Name - Pranava Raman BMS

Roll: 2019103555

1. A. String length without pointer

#include<stdio.h>

int findlength(char string[]){

    int i=0;

    while(string[i]!='\0'){

        i++;

    }

    return i;

}

int main(){

    char c,string[100];

    int i;

    for(i=0; (i<100)&&((c=getchar())!='\n'); i++){

        string[i]=c;

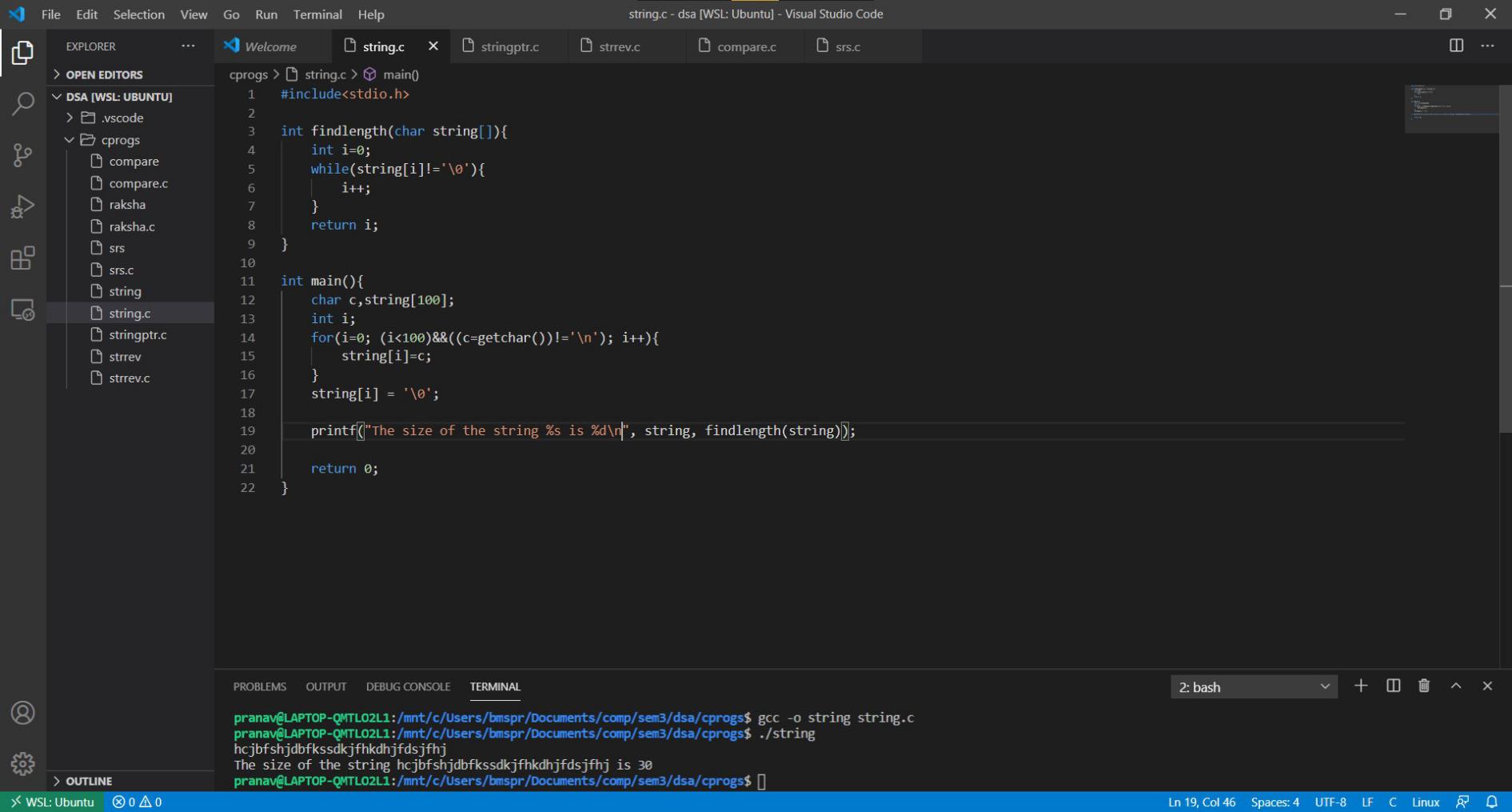
    }

    string[i] = '\0';

    printf("The size of the string %s is %d\n", string, findlength(string));

    return 0;

}



1. B. String length with pointer

#include<stdio.h>

int findlength(char string[]){

    int i=0;

    char \*a = string;

    while(\*a!='\0'){

        i++;

        a++;

    }

    return i;

}

int main(){

    char c,string[100];

    int i;

    for(i=0; (i<100)&&((c=getchar())!='\n'); i++){

        string[i]=c;

