Saad Saiyed | 200415258

Assignment 6

Problem 1 (4 marks). Illustrate how Quicksort, using the median-of-three method for finding a pivot, works on input of the list 20, 13, 17, 25, 18, 2, 29, 14, 8.

Ans:

Array	20	13	17	25	18	2	29	14	8
Index	0	1	2	3	4	5	6	7	8

Choose a pivot element from the array:

Take the first element, middle element and last element to determine the median of the array.

20, 18 and 8, median is 18

Arranging them -> 8, 18, 20 Still the median is 18.

Result:

Array	8	13	17	25	**18**	2	29	**14**	20
Index	0	1	2	3	**4**	5	6	**7**	8

Swap pivot point and second last element:

Array	8	13	17	25	**14**	2	29	**18**	20
Index	0	1	2	3	**4**	5	6	**7**	8

Performing above action (kind of) by starting while loop with two counter i and j:

Array	8	13	17	25	18	2	29	14	20
Index	0	**i = 1**	2	3	4	**j = 5**	6	7	8

While i is less than j (talking about Index), move to right skipping all the elements less than pivot point. If i's value matches bigger than pivot point it stops.

While j is greater than i (talking about Index), move to left skipping all the elements greater than pivot point. If j's value matches smaller than pivot point it stops.

When i and j are stopped, elements at their indexes swap each other's places.

When both counters cross each other, the element at i is swapped with the pivot element.

Re Storing pivot.

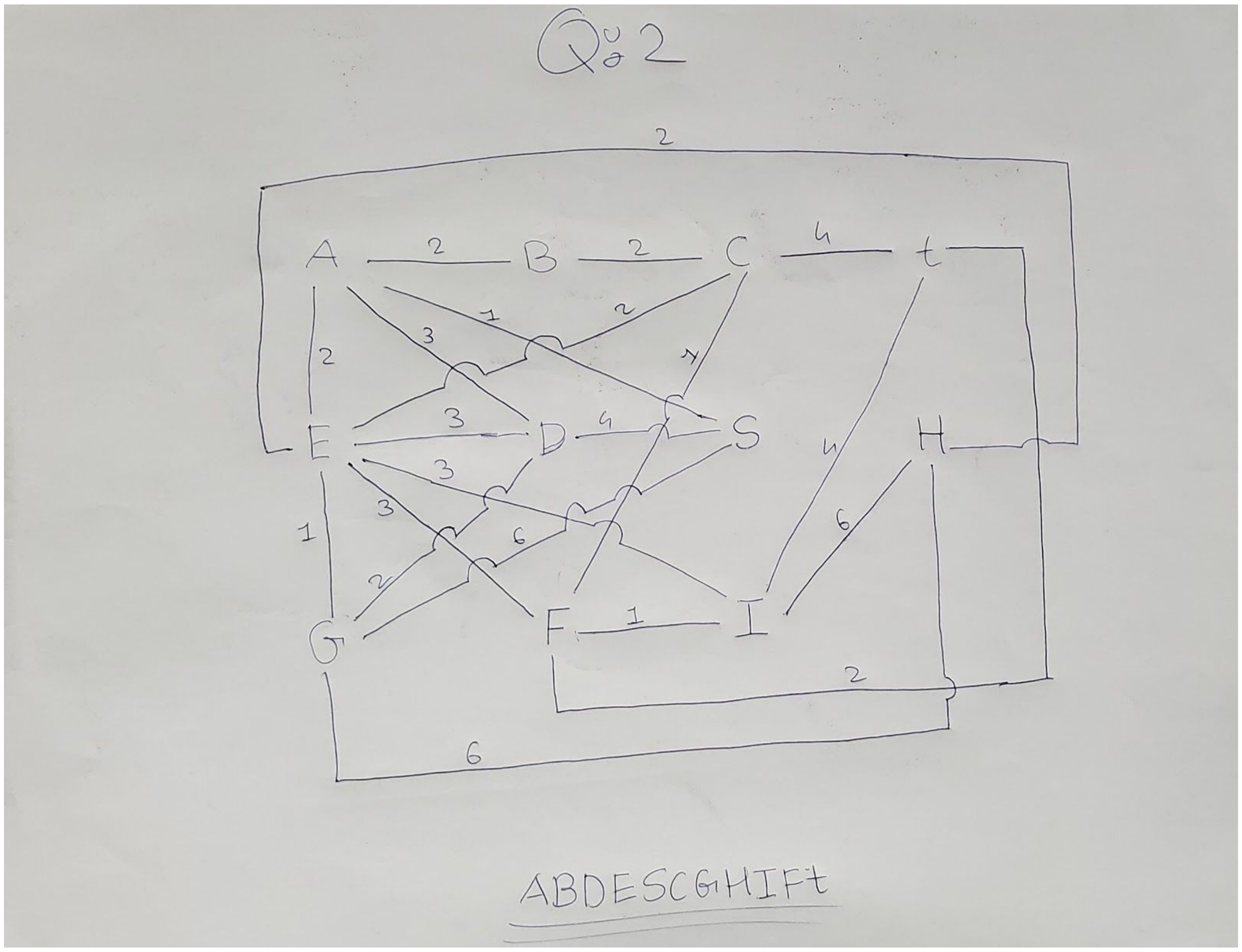
Final result:

