Ministry of Education and Science of Ukraine

National Technical University of Ukraine

«Kyiv Polytechnic Institute. Igor Sikorsky »

Faculty of Informatics and Computer Technologies

Department of Computer Engineering

LAB № 4

from the discipline "Theory of Algorithms"

on the topic «Pyramids»

PERFORMED BY:

1st year student

group ІП-93

Dominskyi V.O.

The credit - 9311

Variant – 11

CHECKED:

Associate Professor of OT

c.t.s.,s.r.

Antoniuk А.І.

Kyiv - 2020

**TASK**

**Goal:**

determining the sequence of medians for a given input array.

**Option task:**

In this work, we need to solve the following problem of determining the sequence of medians for a given input array. Remember that the median for an array is an element that is in the middle position in the sorted array. Thus, if the number of elements in an array is odd, then the median is one and its index in the sorted array is defined as [n / 2] (where n is the size of the input array). If the number of elements in an array is even, then the median will be two and their indices are determined by the formulas [n / 2] and [n / 2] + 1. The problem is formulated as follows. Suppose that the input array A = [x1, ..., xN] is given.

Suppose that the elements of an array arrive at the input of the program sequentially: at each moment in time a new element xi is considered. It is necessary for each i (from 1 to N) to determine the median of the subarray A '= [x1, ..., xi], that is, the median for the array of elements received by the program at this point in time. It is necessary to solve this problem using the structure of the pyramid data and in such a way that each median is determined by the time O (log (i)).

**CODE**

arr\_input = []

arr\_medians = []

def heapify(arr, n, i):

largest = i

l = 2 \* i + 1

r = 2 \* i + 2

if l < n and arr[i] < arr[l]:

largest = l

if r < n and arr[largest] < arr[r]:

largest = r

if largest != i:

arr[i],arr[largest] = arr[largest],arr[i]

heapify(arr, n, largest)

def heapSort(arr):

n = len(arr)

for i in range(n, -1, -1):

heapify(arr, n, i)

for i in range(n-1, 0, -1):

arr[i], arr[0] = arr[0], arr[i]

heapify(arr, i, 0)

return arr

def mediana(arr):

# sorting array

arr = heapSort(arr)

# Calculating the mediana

if len(arr) % 2 != 0:

return arr[int((len(arr)-1) / 2)]

else:

return [arr[int((len(arr)-1) / 2)], arr[int((len(arr)-1) / 2) + 1]]

# Input

n = int(input('Input number of elements: '))

for i in range(n):

el = int(input())

arr\_input.append(el)

arr\_medians.append(mediana(arr\_input))

# Output

for element in arr\_medians:

print('Mediana is:', element)

**RESULTS OF THE PROGRAM WORK**

The input is:

Input number of elements: 5

5

7

10

12

14

Mediana is: 5

Mediana is: [5, 7]

Mediana is: 7

Mediana is: [7, 10]

Mediana is: 10

Output array:

Array of medians: Mediana is: 5

Mediana is: [5, 7]

Mediana is: 7

Mediana is: [7, 10]

Mediana is: 10

**CONCLUSIONS**

I got acquainted with the topic of laboratory work.

Have acquired relevant work skills.

An appropriate test program has been developed.

Familiarized with the pyramid sorting. With his help I calculated the medians of the arrays.