them to the corresponding integers.

Then saved them into the stack. I did not print the inner results of the program. And I did not write the result into the file. Instead of this, I printed the result into the console of MARS.

Also, I tested program with one array (one line in “inputs.txt”)