**University of Petroleum and Energy Studies**

**SoCS**

**Semester 4, 2020-2024**

**Data Communication and Computer Networks**



**Submitted to: Mr. Dhirendra Sharma**

**Name: Anurag Singh**

**Sapid:500083382**

**Batch: B-5(H)**

**Lab-1**

**Title: Design a program to calculate baud rate**

**Code:**

#include <stdio.h>

int baud(float c, float N, int r)

{

    return c \* N \* (1 / r);

}

int main()

{

    int r;

    float N, c, S;

    printf("\n enter case factor");

    scanf("%f", &c);

    printf("\n enter ratio");

    scanf("%d", &r);

    printf("\n enter data rate ");

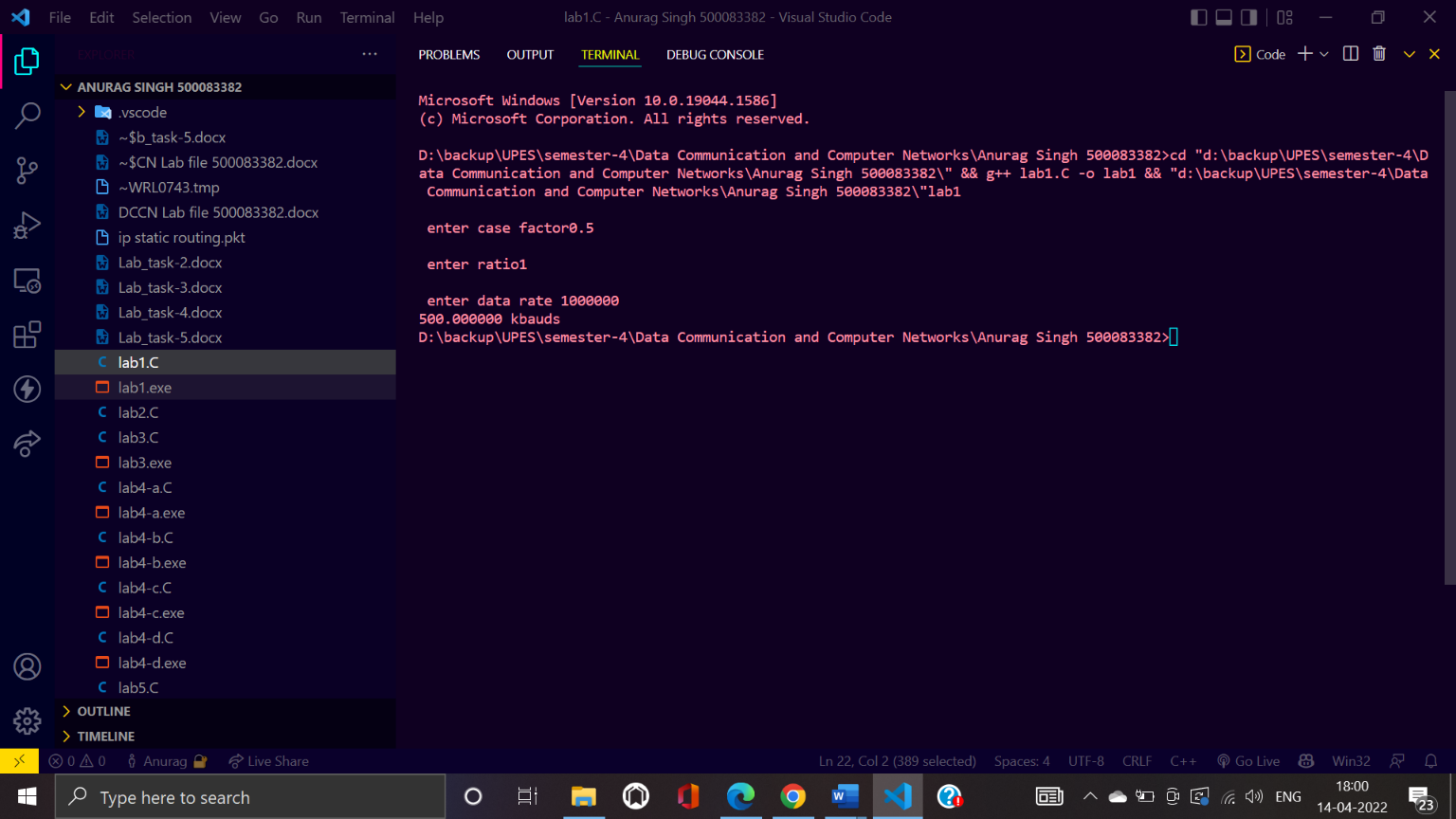
    scanf("%f", &N);

    S = baud(c, N, r);

    printf("%f kbauds", S / 1000);

    return 0;

}

**Screenshot:**

**Lab-2**

**Title: Design a program to calculate baud rate by taking values from text file and writing output into other text file.**

**Code:**

#include <stdio.h>

int main()

{

    int r;

    float N, c, S;

    FILE\* ptr;

    FILE\* ptr1;

    char ch;

    ptr = fopen("test.txt", "r");

    ptr1 = fopen("output.txt","w");

    if (NULL == ptr) {

        printf("file can't be opened \n");

    }

    while (feof(ptr)); {

        for(int i=0;i<10;i++){

        fscanf(ptr, "%f", &c);

        fscanf(ptr, "%d", &r);

        fscanf(ptr, "%f", &N);

        S=c \* N \* (1 / r);

        printf("%f kbauds", S / 1000);

        fprintf(ptr1,"%f kbauds \n",S/1000);