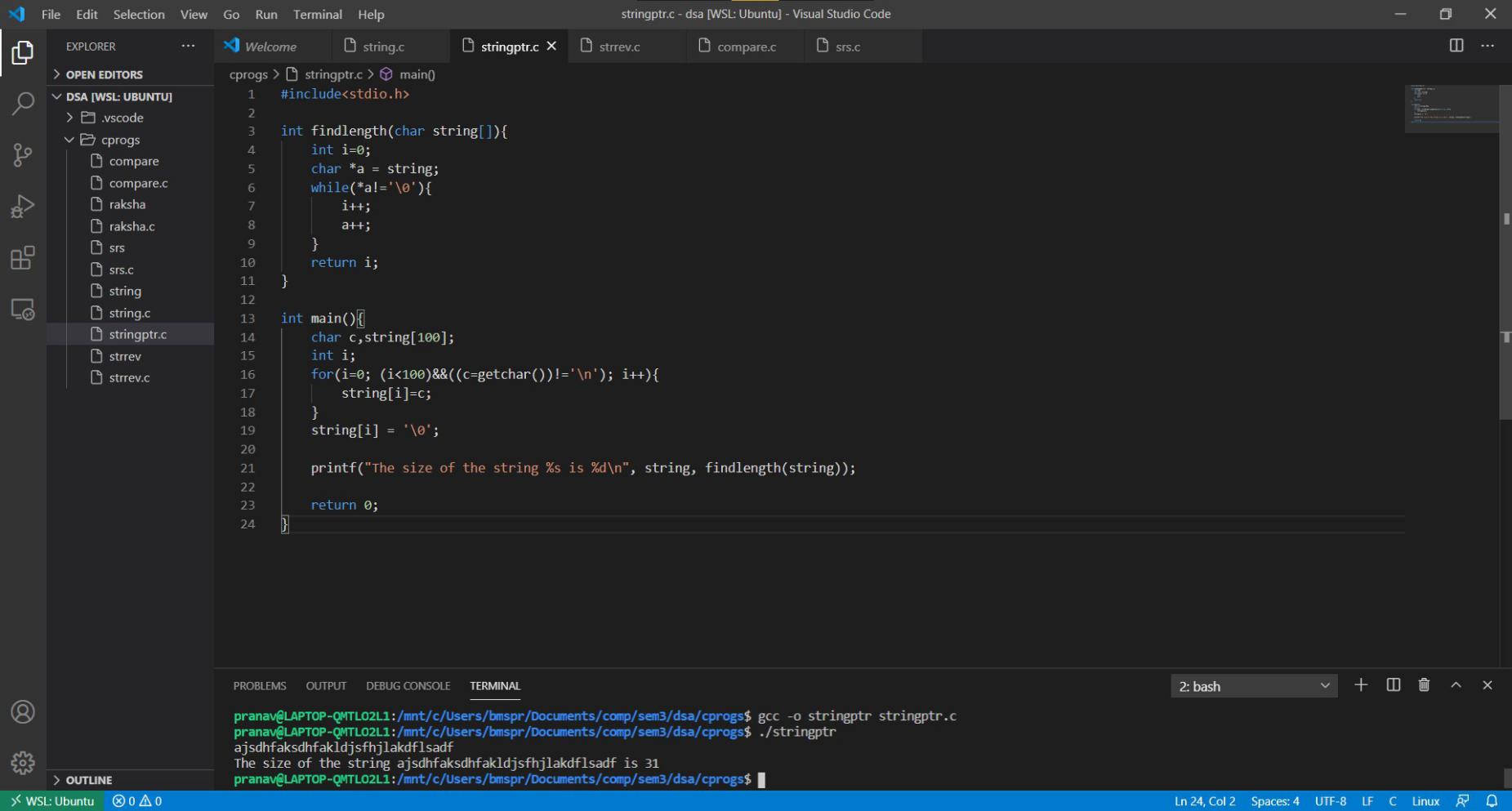
    }

    string[i] = '\0';

    printf("The size of the string %s is %d\n", string, findlength(string));

    return 0;

}



1. String Reversal using ptr

#include<stdio.h>

#define num 100

char string[num],newstr[num];

char \*revstr(char string[]){

    char \*a, \*b;

    for(a=string; \*a!='\0'; a++);

    a--;

    b = newstr;

    while(a!=string){

        \*b = \*a;

        a--;

        b++;

    }

    \*b = \*a;

    b++;

    \*b = '\0';

    return newstr;

}

int main(){

    char c;

     int i;

    for(i=0; (i<100)&&((c=getchar())!='\n'); i++){

        string[i]=c;

