**Reg. No: 21BCE1297 Name: Vidhi Shah Date: 03/06/22**

**DA1 & DA2**

**Q1**

**Aim:**

Write a C program to store the computer network details using a structure and pass this structure as an argument to a function that displays a list of messages identifying each pair of computers from the same locality. In the messages, the computers should be identified by their nicknames.

**Procedure:**

**Input:**

Number of computers, n

Nickname

IP Address

**Output:**

Pair of computers from the same locality identified by their nickname

**Algorithm:**

Step 1: Create a structure IPAddress with data types ipa(string), xxx(int),

yyy(int) and name(string)

Step 2: Create a split function to split the first two part of the addresses by

using ‘.’ As a delimiter and string concatenation

Step 3: Create a get\_data function that takes input for array of computer

networks and use split function for each network

Step 4: Create a compare function to compare ‘xxx’ and ‘yyy’ component of

each object in the array and print the nicknames of computers whose

first two components match.

**Code:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

//Structure Definition

typedef struct IPAddress{

    char ipa[15];

    int xxx,yyy;

    char name[30];

} IP;

//Split a part from IP Address

int \*y;

int split(char ip[15], int j, int len) {

    char temp[3] = "";

    while ((ip[j]!='.') && (j != len)) {

        strncat(temp,&ip[j],1);

        j++;

    }

    int x = strlen(temp);

    y = &x;

    int num = atoi(temp);

    return num;

}

//Get input for all computer networks

void get\_data(IP arr[], int n) {

    int i, j;

    for (i = 0; i < n; i++) {

        //Input Name & IP Address

        printf("Enter Nickname of Computer %d: ", i+1);

        scanf("%s", arr[i].name);

        printf("Enter IP Address of Computer %d: ", i+1);

        scanf("%s", arr[i].ipa);

        printf("\n");

//Call split function on the IP Address

        int len = strlen(arr[i].ipa);

        j = 0;

        arr[i].xxx = split(arr[i].ipa, j, len);

        j += \*y + 1;

        arr[i].yyy = split(arr[i].ipa, j, len);

    }

}

//Cmpare 2 IP Addresses

void compare(IP arr[], int n) {

    int i, j;

    for (i = 0; i < n - 1; i++) {

        for (j = i + 1; j < n; j++) {

            if ((arr[i].xxx == arr[j].xxx) && (arr[i].yyy == arr[j].yyy)) {

                printf("Machines %s and %s are on the same local network.\n\n", arr[i].name, arr[j].name);

            }

        }

    }

}

//Main function

int main() {

    int n, i;

    printf("\nEnter number of computers: ");

    scanf("%d", &n);

    printf("\n");

    IP cndetails[n];