        }

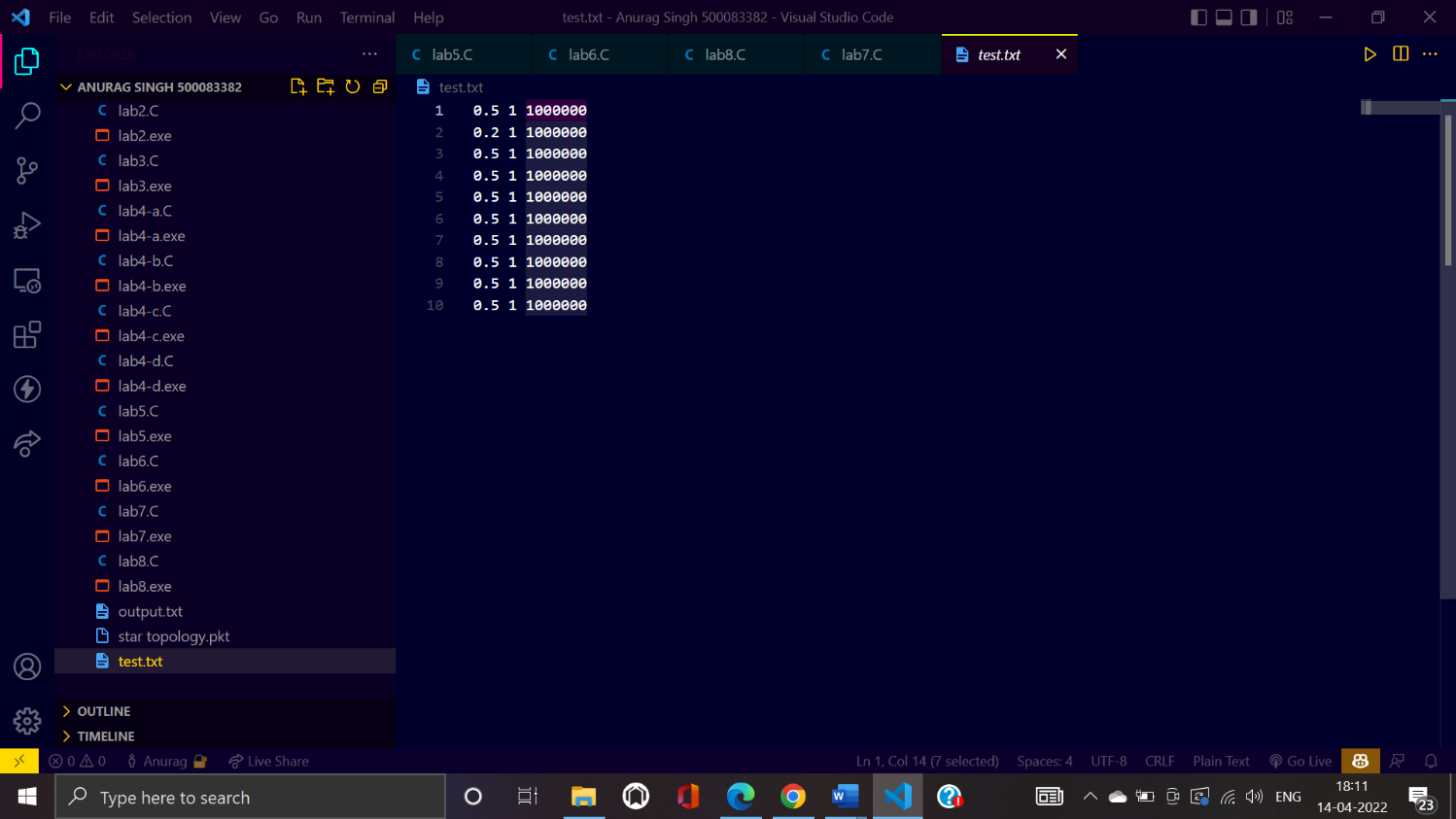
    }

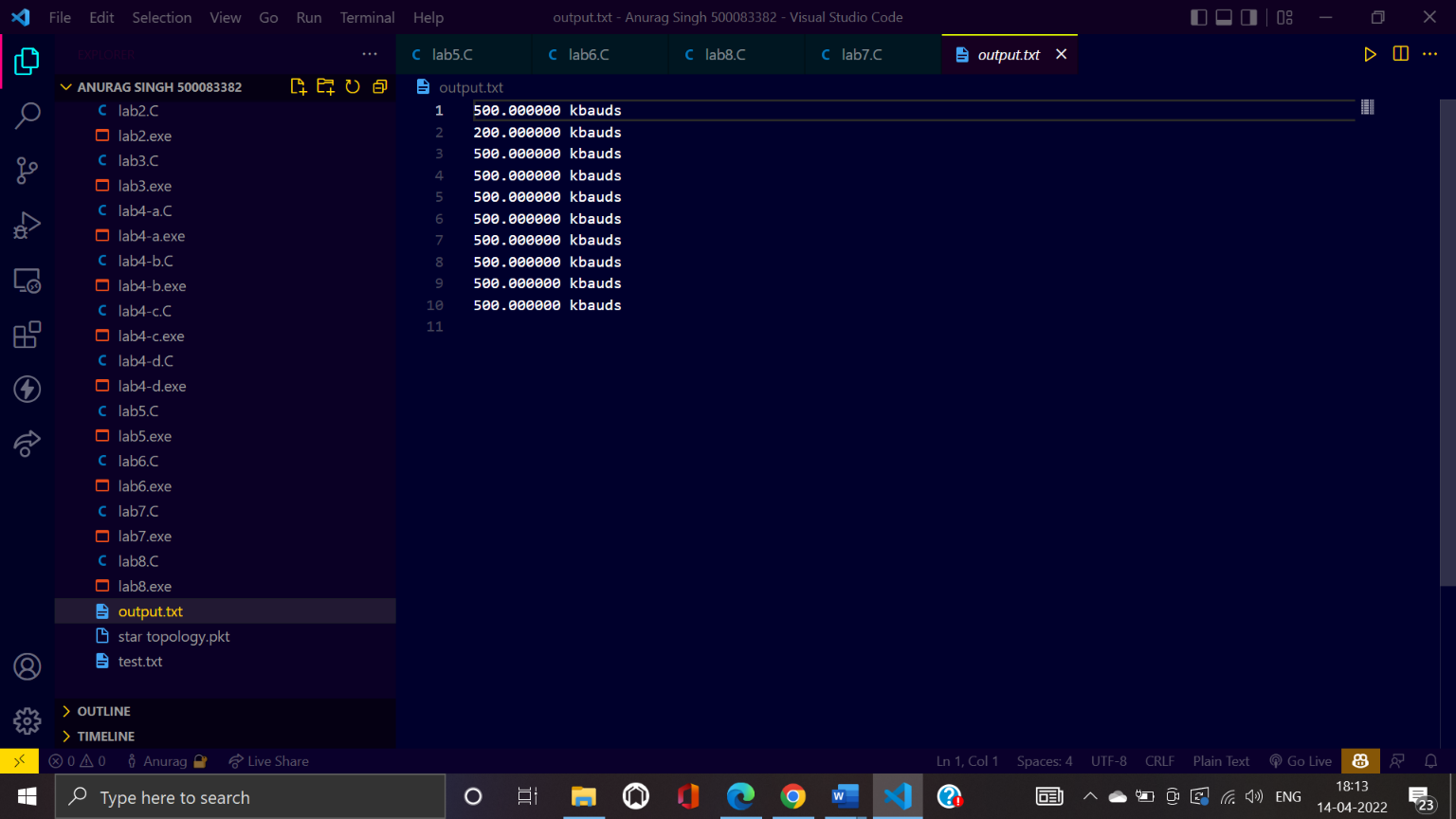
    fclose(ptr);

    return 0;

}

**Screenshot:**

**Input File:**

**Output File:**

**Lab-3**

**Title: Design a program to calculate data rate**

**Code:**

#include<stdio.h>

#include <math.h>

int main()

{

    int length;

    double levels;

    int arr[] = {1, 1, 0, 0, 0, 0, 1, 0};

    length = sizeof(arr) / sizeof(int);

    printf("So, array length is:%d\n", length);

    int result = pow(2, length); // 2^levels

    printf("levels are  %d\n", result);

    double Log = log2(result);

    printf("the log2 value is %lf\n", Log);

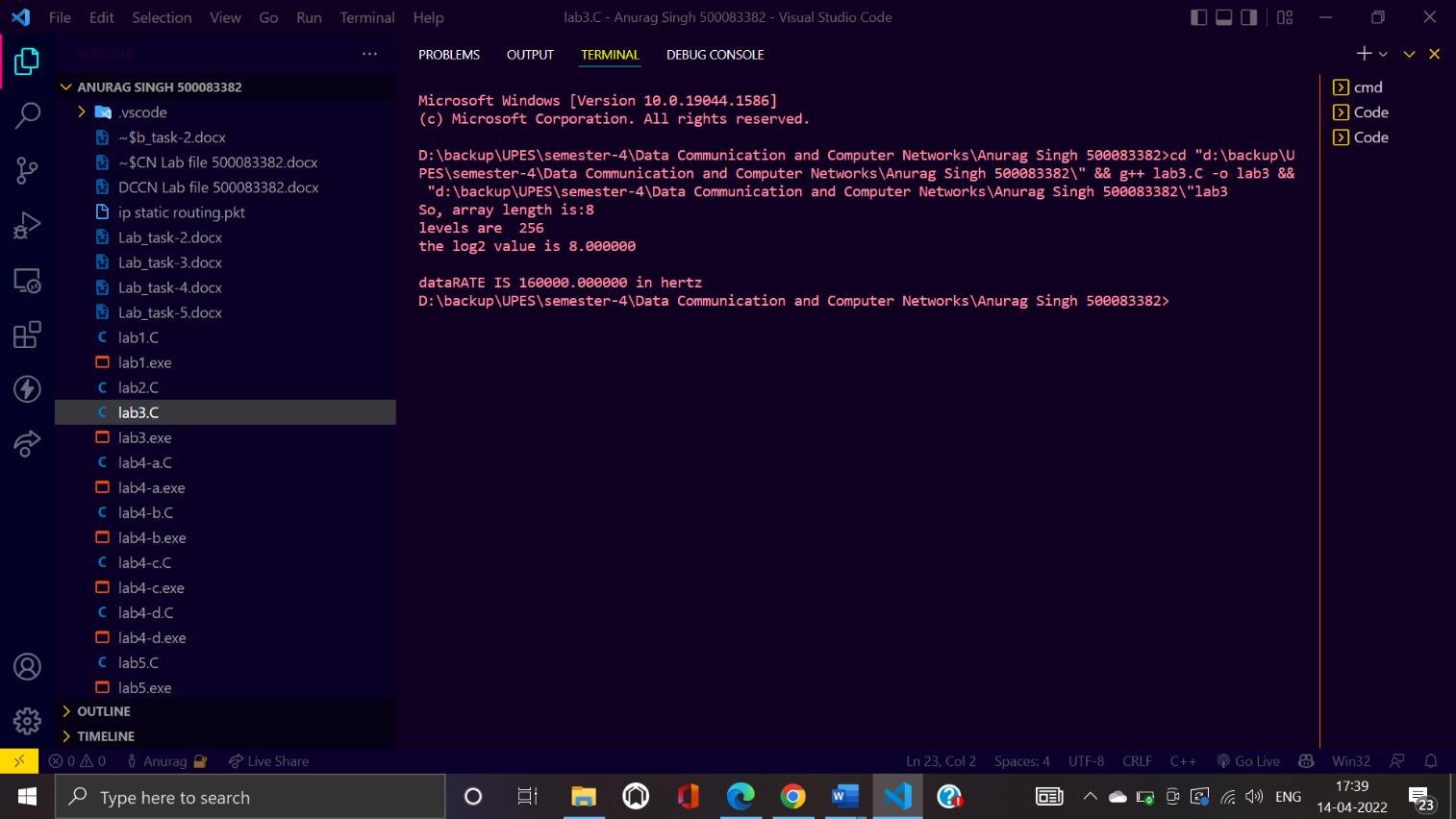
    double h = 10000;