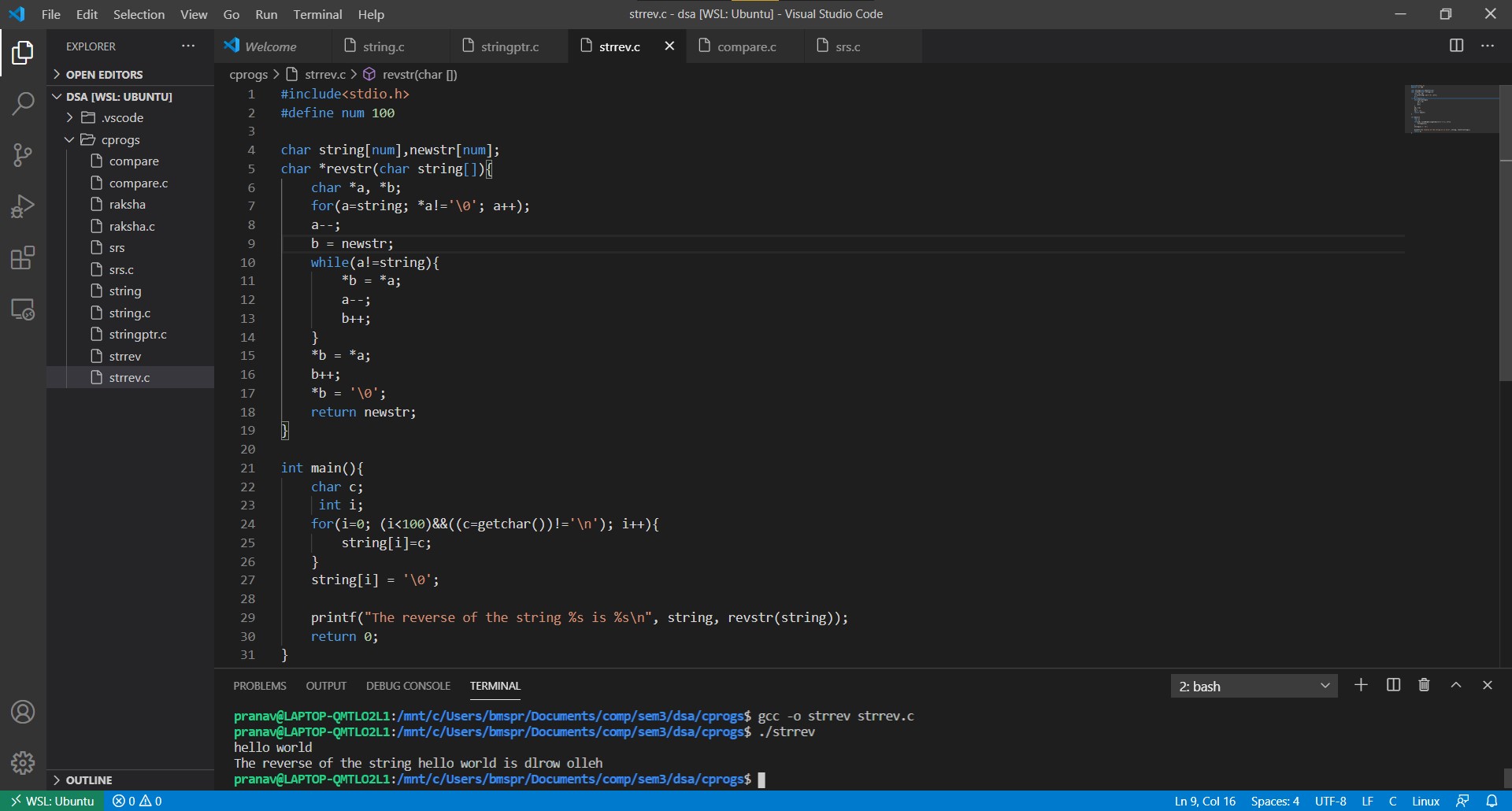
    }

    string[i] = '\0';

    printf("The reverse of the string %s is %s\n", string, revstr(string));

    return 0;

}



1. Comparing strings

#include<stdio.h>

#define num 100

void strcomp(char string1[], int l1, char string2[], int l2){

    if(l1>l2){

        printf("Strings %s and %s are unequal and %s has the larger length\n", string1, string2, string1);

        return;

    } else if(l1<l2){

        printf("Strings %s and %s are unequal and %s has the larger length\n", string1, string2, string2);

        return;

    }else if(l1==l2){

        char \*a = string1, \*b = string2;

        int diff=0;

        while(\*a!='\0'){

            if(\*a!=\*b){

                diff=1;

                break;

            }

            a++;

            b++;

}

        if(diff==0){

            printf("Strings %s and %s are equal \n", string1, string2);

            return;

        }else{

            printf("Strings %s and %s have equal length but are not equal \n", string1, string2);

            return;

        }

    }

}

int main(){

    char str1[num], str2[num];

    char c;

     int i, l1, l2;

    for(i=0; (i<num)&&((c=getchar())!='\n'); i++){

        str1[i]=c;

    }

    str1[i] = '\0';

    l1 = i;

    for(i=0; (i<num)&&((c=getchar())!='\n'); i++){

        str2[i]=c;

    }

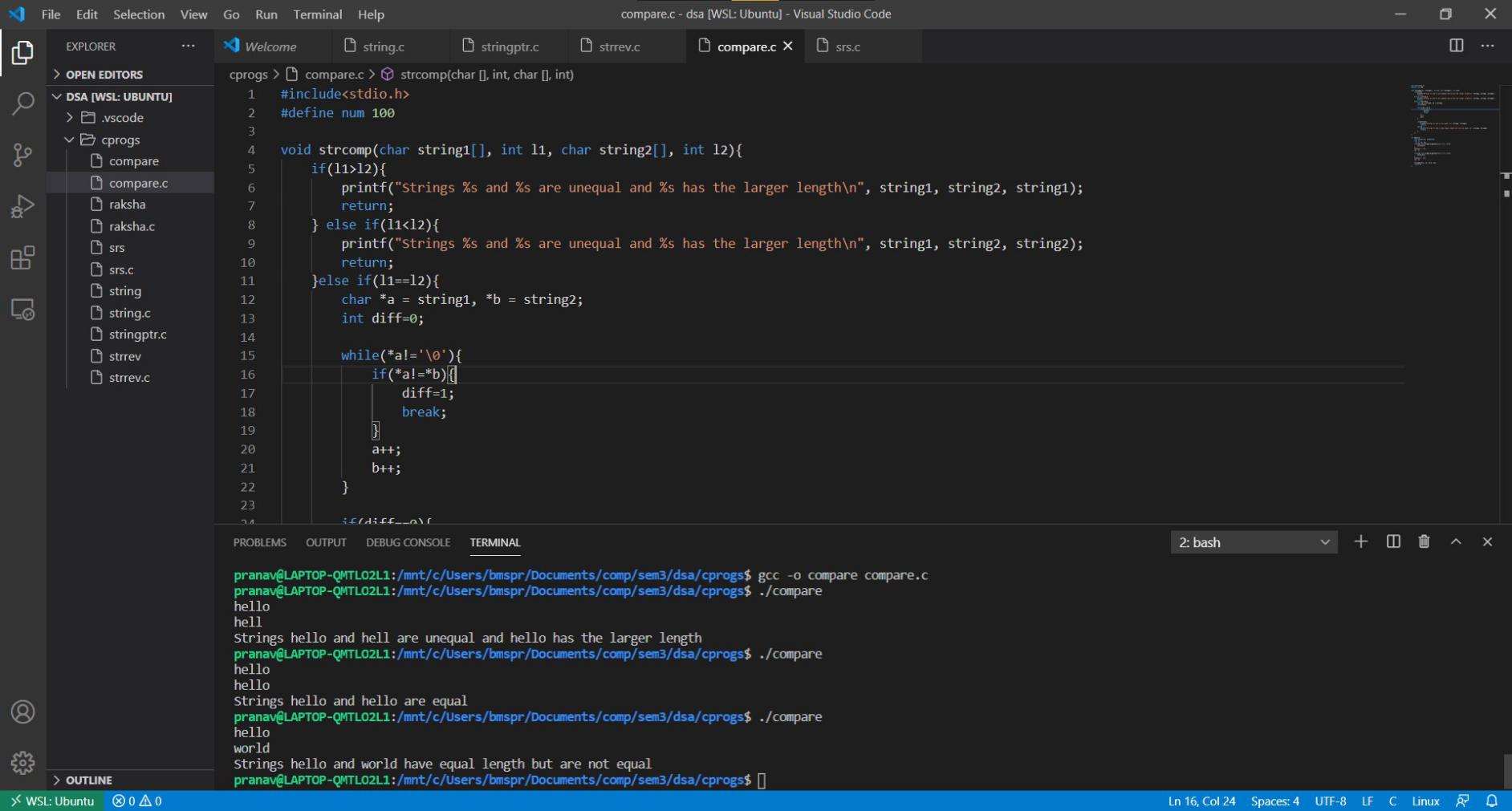
    str2[i] = '\0';

    l2 = i;

    strcomp(str1, l1, str2, l2);

    return 0;