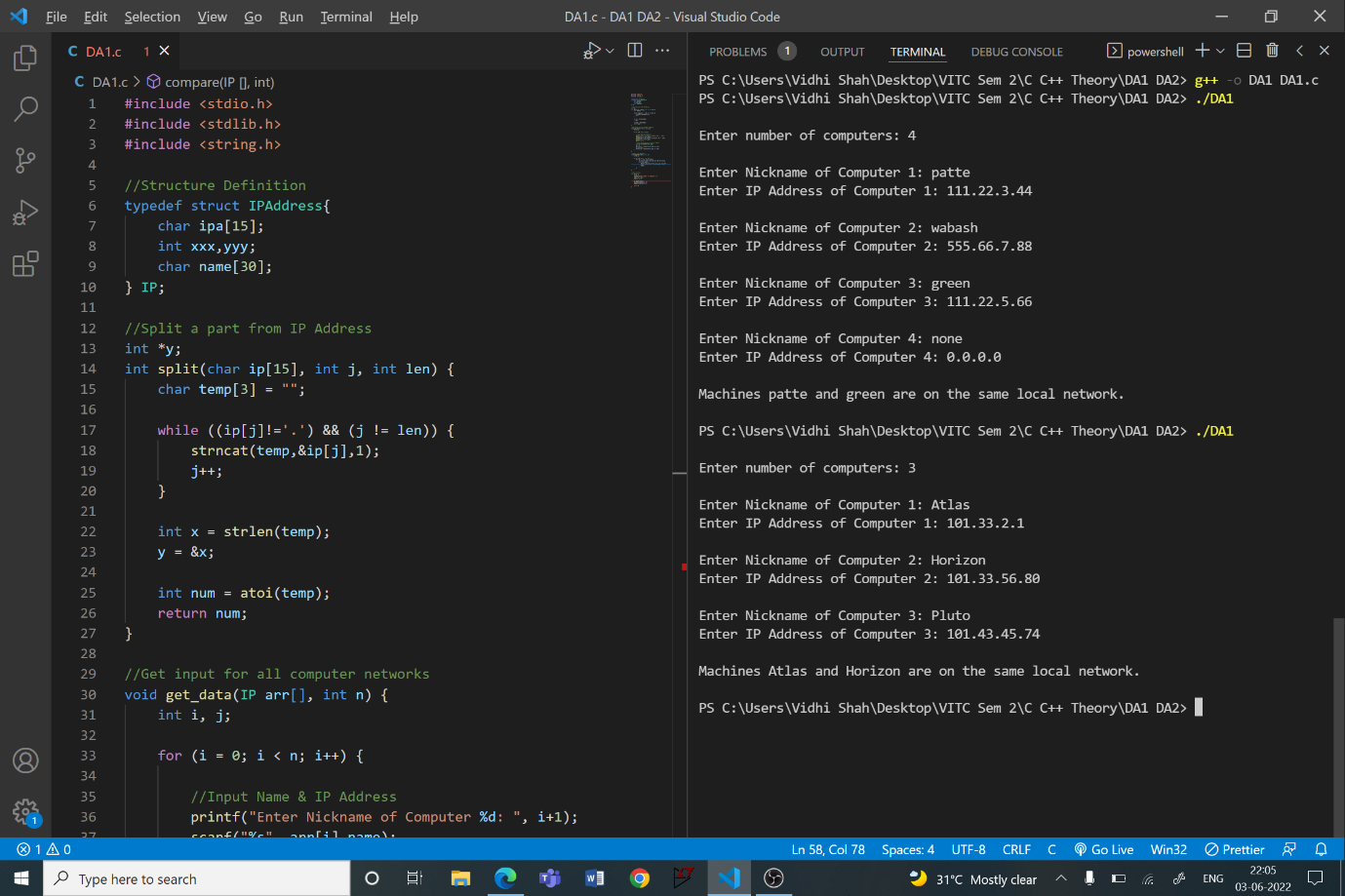
    get\_data(cndetails, n);

    compare(cndetails, n);

    return 0;

}

**Output:**



**Q2**

**Aim:**

In a library, the books are arranged vertically in the rack, one above the other. The book(s) can be added or removed only from the top and not in the middle. You have been assigned to add 10 books and remove the books until the book rack is empty. Develop the program using generic function.

**Procedure:**

**Input:**

Title of 10 books

**Output:**

Books removed from top to bottom until rack is empty

**Algorithm:**

Step 1: Create a template ‘Rack’ with class ‘Elements’

Step 2: Private data members: Array ‘stack’ of type Rack and integer type

variable ‘pos’

Step 3: Initialise ‘pos’ to 0 with constructor

Step 4: Public member functions

Step A: ‘Add’ function to add elements on top

Step B: ‘Remove’ function to remove elements from top

Step 5: Main function

Step A: Create object with string data type

Step B: For loop to add books to rack

Step C: While loop to remove books until rack is empty

**Code:**

#include <iostream>

#include <string>

using namespace std;

//Size of stack

const int SIZE = 10;

//Template for a stack of elements

template <class Rack>

class Elements {

    Rack stack[SIZE];

    int pos;

public:

    //Initialise empty stack

    Elements() {

        pos = 0;

    }

    //Function to add elements on top

    void add(Rack ele) {

        stack[pos] = ele;

        pos++;

    }

    //Function to remove elements from top

    Rack remove() {

        if (pos == 0) {

            cout<<"Rack is empty.\n\n";

            try

            {

                return 0;

            }

            catch(const exception &e)

            {

                return "";

            }

        }

        pos--;

        return stack[pos];

    }

};

//Main function

int main() {

    /\*Create a object "Books" of string type from

      template class\*/

    Elements<string> Books;

    //Add books to the rack

    cout<<"\nEnter name of the books to be added: \n";

    for(int i = 0; i < SIZE; i++) {

        string book;

        cout<<"Book "<<i+1<<": ";

        getline(cin, book);

        Books.add(book);

    }

//Remove books from the rack

    string r = "\nBooks removed in the order:";

    while (!r.empty()) {

        cout<<r<<endl;

        r = Books.remove();

    }

    return 0;

}

**Output:**