    double datarate = 2 \* h \* Log;

    printf("\ndataRATE IS %lf in hertz", datarate);

    return 0;

}

**Screenshot:**

**Lab-4**

**Title: Design program using array and pointer**

1. Define two array and store and traverse the array

**Code:**

#include <stdio.h>

#include<malloc.h>

int main()

{

    // Define two pointer array to store the input data

    int \*arr1 = NULL;

    int \*arr2 = NULL;

    // Take the input from the user

    int length1, length2;

    printf("\nEnter the length of the first array: ");

    scanf("%d", &length1);

    printf("\nEnter the length of the second array: ");

    scanf("%d", &length2);

    // Allocate memory for the array

    arr1 = (int \*)malloc(sizeof(int) \* length1);

    arr2 = (int \*)malloc(sizeof(int) \* length2);

    // Take the input from the user

    printf("\nEnter the elements of the first array: ");

    for (int i = 0; i < length1; i++)

    {

        scanf("%d", &arr1[i]);

    }

    printf("\nEnter the elements of the second array: ");

    for (int i = 0; i < length2; i++)

    {

        scanf("%d", &arr2[i]);

    }

    // Print the input data

    printf("\nThe first array is: ");

    for (int i = 0; i < length1; i++)

    {

        printf("%d ", arr1[i]);

    }

    printf("\nThe second array is: ");

    for (int i = 0; i < length2; i++)

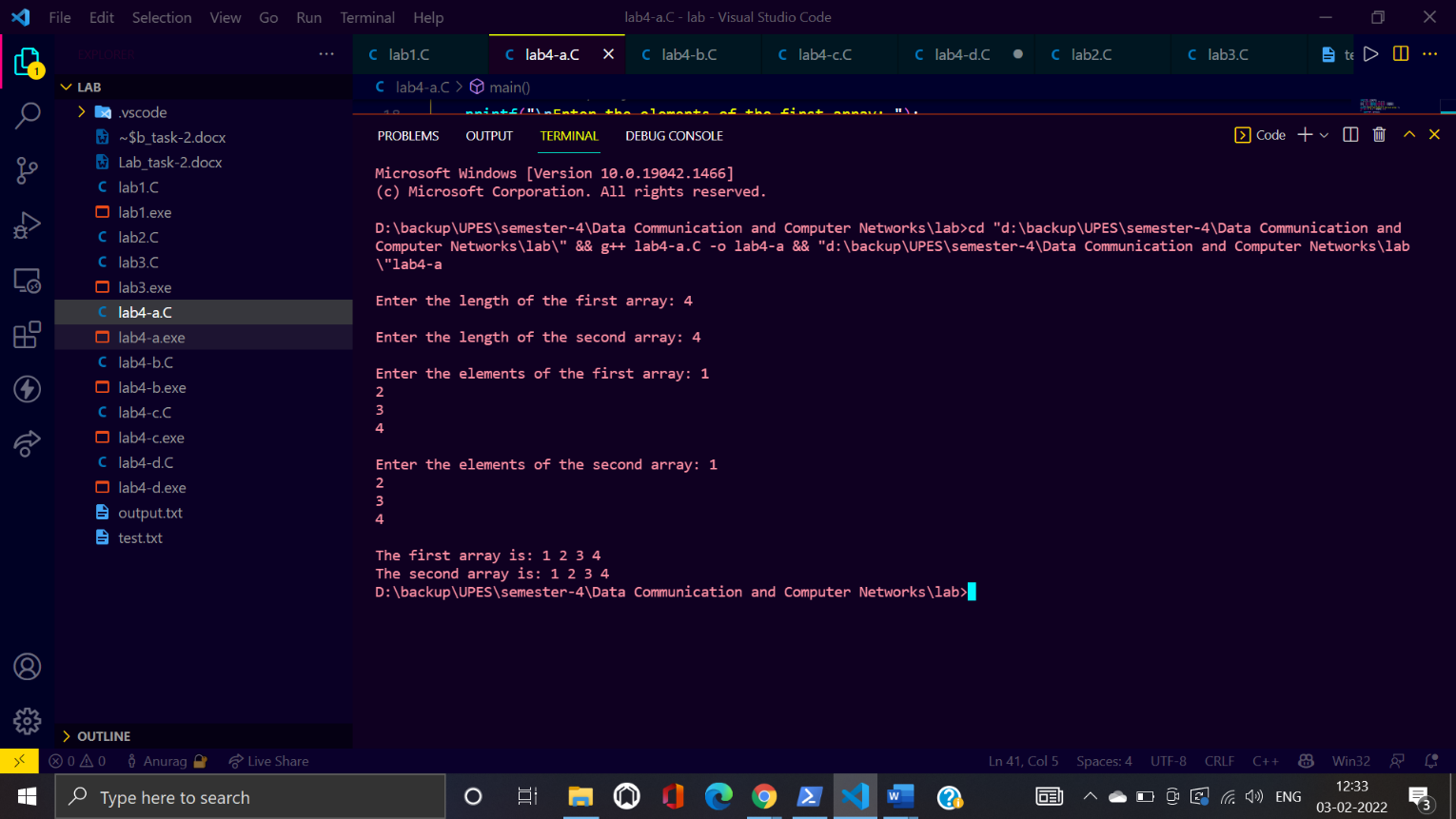
    {

        printf("%d ", arr2[i]);

    }

    return 0;

}

**Screenshot:**

1. Find the element inside the array

**Code:**

#include<stdio.h>

#include<malloc.h>

int main()

{

    // find element in pointer array

    int \*arr1 = NULL;

    // Take the input from the user

    int length;

    printf("\nEnter the length of the first array: ");

    scanf("%d", &length);

    // Allocate memory for the array

    arr1 = (int \*)malloc(sizeof(int) \* length);

    // Take the input from the user

    printf("\nEnter the elements of the first array: ");

    for (int i = 0; i < length; i++)

    {

        scanf("%d", &arr1[i]);

    }

    // search element

    int element;

    printf("\nEnter the element to be searched: ");

    scanf("%d", &element);

    int flag = 0;

    for (int i = 0; i < length; i++)