}



1. Matrix Multiplication

#include<stdio.h>

#include<stdlib.h>

int main(){

    int a[10][10],b[10][10],c[10][10];

    int x,y,r,s,i,j,k;

    printf("Enter the number of rows and columns for first matrix\n");

    scanf("%d",&r);

    scanf("%d",&s);

    printf("Enter the number of rows and columns for second matrix\n");

    scanf("%d",&x);

    scanf("%d",&y);

    if(s == x){

        printf("Enter the First matrix\n");

        for(i=0;i<r;i++)

        {

            for(j=0;j<s;j++)

            {

                scanf("%d",&a[i][j]);

            }

        }

        printf("Enter the Second matrix:-\n");

        for(i=0;i<x;i++)

        {

            for(j=0;j<y;j++)

            {

                scanf("%d",&b[i][j]);

            }

        }

        printf("Multiplication of the two matrices are\n");

        for(i=0;i<r;i++)

        {

            for(j=0;j<s;j++)

            {

                c[i][j]=0;

                for(k=0;k<y;k++)

                {

                    c[i][j] += a[i][k] \* b[k][j];

                }

            }

        }

        for(i=0;i<r;i++)

        {

            for(j=0;j<y;j++)

            {

                printf("%d\t",c[i][j]);

            }

            printf("\n");

        }

    }

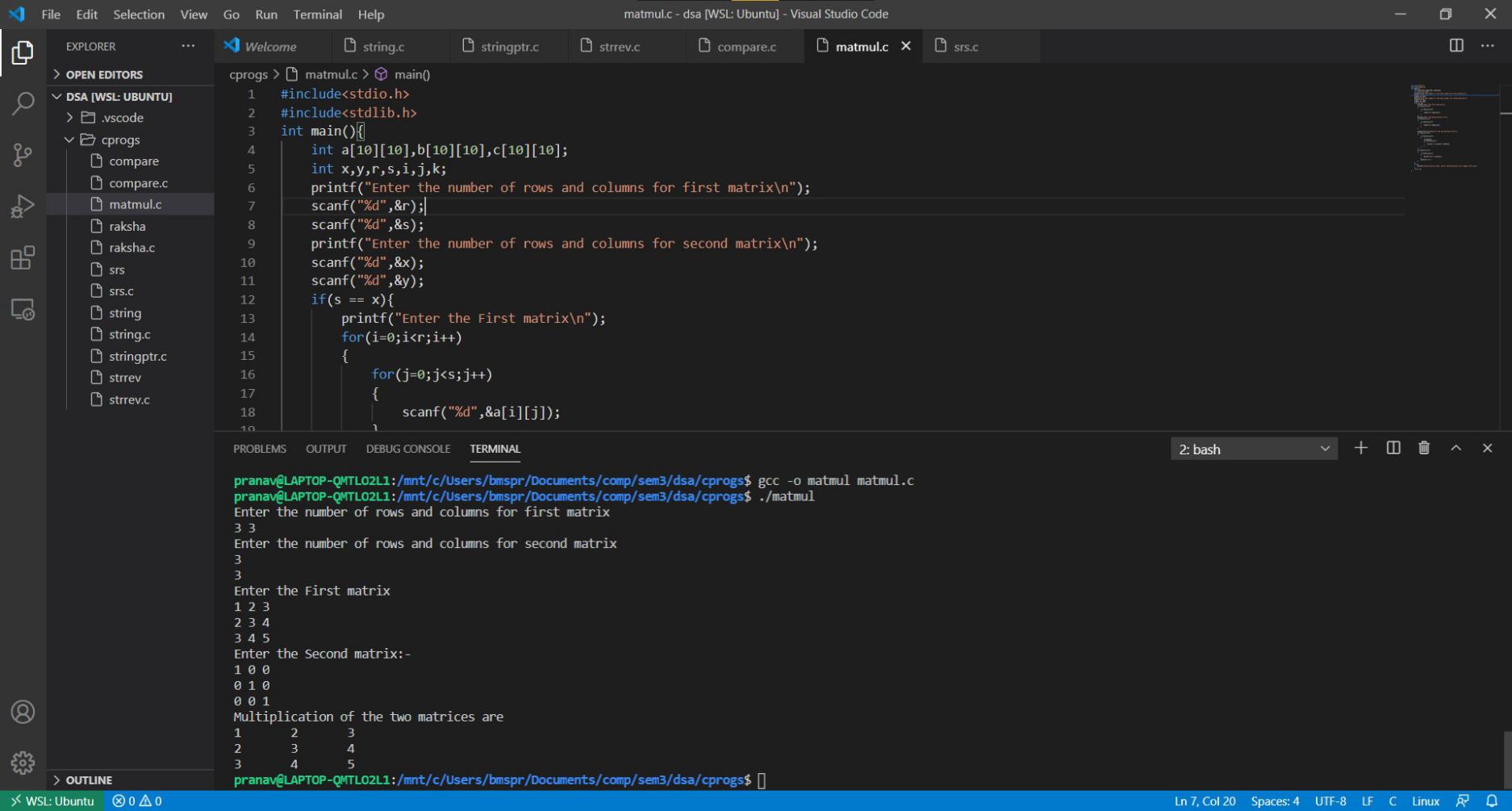
    else{

        printf("Invalid matrix order, matrix multiplication can't happen that way");

    }

    return 0;

}



1. Matrix Addition using ptr

#include<stdio.h>

#include<stdlib.h>

int main(){

    int a[10][10],b[10][10],c[10][10],p,q,i,j,k;

    printf("Enter the number of rows\n");

    scanf("%d",&p);

    printf("Enter the number of columns\n");

    scanf("%d",&q);

    printf("Enter the First matrix:\n");

    for(i=0;i<p;i++)

    {

        for(j=0;j<q;j++)

        {

            scanf("%d",\*(a+i+j));

        }

    }

    printf("Enter the Second matrix:\n");

    for(i=0;i<p;i++)

    {

        for(j=0;j<q;j++)