Array	8	13	17	14	2	18	29	25	20
Index	0	1	2	3	4	5	6	7	8

Saad Saiyed | 200415258

Problem 2 (4 marks). Find a topological ordering for the graph described by the following adjacency matrix.

Ans:



Problem 3 (2+3 marks). Let G = (V;E) be a graph with nitely many vertices. Prove the following statements.

1. If G has a cycle, then G has no topological ordering.

Ans: Considering a case where G has a cycle running through vertices A,B,C as (a, b) and (b, c). Now for this cycle to have topological ordering the A must come before B and B must come before A which is impossible.

2. If G has no topological ordering, then G has a cycle.

Ans: For a cycle which does not have topological ordering, its vertices have incoming edges i.e. for each vertice we can go back through the incoming edge. So we can travel back n times where n is the number of vertices. So at some point we will cross an already visited vertex. Then it proves to be a cycle.

Problem 4 (5+2 marks).

(a) Implement QuickSort, as explained in class, with varying pivot calculation methods and varying cutoffs. You may assume that all your input lists contain only integers, are duplicate-free, and are of length at most 100.

- pivot calculation methods: pivot = first list element, pivot = second list element, pivot = element at middle position of list, pivot = median-of-three (as explained in class)
- cutoffs: no cutoff, cutoff c = 5, cutoff c = 10, cutoff c = 20 (a cutoff here means an array size such that every array of size less than or equal to c will be sorted with Insertion Sort).
- Add a counter to your implementation. Every time a comparison of two list elements is made, the counter is increased by one (also in the Insertion Sort routine).

(b) Test your implementation (all 4 * 4 = 16 variants) on three different lists of length 100 and, each time, report the total number of comparisons recorded in your counter. One of the input lists should be sorted in increasing order, one sorted in decreasing order, and one random. Show the 48 resulting values organized in a table, for better readability.

Ans:

```
#include <stdio.h>

using namespace std;

void Sort(int[] arr, int size) {
    quickSort(arr, 0, size-1);
}

void quickSort(int[] arr, int i, int j) {
    if (i < j) {
        int pivotIdx = partition(a, i, j);

        quickSort(arr, i, pivotIdx-1);
        quickSort(arr, pivotIdx+1, j);
    }
}

void swap(int[] arr, int firstPosition, int secondPosition) {
    int temp = arr[firstPosition];
    arr[firstPosition] = arr[secondPosition];
    arr[secondPosition] = temp;
}

int partition(int[] arr, int i, int j) {
    int pivot = arr[i];
    int counter = i;

    for (int k = i+1; k <= j; k++) {
        if (arr[k] < p) {
            counter++;
            swap(arr, k, counter);
        }
    }

    swap(arr, i, counter);
    return counter;
}

int main() {
```

Saad Saiyed | 200415258

```
    int arr =
["971","215","703","520","508","385","218","345","149","11","106","708","8
63","661","993","777","37","994","830","704","252","823","558","637","647"
,"392","386","691","499","188","680","685","454","312","626","435","447","
55","25","408","169","726","449","428","154","521","370","560","459","836"
,"816","903","969","920","444","970","781","158","483","597","401","298","
365","889","224","753","565","646","752","604","783","244","488","176","93
1","758","402","204","659","972","373","417","369","507","880","261","131"
,"588","260","91","514","63","247","617","410","869","455","160","907","12
2"];

    Sort(arr, 100);
    return 0;
}
```