    {

        if (arr1[i] == element)

        {

            printf("\nElement found at index %d", i);

            flag = 1;

            break;

        }

    }

    if (flag == 0)

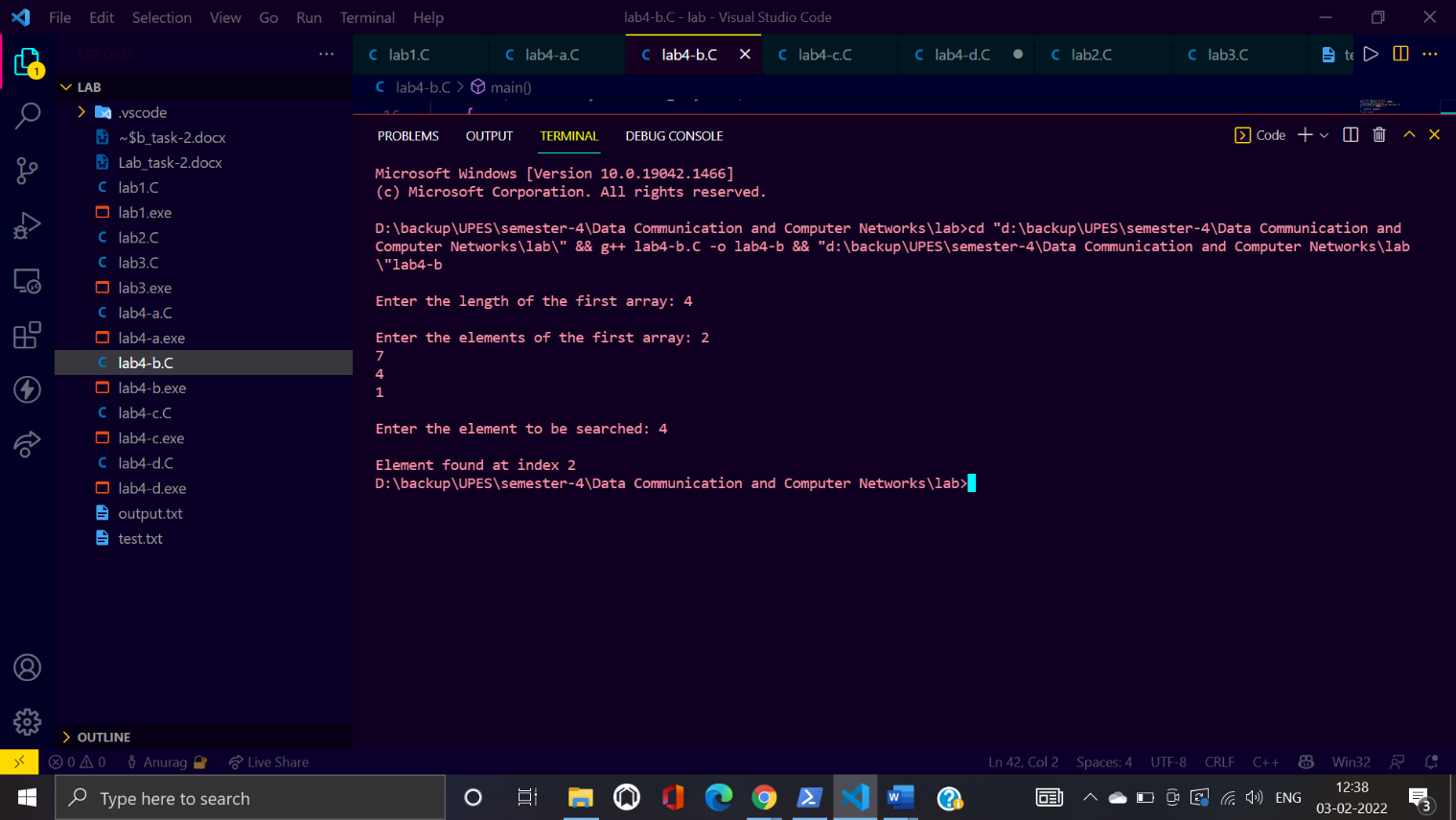
    {

        printf("\nElement not found");

    }

    return 0;

}

**Screenshot:**

1. Program for reading and writing string

**Code:**

#include<stdio.h>

int main()

{

    // string writing and reading

    char str[100];

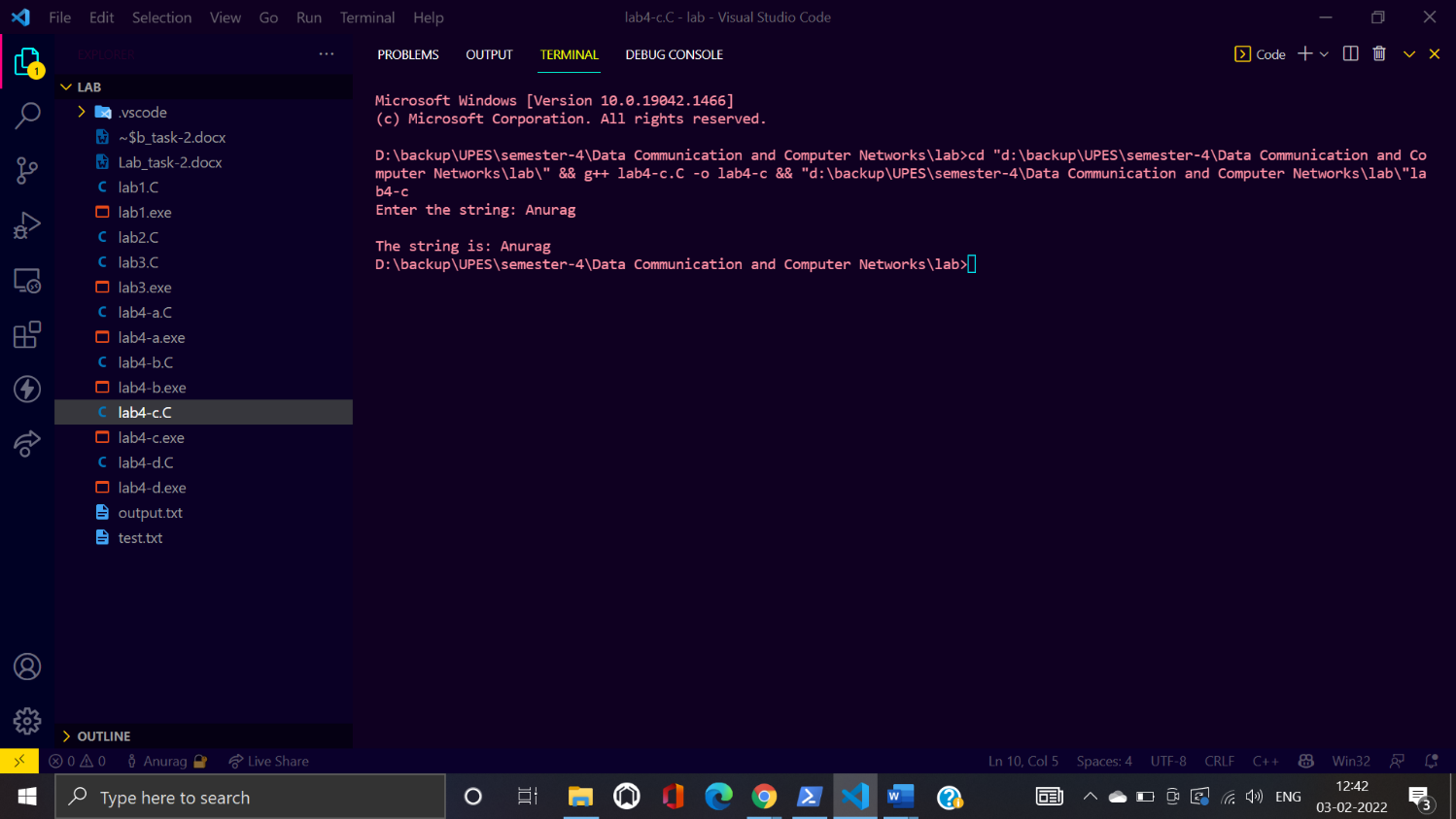
    printf("Enter the string: ");

    scanf("%s", str);

    printf("\nThe string is: %s", str);

    return 0;