        {

            scanf("%d",\*(b+i+j));

        }

    }

    for(i=0;i<p;i++)

    {

        for(j=0;j<q;j++)

        {

            \*\*(c+i+j) = \*\*(a+i+j) + \*\*(b+i+j);

        }

    }

    printf("Sum of the two matrices using pointers is:\n");

    for(i=0;i<p;i++)

    {

        for(j=0;j<q;j++)

        {

            printf("%d\t",\*\*(c+i+j));

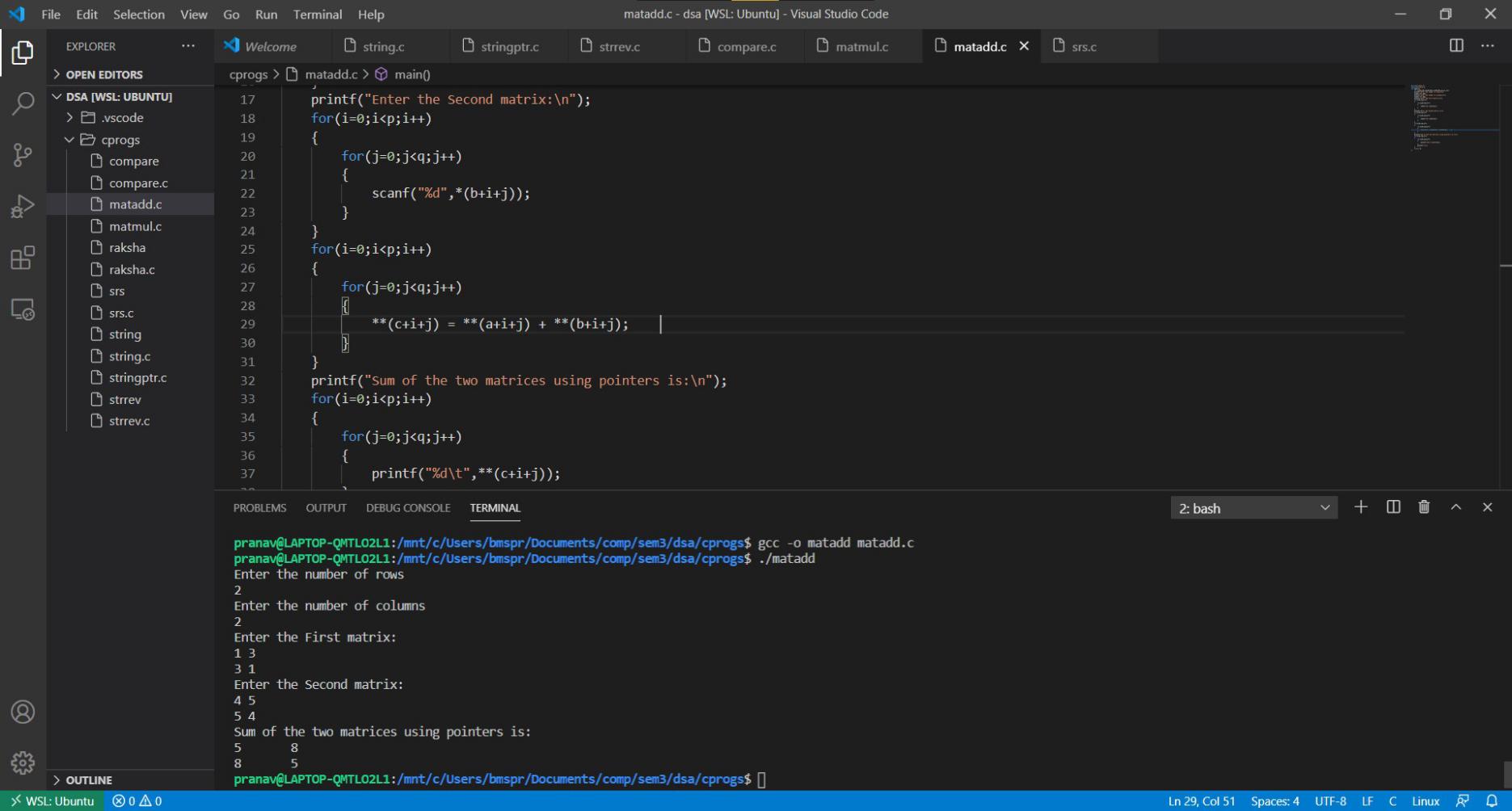
        }

        printf("\n");

    }

    return 0;

}



1. AREA, VOLUME using function

#include<stdio.h>

void cuboid(){

    float l,b,h;

    printf("Enter the length, breadth, height: ");

    scanf("%f%f%f", &l, &b, &h);

    float volume = l\*b\*h;

    float surface= 2\*(l\*b+ b\*h + h\*l);

    printf("Volume = %f\nSurface area = %f\n", volume, surface);

}

void sphere(){

    float r;

    printf("\nEnter the radius of sphere: ");

    scanf("%f",&r);

    float volume = 1.33\*3.14\*r\*r\*r;

    float surface = 4\*3.14\*r\*r;

    printf("Volume = %f\nSurface area = %f\n", volume, surface);

}

void cube(){

    float x;

    printf("\nEnter the side of cube: ");

    scanf("%f",&x);

    float volume = x\*x\*x;

    float surface = 6\*x\*x;

    printf("Volume = %f\nSurface area = %f\n", volume, surface);

}

int main(){

    int ch;

    while(1){

        printf("1.Cuboid\n2.Sphere\n3.Cube\n4.Exit\nEnter your choice:");

        scanf("%d", &ch);

        if(ch==4)

break;

        switch (ch)

        {

        case 1:

            cuboid();

            break;

        case 2:

            sphere();

            break;

        case 3:

            cube();

            break;

        default:

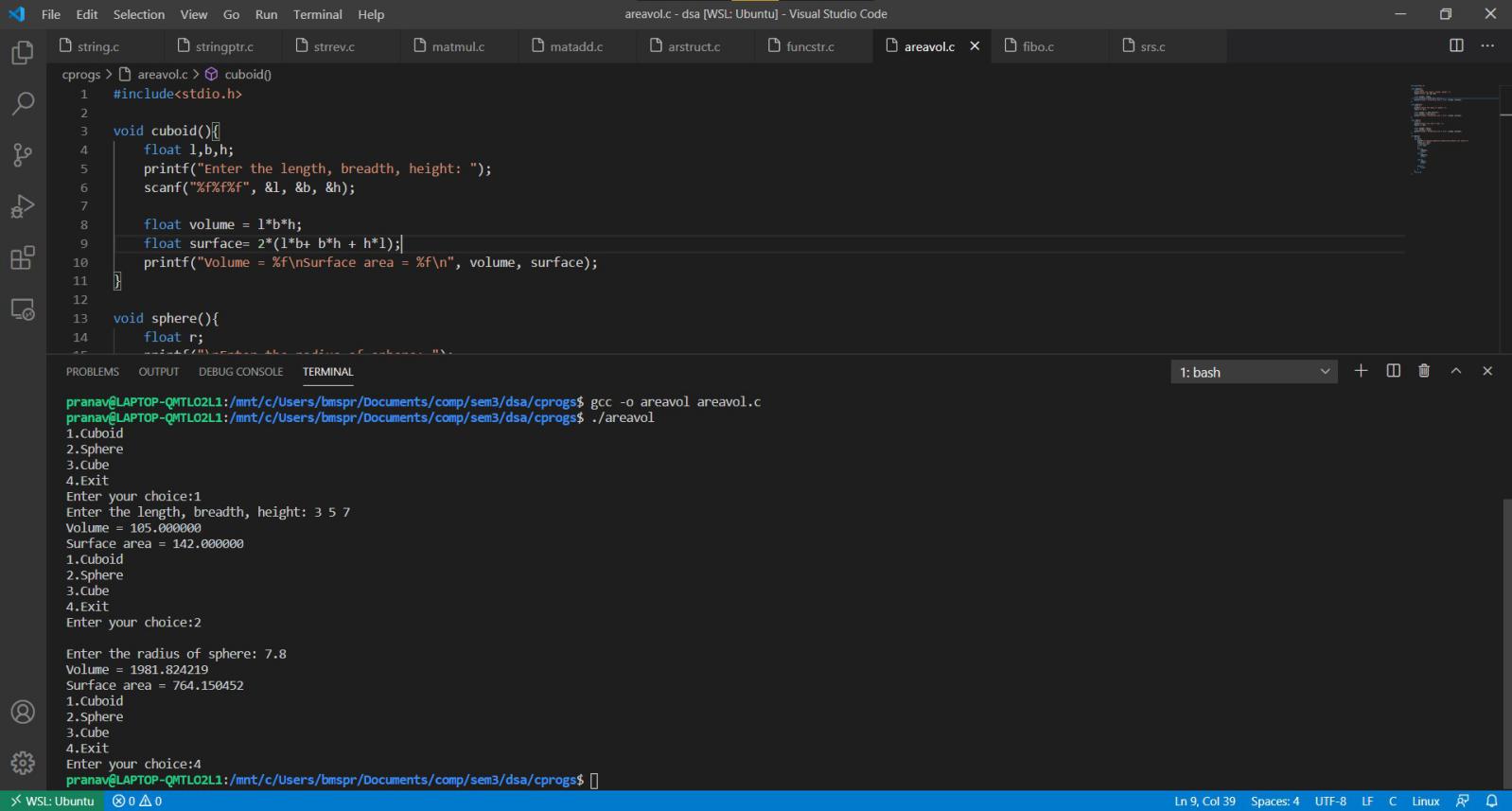
            break;

        }

    }

    return 0;

}



1. Structure in function

#include<stdio.h>

#include<stdlib.h>

struct student{

    char name[20];

    int rollno;

};

void initialise(struct student\* s){

    printf("Enter student name\n");

    fgets(s->name, 20, stdin);

    printf("Enter student roll number\n");

    scanf("%d",&(s->rollno));

}

void display(struct student\* s){

    printf("Student name: %s\n",s->name);

    printf("Roll number: %d\n",s->rollno);

}

int main(){

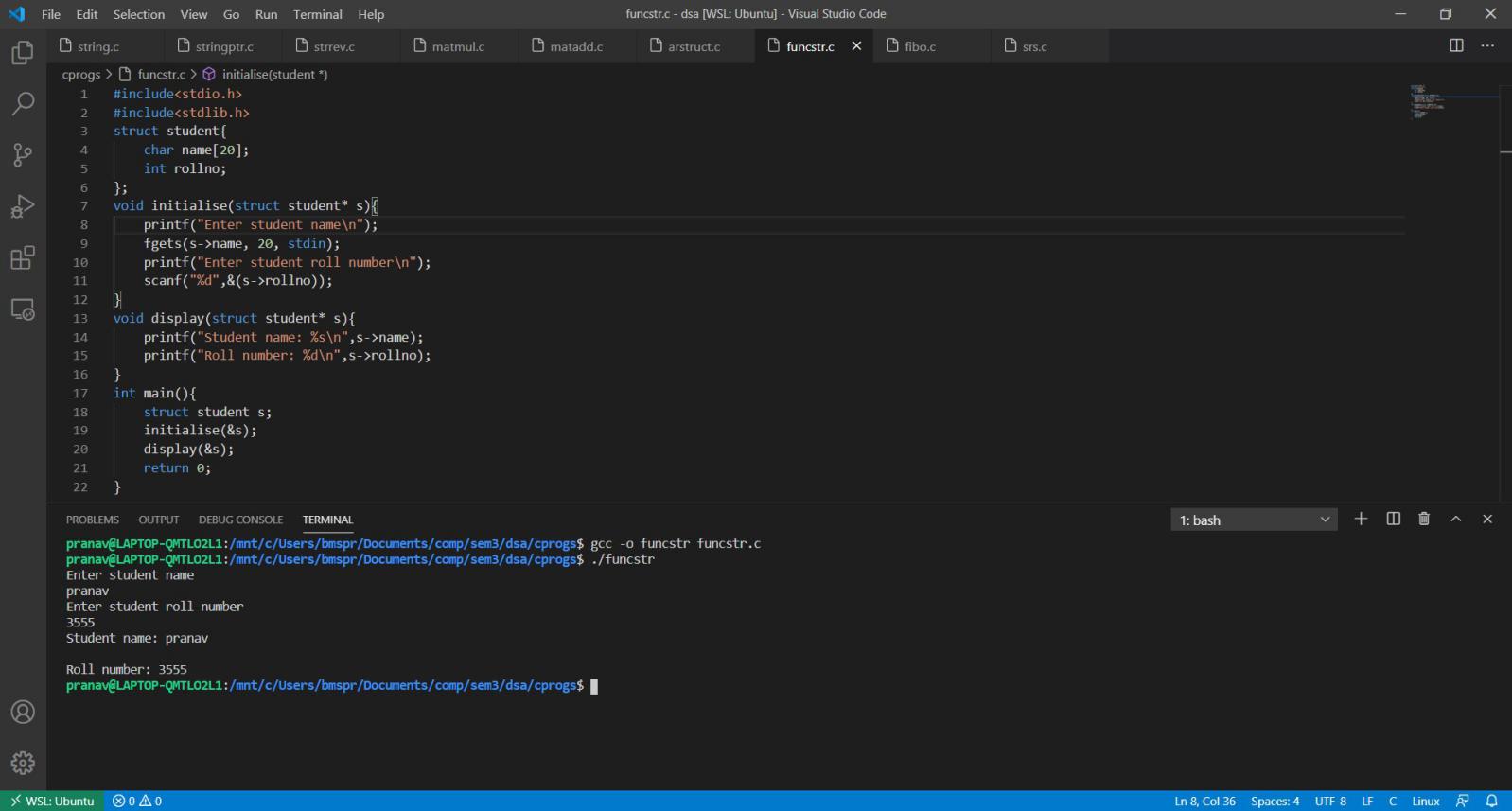
    struct student s;

