Algorithms Assignment

Q1 –

I designed an array of structures “modules” that holds the code of the 4 modules, if they are full or part time, the admission numbers or maximum numbers, the current number of students and another structure that holds the name and surname of all the students.
With this structure you can manipulate and store this data, by entering the module you want to join/ leave and you will be registered into the array of structures. When displaying the structure, you can see all the data mentioned before, the code of the module, type, maximum/current number of students and their names/surnames. You can change the data every time you run the code, and also while running you can add more student into a module/modules and also remove them.
I designed this structure in C programming, used different functions to add/remove people from the structure and also to display the data. In the main function there is a menu in a do...while loop, in which you can chose your preferred option.
Here is the code in C:

```c
#include <stdio.h>
#include <string.h>

// size ofchars
#define SIZE 10
// number of modules
#define MOD 4
// number of students max 13+9+14+6=42
#define MAXST 42

// structure inside module structure to hold all names/surnames
struct names
{
    char firstname[SIZE];
    char surname[SIZE];
};
// structure to hold all data from each module
struct modules
{
    char code[SIZE];
    // 1 FULL -- 0 PART
    int type;
    int maximum;
    int current;
    struct names students[MAXST];
};
```

```c
// functions for each option
void join(struct modules *modsf);
void leave(struct modules *modsf);
void display(struct modules *modsf);

int main()
{
    // constant module codes, and max amount of students
    struct modules mods[MOD] = {{"DT265A", 0, 13, 0, {' ', ' '}},
                                {"DT265C", 0, 9, 0, {' ', ' '}},
                                {"DT265B", 1, 14, 0, {' ', ' '}},
                                {"DT8900", 1, 6, 0, {' ', ' '}}};

    // menu choice
    int choice;
    // end while loop
    int end = 1;

    do
    {
        // menu
        printf("1.Join module\n");
        printf("2.Leave module\n");
        printf("3.DIsplay modules\n");
        printf("4.Quit program\n");

        printf("\nEnter your choice:\n");
        scanf("%d", &choice);

        switch (choice)
        {
        case 1:
        {
            join(mods);
            break;
        }
        case 2:
        {
            leave(mods);
            break;

        } // end case2
        case 3:
        {
            display(mods);
            break;
        } // end case3
```

```
        case 4:
        {
            end = -1;
            break;
        } // end 4
        // anyother number thats not in the menu
        default:
        {
            printf("\nPlease enter an option from the menu");
            break;
        }

        } // end switch

    } while (end == 1); // end while
    return 0;
} // end main

void join(struct modules *modsf)
{
    // modulechoice
    int modch = 0;
    // enter the module of your choice
    char mchoice[SIZE];
    int find = 0;
    char firstname[SIZE];
    char surname[SIZE];
    // enter what module to join
    printf("what module do you want to join:\n");
    scanf("%s", mchoice);
    // enter firstname and surname
    printf("Please enter your name:\n");
    scanf("%s", firstname);
    printf("Please enter your surname:\n");
    scanf("%s", surname);

    for (int i = 0; i < MOD; i++)
    {

        // compare/ check that it exists
        modch = strcmp(modsf[i].code, mchoice);
        if (modch == 0)
        {
            // module has been found
            find = 1;
```

```
            // if number of students is less than the max amount
            if (modsf[i].current < modsf[i].maximum)
            {
                // add one person to the module
                // register them
                modsf[i].current++;
                strcpy(modsf[i].students[modsf[i].current - 1].firstname,
firstname);
                strcpy(modsf[i].students[modsf[i].current - 1].surname,
surname);
                printf("\nYou have been added to the module\n");

            }    // end if
            else // number of students is greater than max
            {
                printf("\nModule is full\n");
            } // end else
            break;
        } // end if

    } // end for
    // if module is not found
    if (find == 0)
    {
        printf("Please chose an existing
module\nDT265A\nDT265C\nDT265B\nDT8900\n");
    } // end if
} // end function join

void leave(struct modules *modsf)
{
    // modulechoice
    int modch = 0;
    int nameexist = 0;
    char mchoice[SIZE];
    int find = 0;
    char surname[SIZE];
    // enter what module to leave
    printf("what module do you want to leave:\n");
    scanf("%s", mchoice);
    printf("Please enter your surname:\n");
    scanf("%s", surname);
    for (int i = 0; i < MOD; i++)
    {
        // compare/ check that it exists
        modch = strcmp(modsf[i].code, mchoice);
```

```
        if (modch == 0)
        {
            // module has been found
            find = 1;
            for (int j = 0; j < modsf[i].current; j++)
            {
                nameexist = strcmp(modsf[i].students[j].surname, surname);
                if (nameexist == 0)
                {
                    // SUBSTRACT one person to the module
                    // UNregister them
                    modsf[i].current--;
                    for (int k = 0; k < modsf[i].current; k++)
                    {
                        strcpy(modsf[i].students[k].surname,
modsf[i].students[k + 1].surname);
                    }

                    printf("\nyou have exited the module\n");
                    break;
                } // end if

            } // end for

        } // end if

    } // end for
    // if module not found
    if (find == 0)
    {
        printf("Please chose an existing
module\nDT265A\nDT265C\nDT265B\nDT8900");
    } // end if
} // end function leave

void display(struct modules *mods)
{
    // display all the data
    printf("\nFULL-TIME = 1\nPART-TIME = 0\n");
    printf("\nMODULE:  TYPE:  MAX:  CURRENT:\n ");
    for (int i = 0; i < MOD; i++)
    {
        printf("\n%s     %d    %d     %d ", mods[i].code, mods[i].type,
mods[i].maximum, mods[i].current);
        for (int j = 0; j < mods[i].current; j++)
        {
            printf(" %s %s \n", mods[i].students[j].firstname,
mods[i].students[j].surname);
```

```
        } // end for inner

    } // end for outer

} // end display
```