}

**Screenshot:**

1. Add two string

**Code:**

#include<stdio.h>

#include<string.h>

int main()

{

    // add two  strings

    char str1[100];

    char str2[100];

    printf("Enter the first string: ");

    scanf("%s", str1);

    printf("Enter the second string: ");

    scanf("%s", str2);

    printf("\nThe first string is: %s", str1);

    printf("\nThe second string is: %s", str2);

    int length1 = strlen(str1);

    int length2 = strlen(str2);

    char str3[100];

    int i = 0;

    for (i = 0; i < length1; i++)

    {

        str3[i] = str1[i];

    }

    for (int j = 0; j < length2; j++)

    {

        str3[i] = str2[j];

        i++;

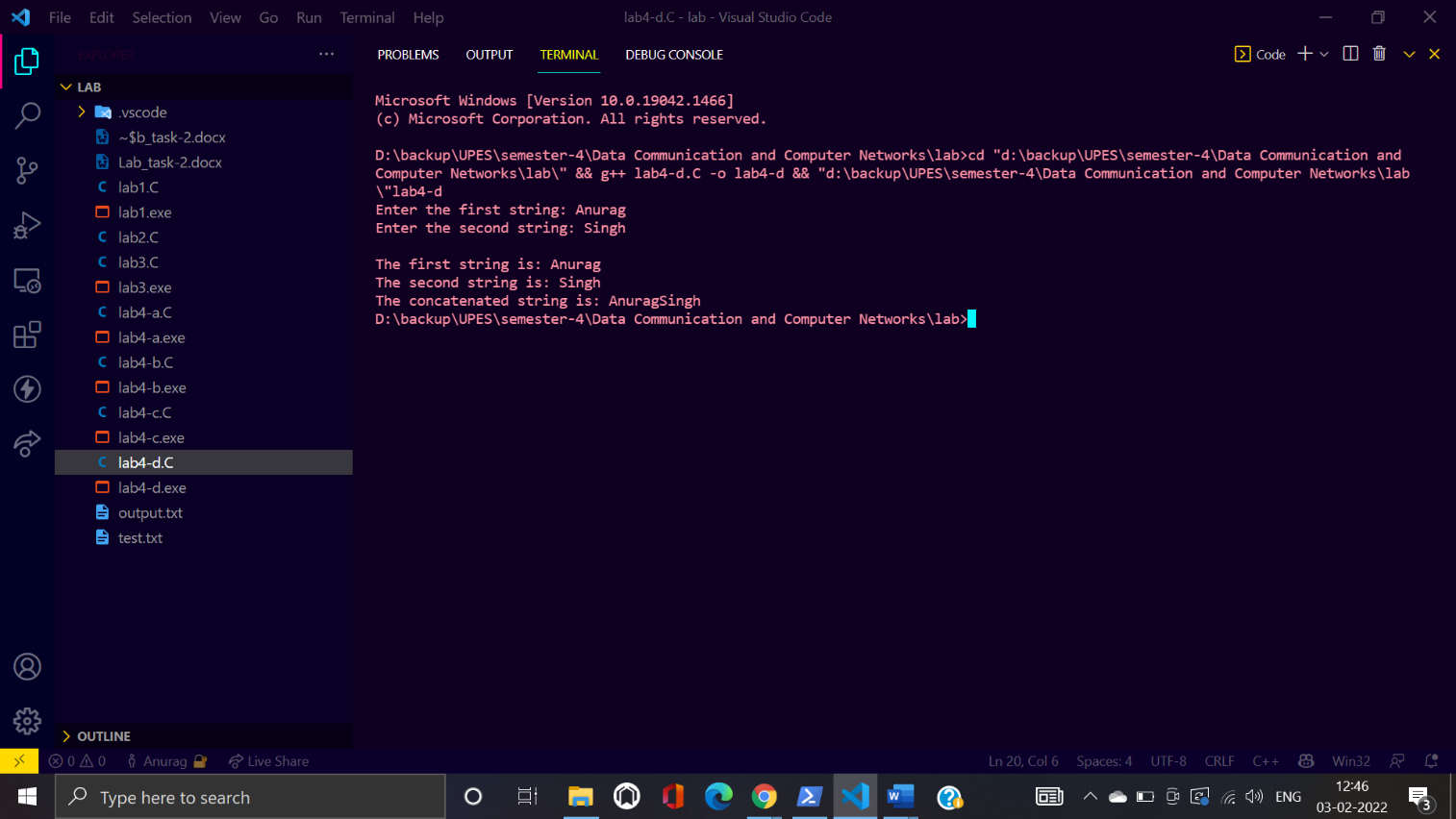
    }

    str3[i] = '\0';

    printf("\nThe concatenated string is: %s", str3);

    return 0;

}

**Screenshot:**

**Lab-5**

**Task: Write a C program to implement bit stuffing and de-stuffing**

**Code:**

#include <stdio.h>

#include <stdlib.h>

#define MAXSIZE 100

int main()

{

    char \*ptr1, \*ptr2;

    char temp;

    char input[MAXSIZE];

    char stuff[MAXSIZE];

    char destuff[MAXSIZE];

    int c = 0;

    printf("enter the input character string (0's & 1's only):\n");

    scanf("%s", input);

    ptr1 = input;

    ptr2 = stuff;

    while (\*ptr1 != '\0')

    {

        if (\*ptr1 == '0')

        {

           \*ptr2 = \*ptr1;

            ptr2++;

            ptr1++;

        }

        else

        {

            while (\*ptr1 == '1' && c != 5)

            {

                c++;

                \*ptr2 = \*ptr1;

                ptr2++;

                ptr1++;

            }

            if (c == 5)

            {

                \*ptr2 = '0';

                ptr2++;

            }

            c = 0;

        }

    }

    \*ptr2 = '\0';

    printf("\nthe stuffed character string is");

    printf("\n%s", stuff);

    ptr1 = stuff;

    ptr2 = destuff;

    while (\*ptr1 != '\0')

    {

        if (\*ptr1 == '0')

        {

            \*ptr2 = \*ptr1;

            ptr2++;

            ptr1++;

        }

        else

        {

            while (\*ptr1 == '1' && c != 5)

            {

                c++;

                \*ptr2 = \*ptr1;

                ptr2++;

                ptr1++;

            }

            if (c == 5)

            {

                ptr1++;

            }

            c = 0;

        }

    }

    \*ptr2 = '\0';

