Technical University of Moldova
Inginerical department S.A.

Report

№: 2

APA

Subject: Divide et empera

Author: Terman Emil FAF161
Prof: M. Catruct

Chisinau 2017

1 Mergesort

Algorithm 1: Mergesort

```
function mergeSort(tab: array of int, int firstI, int lastI):
    int mid

    if (firstI < lastI):
        mid = (firstI + lastI) / 2
        mergeSort(tab, firstI, mid)
        mergeSort(tab, mid + 1, lastI)
        merge(tab, firstI, mid, lastI)

function merge(tab: array of int, int firstI, int mid, int lastI):
    tmp: array[lastI - firstI + 1] of int
    int i, j, k

    i := k := firstI
    j := mid + 1
    while (i <= mid and j <= lastI):
        if (tab[i] <= tab[j]):
            tmp[k - firstI] := tab[i]
            i++
        else:
            tmp[k - firstI] := tab[j]
            j++
        k++

    while (i <= mid):
        tmp[k - firstI] := tab[i]
        k++
        i++

    while (j <= lastI):
        tmp[k - firstI] := tab[j]
        k++
        j++

    for (i := firstI; i < lastI + 1; i++):
        tab[i] := tmp[i - firstI]
```

2 Quicksort

Algorithm 2: Quicksort

```
function quickSort(tab, firstI, lastI):
    int pi

    if (firstI < lastI):
        pi := partition(tab, firstI, lastI)
        quickSort(tab, firstI, pi)
        quickSort(tab, pi + 1, lastI)

function partition(tab, firstI, lastI):
    int x, i, j

    x := tab[firstI]
    i := firstI - 1
    j := lastI + 1

    while True:
        repeat j-- until tab[j] <= x
        repeat i++ until tab[i] >= x

        if i < j:
            swap(tab[i], tab[j])
        else:
            return j
```