    initialise(&s);

    display(&s);

    return 0;

}



1. Array of structures

#include<stdio.h>

#define size 100

struct student{

    int regn;

    float cgpa;

};

void main()

{

    struct student students[size];

int count = 0,i;

    printf("\n Enter students reg.no. followed by cgpa and when you want to stop, enter reg.no as '0': \n");

    for(i=0; i<size; i++)

    {

        printf("Enter the details of student %d \n", i+1);

        printf("Enter the regn no. :");

        scanf("%d", &(students[i].regn));

        if(students[i].regn==0)

            break;

        printf("Enter the CGPA: ");

        scanf("%f", &(students[i].cgpa));

        count++;

    }

    printf("\n Showing all details: ");

    for(i=0; i<count; i++)

    {

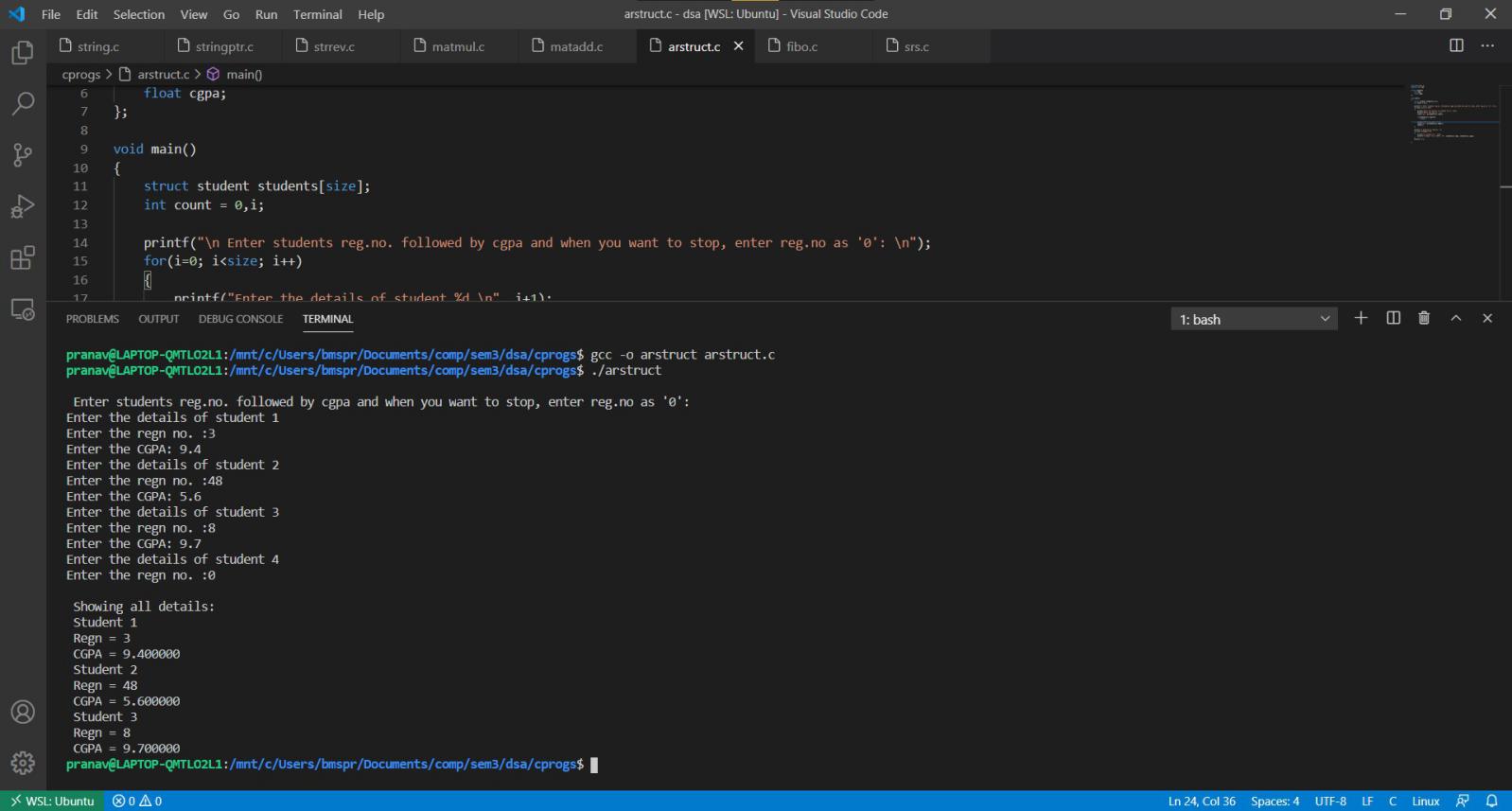
        printf("\n Student %d ", i+1);

        printf("\n Regn = %d \n CGPA = %f", students[i].regn, students[i].cgpa);