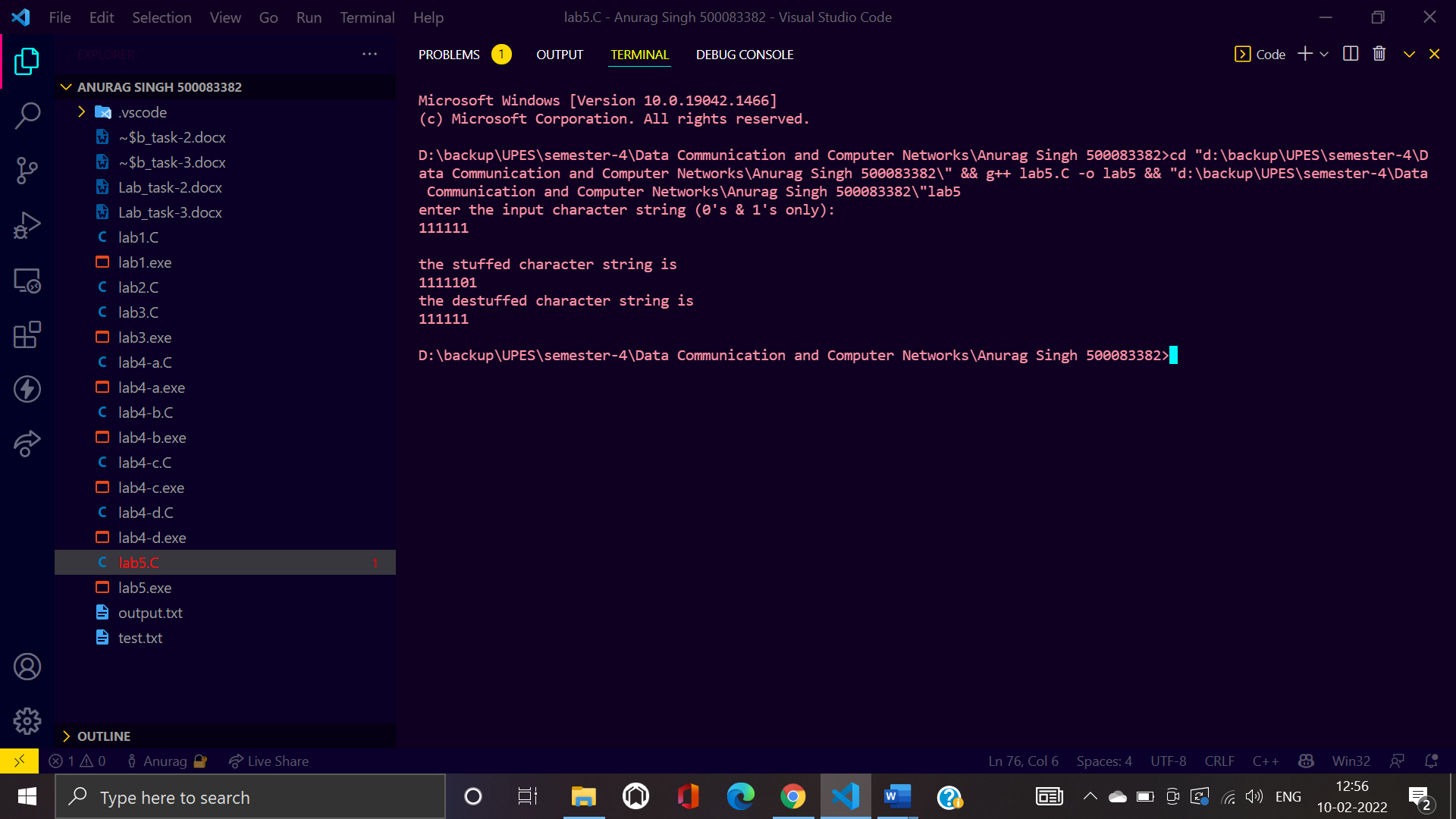
    printf("\nthe destuffed character string is");

    printf("\n%s\n", destuff);

    return 0;

}

**Screenshot:**



**Lab-6**

**Task: Design a program for error detection using C program**

**Code:**

#include <stdio.h>

int main()

{

    // design a program for error detection using c program

    int data[10]={1,0,1,1,0,1};

    int noise[10]={0,0,0,0,0,1};

    int result[10];

    // binary addition of data and noise

   int carry=0;

    for( int i = 0; i < 6; i++ )

    {

        int bitA = i < 6 && data[i] ? 1 : 0;

        int bitB = i < 6 && noise[i] ? 1 : 0;

        int sum = bitA + bitB + carry;

        result[i] = sum == 1 || sum == 3 ? 1 : 0;

        carry = sum > 1 ? 1 : 0;

    }

     result[ 6 ] = carry;

     printf("\n result:");

     for(int i=0;i<6;i++)

     {

         printf("%d",result[6-i]);

     }

     for(int i=0;i<6;i++){

         if(result[i]!=data[i]){

             printf("\nError detected at bit %d",i);

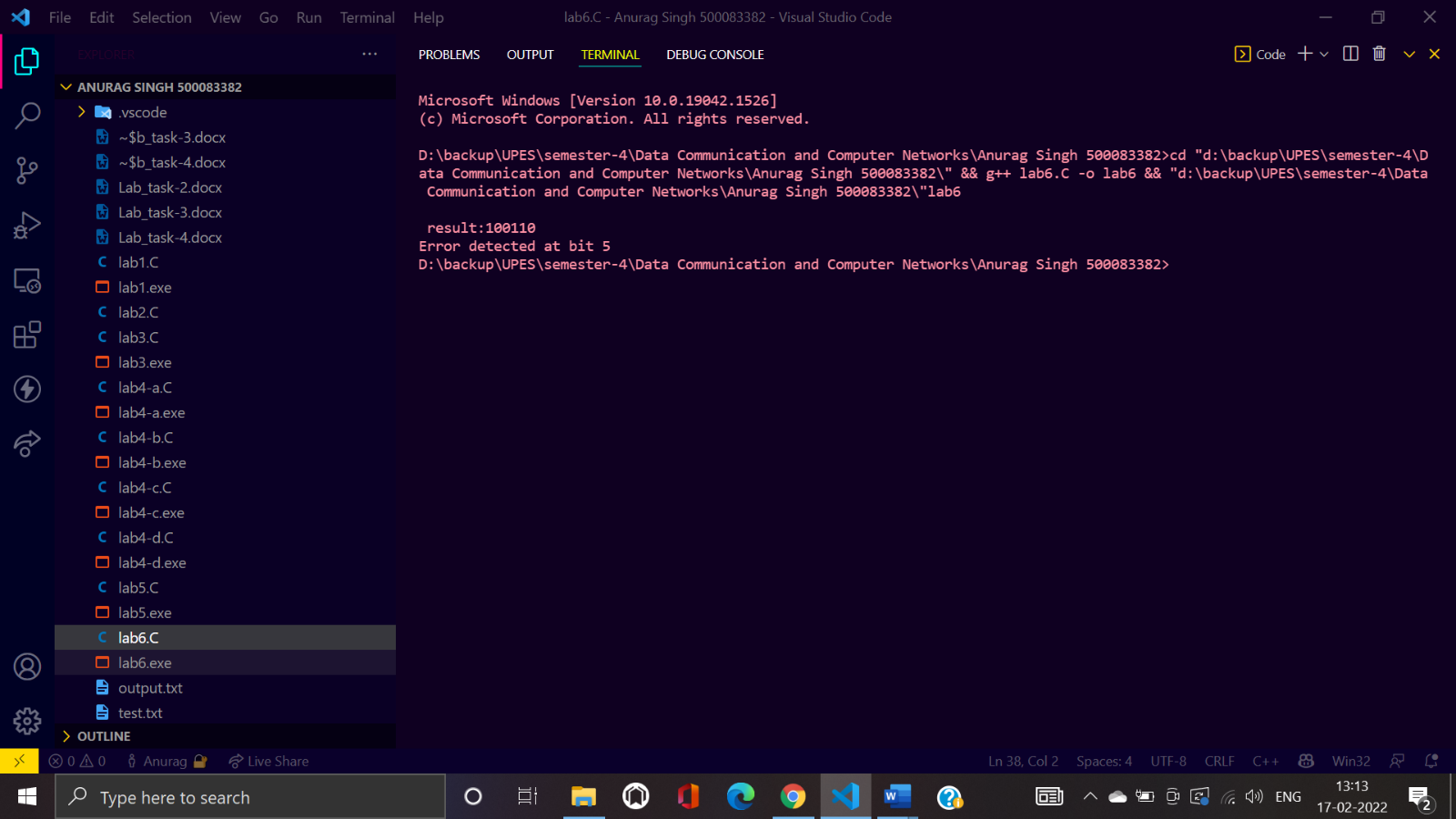
             break;

         }

     }

    return 0;

}

**Screenshot:**

**Lab-7**

**Task: Design a program for pure aloha protocol**

**Code**

#include <stdio.h>

#include <math.h>

#include <time.h>

#define FRAME\_TIME 250

int main()

{

float S1, S2, G, J, val[100];

int I, n, K, delay;

void wait();

clrscr();

printf("Please Give the Total Load : ");

scanf("%d", &n);

printf("Please Enter the value of load \n");

for (I=0; I<n; I++)

{

scanf("%f", &val[I]);

}

printf("\nOUTPUT 1: (THROUGHPUT Vs LOADCURVE)\n\n");

printf("s=g\*exp(-G) FOR SLOTTED ALOHA \* \n");

printf("s=g\*exp(-2G) FOR PURE ALPHA #\n")

printf("\n------ (THROUGHPUT PER FRAME TIME)----\n");

for(K=0; K<n; K++)

{

G=val[K];

S1 = G \* exp (-G);

S2 = G \* exp(-2 \* G);

printf("%1.3f", G );

for (I=0; I <=S1\*20; I++)

{

printf(" ");

}

printf("\*");

for(I=S2\*20; I<=S2\*75; I++ )

{

printf(" ");

}

printf("#\n");

}

printf("G (ATTEMPTS PER PACKET TIME) \n\n");

wait();

getch();

clrscr();

printf("\nOUTPUT 2 (DELAY Vs THROUGHPUT) \n\n");

printf("\n-----(THOUGHPUT PER FRAME TIME)----\n");

for(K=0; K<n; K++)

{

G=val[K];

S1 = G \* exp (-G);

printf("3");

for (I=0; I <=S1\*2.7; I++)

{

printf(" ");

}

printf("\*\n");

}

printf("\n");

printf("---- DELAY -----");

wait();

getch();

clrscr();

}

void wait()

{

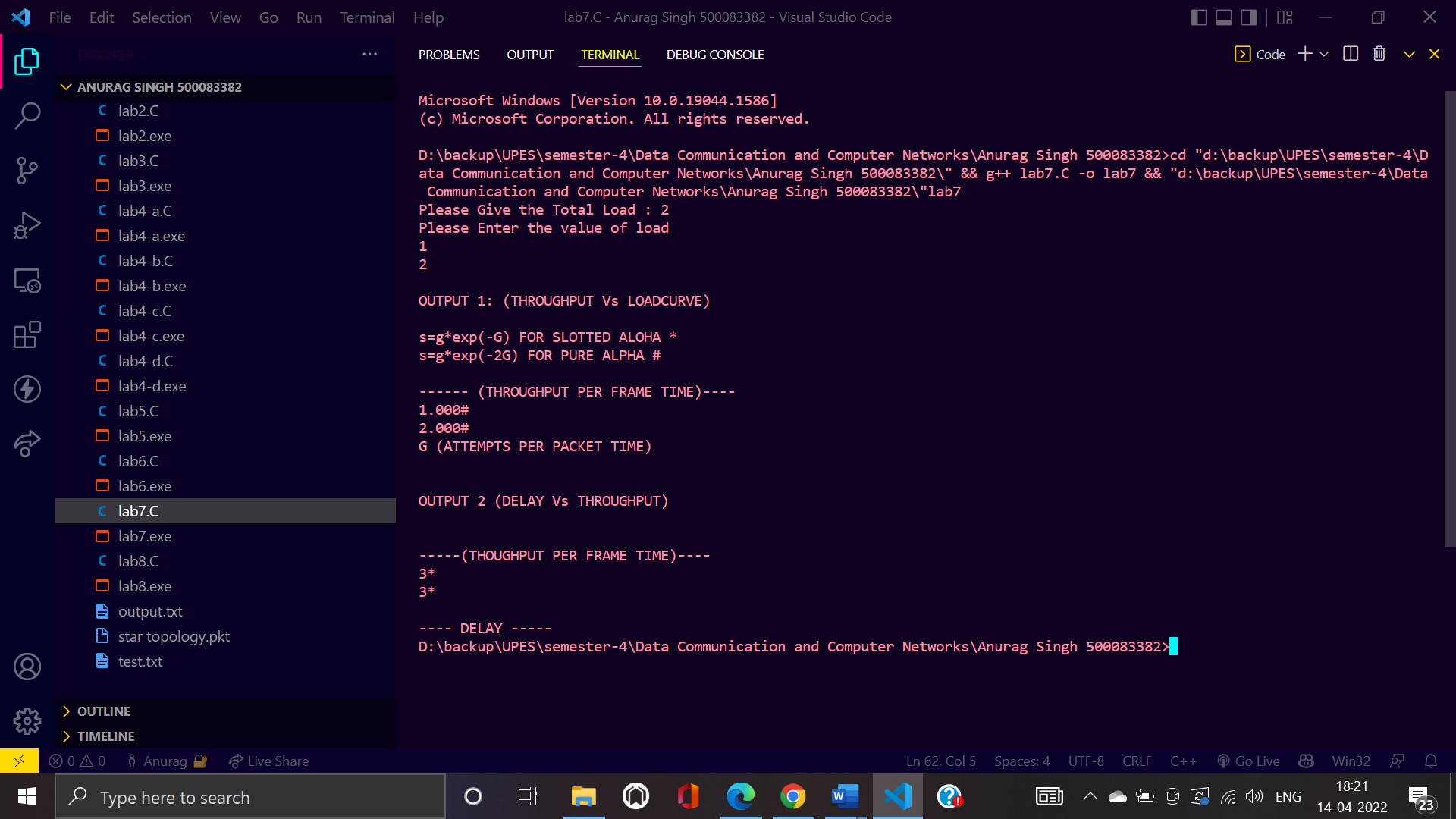
sound(440);

delay(300);

nosound();

}

**Screenshot:**



**Lab-8**

**Task: Design a program for CSMA**

**Code:**

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#include <time.h>

int main() {

    srand(time(0));

    int deviceDemand[10], maxBandwidth = 9, numOfDevice = 3;

    float avgBandwidth = maxBandwidth / numOfDevice, bandwidthProvide[10];

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

        deviceDemand[i] = rand() % 5 + 1;

    }

    printf("Bandwidth limit of the channel: %d\n", maxBandwidth);

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

        printf("Bandwidth demand for device %d: %d mbps\n", i+1, deviceDemand[i]);

    }

    float excessBandwidth = 0, condition[10], count = 0;

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

        if(deviceDemand[i] <= avgBandwidth) {

            bandwidthProvide[i] = deviceDemand[i];

            excessBandwidth += avgBandwidth - deviceDemand[i];

            condition[i] = 1;

        } else {

            bandwidthProvide[i] = avgBandwidth;

            condition[i] = 0;

            count++;

        }

    }

    printf("Excess Bandwidth after providing average bandwidth = %f\n", excessBandwidth);

    float BonusBandwidth = excessBandwidth/count;

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

        if(condition[i] == 0) {

            bandwidthProvide[i] += BonusBandwidth;

        }

    }

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

        if(deviceDemand[i] == bandwidthProvide[i]) {

            printf("Device %d: (%f mbps provided) Demand Satisfied\n", i+1, bandwidthProvide[i]);