    }

    printf("\n");

}



1. Recursive Function (fibonacci series)

#include<stdio.h>

int fib(int n){

    if(n <= 1)

        return n;

    else

        return fib(n-1) + fib(n-2);

}

int main(){

    int n,res;

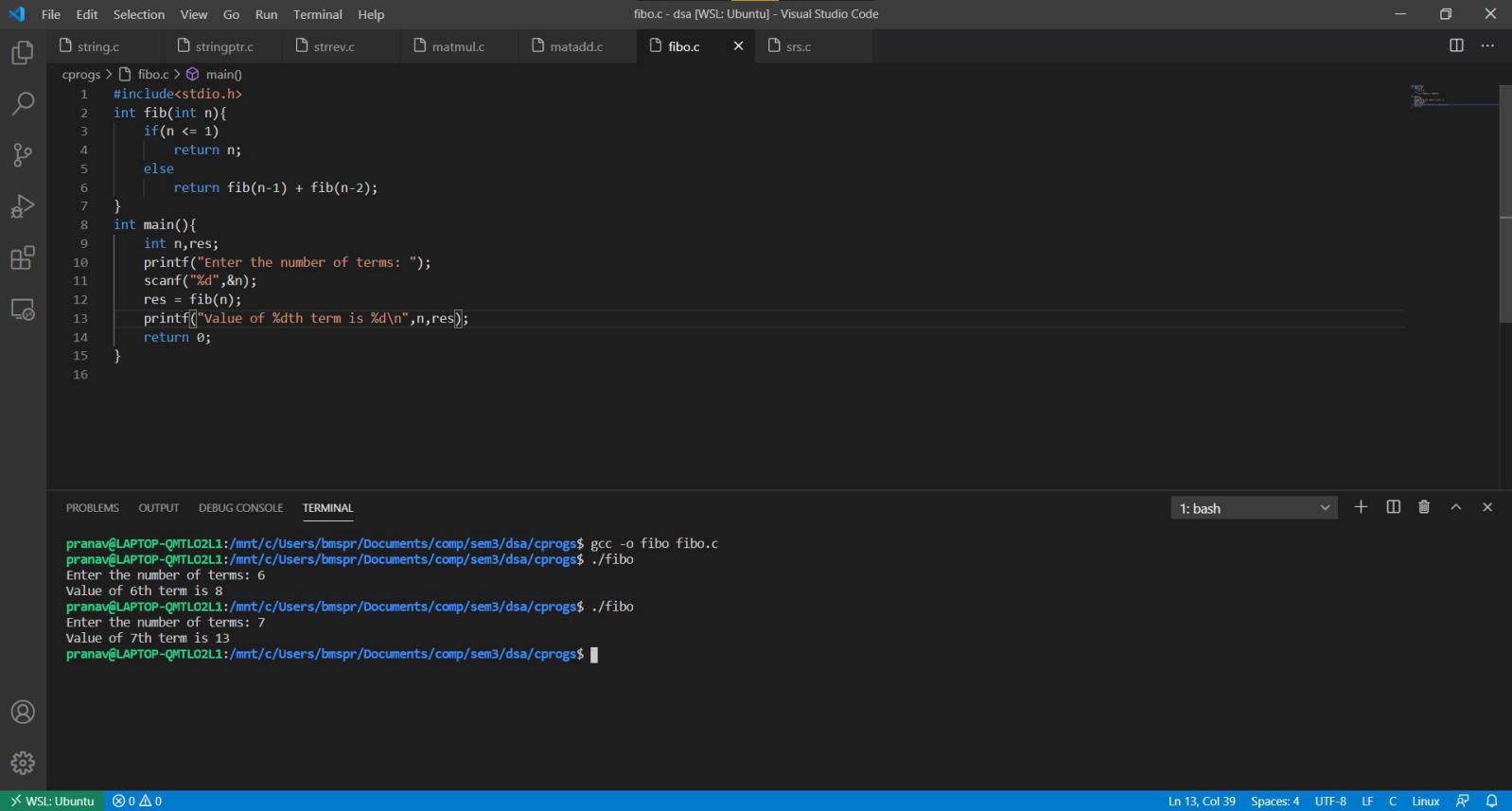
    printf("Enter the number of terms: ");

    scanf("%d",&n);

    res = fib(n);

    printf("Value of %dth term is %d\n",n,res);

    return 0; }



1. Self Referential Structures (Player’s names)

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

#define num 30

struct node{

    char name[num];

    int jerseyno;

    struct node\* next;

}\*head = NULL,\*temp = NULL;

void create(char \*nam, int n){

    struct node\*p = (struct node\*)malloc(sizeof(struct node));

    int l = strlen(nam);

    int i;

    for(i=0;i<l;i++)

        p->name[i] = nam[i];

    p->name[i] = '\0';

    p->jerseyno = n;

    p->next = NULL;

    if(head == NULL){

        head = p;

        temp = p;

    }

    else{

        temp->next = p;

        temp = p;

    }

}

void display(){

    printf("\nYour Entered Players Details are: \n");

    struct node\*p = head;

    while(p != NULL){

        printf("Name: %s",p->name);

        printf("Jersey no: %d\n",p->jerseyno);

        p = p->next;

    }

}

int main(){

    int jerseyno;

    char name[20];

    printf("Press 0 in jersey number to quit\n");

    while(1){

        printf("Enter jersey no: ");

        scanf("%d",&jerseyno);

        if(jerseyno == 0)

            break;

        getchar();

        printf("Enter the player's name: ");

        fgets(name, num, stdin);

        if(jerseyno == 0)

            break;

        create(name, jerseyno);

    }

    display();

    return 0;

}