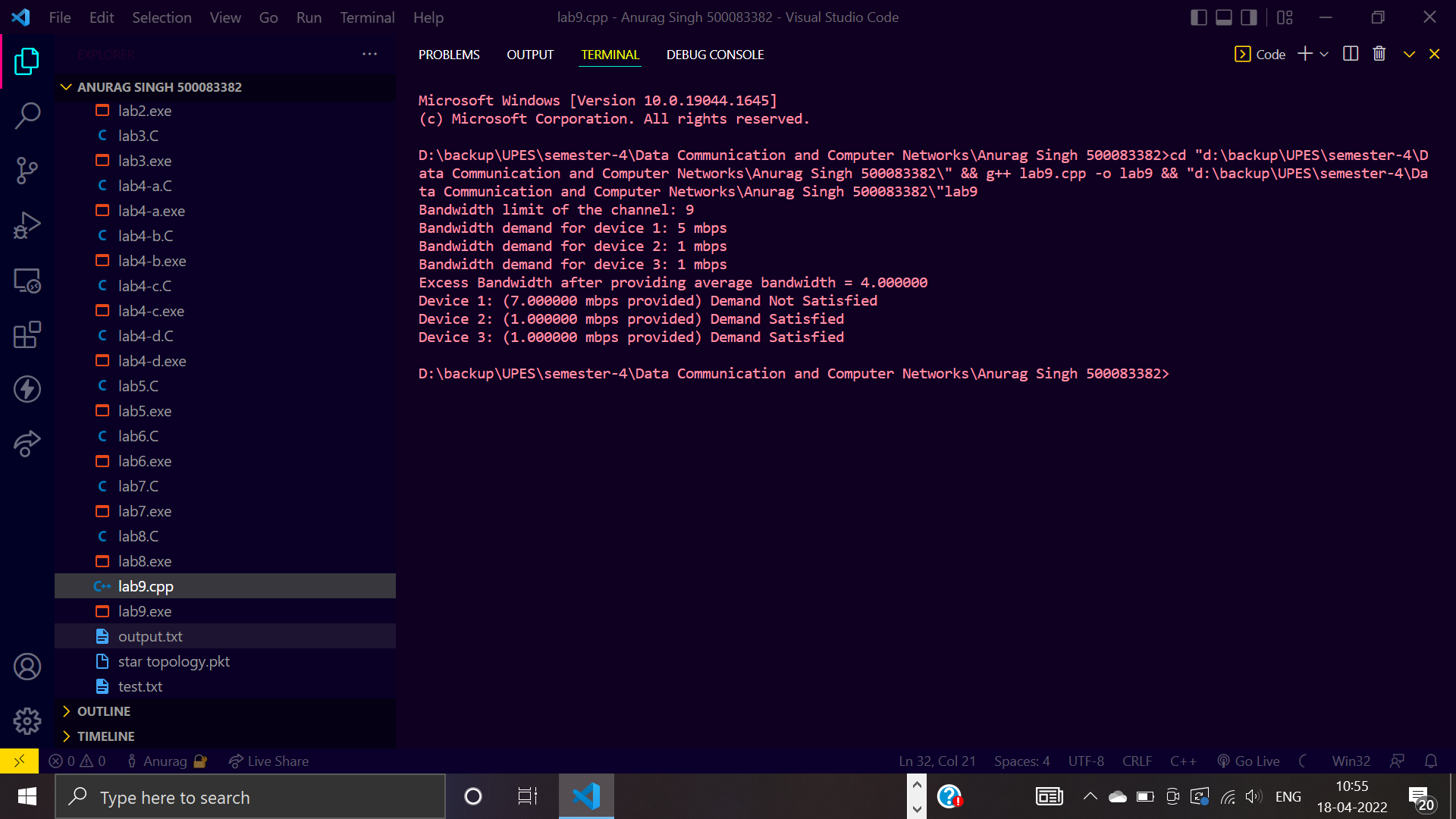
        } else {

            printf("Device %d: (%f mbps provided) Demand Not Satisfied\n", i+1, bandwidthProvide[i]);

        }

    }

}

**Screenshot:**

**Lab-9**

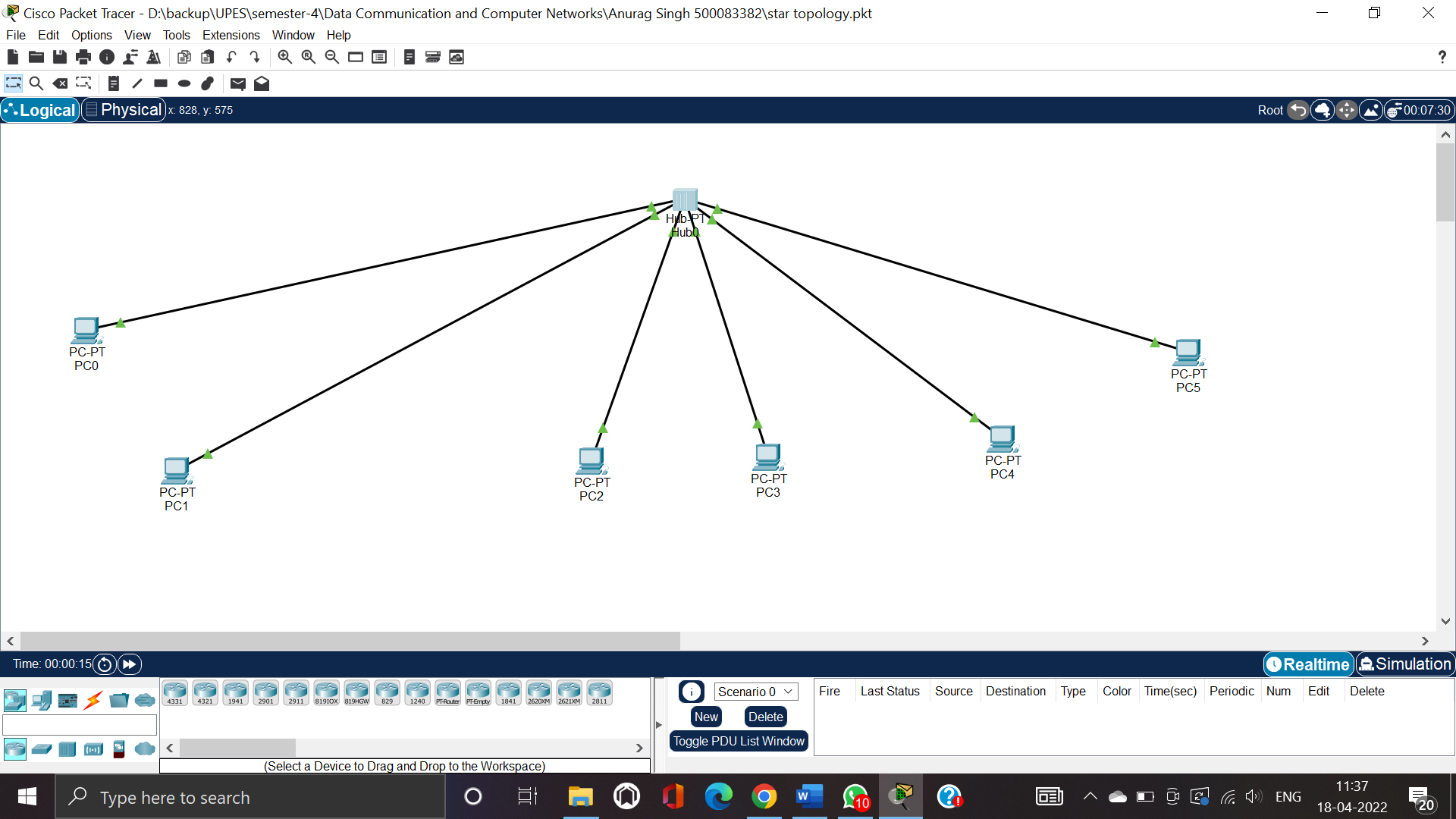
**Task: Static Routing in Cisco Packet Tracer**

**Theory**

In this network, a router and 6 PCs are used. Computers are connected with routers using a copper straight-through cable. After forming the network, to check network connectivity a simple PDU is transferred from PC0 to PC1. The network simulation status is successful. From this network, it can be observed that the router handles data transfers between multiple devices.

1. Procedure
2. Select the router and Open CLI.
3. Press ENTER to start configuring Router1.
4. Type enable to activate the privileged mode.
5. Type config t(configure terminal) to access the configuration menu.
6. Configure interfaces of Router1

**Screenshot:**



**Lab-10**

**Topic: Cisco Packet Tracer - Static Routing**

**Theory**

Routing is simply a process of choosing a route for delivering data to its destination. All hosts who can generate a routing table can do a routing. Routing process is needed when we are going to deliver packets of data to a network that isn’t directly connected with the sender.

IP Addresses

PC0: 192.168.1.2

PC1: 192.168.1.3

PC2: 192.168.2.2

PC3:192.168.2.3

PC4:192.168.3.2

PC5: 192.168.3.3

**Screenshot:**

