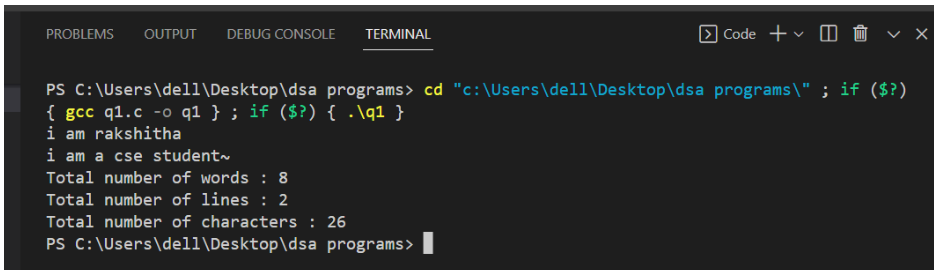
## 1) Develop a C program to count the number of words, characters and line in a given text.

Program:

```
#include<stdio.h>

int main()

{

char str[200];

int line, word, ch;

line = word = ch = 0;

printf("Enter string terminated with ~ :\n");

scanf("%[^~]", str);

for(int i=0; str[i]!='\0'; i++)

{

if(str[i]=='\n')

{

line++;

word++;

}

else

{

if(str[i]==' '||str[i]=='\t')

{

word++;

ch++;

}

else {

ch++;

}
}
}
printf("\nCharacter counts = %d\n", ch);
printf("Word counts = %d\n", word);
printf("Line counts = %d\n", line);
return 0;
}
```



```
PS C:\Users\dell\Desktop\dsa programs> cd "c:\Users\dell\Desktop\dsa programs\" ; if ($?)
{ gcc q1.c -o q1 } ; if ($?) { .\q1 }
i am rakshitha
i am a cse student~
Total number of words : 8
Total number of lines : 2
Total number of characters : 26
PS C:\Users\dell\Desktop\dsa programs>
```

**2) Define a structure CENSUS with the following members:**

**• A character array city[ ] to store names**

**• A long integer to store population of the city**

**• A float member to store the area of the city**

**Develop a C program to perform the following operations:**

**i. To read details for 5 cities**

**ii. To sort the list alphabetically**

**iii. To sort the list based on population**

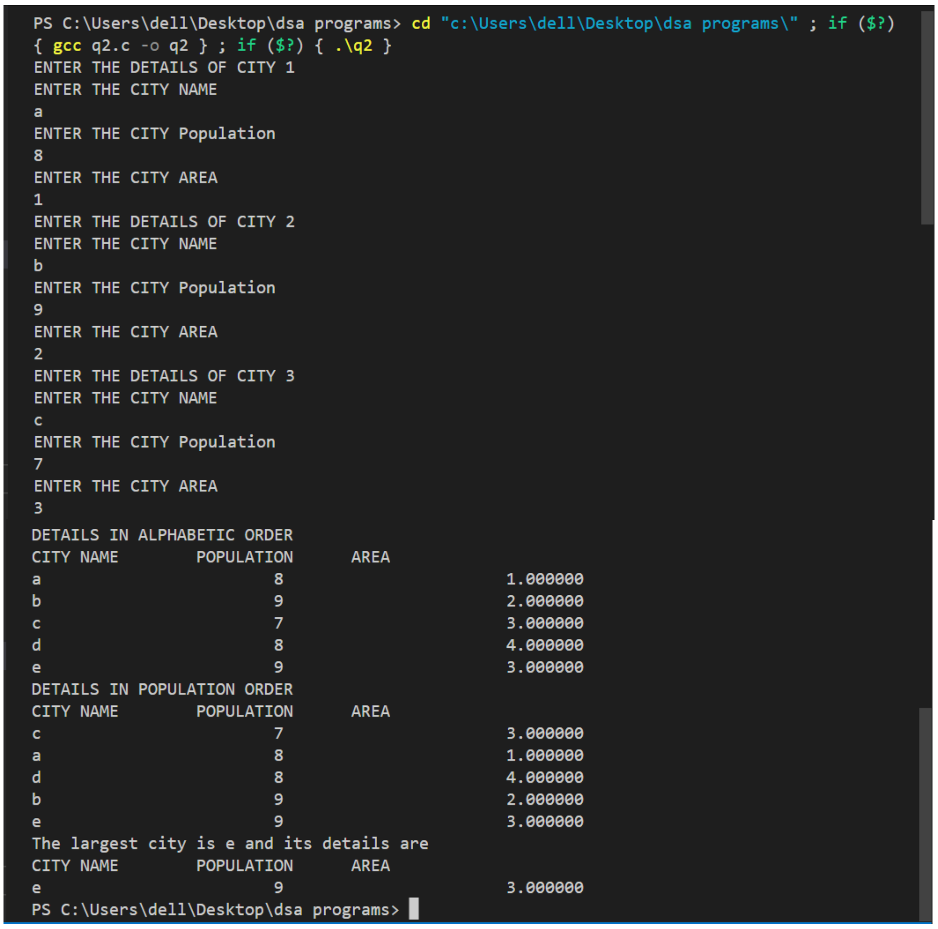
**iv. To display the city with the largest area.**

Program:

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
void read();
void sortalp();
void sortpop();
void display();

void largest();

typedef struct

{

char city[20];

long pop;

float area;

}CENSUS;

CENSUS c[5];

void swap(CENSUS,CENSUS);

int main()

{

read();

sortalp();

printf("DETAILS IN ALPHABETIC ORDER\n");

display();

sortpop();

printf("DETAILS IN POPULATION ORDER\n");

display();

largest();

}

void read()

{

int i;

for(i=0;i<5;i++)

{

printf("ENTER THE DETAILS OF CITY %d\n",i+1);

printf("ENTER THE CITY NAME\n");
scanf("%s",c[i].city);

printf("ENTER THE CITY Population\n");

scanf("%ld",&c[i].pop);

printf("ENTER THE CITY AREA\n");

scanf("%f",&c[i].area);}}

void sortalp()

{

int i,j;

char tcity[20];

long tpop;

float tarea;

for(j=0;j<5;j++)

for(i=0;i<5-1;i++)

if(strcmp(c[i].city,c[i+1].city)>0)

//swap(c[i],c[i+1]);

{

strcpy(tcity,c[i].city);

strcpy(c[i].city,c[i+1].city);

strcpy(c[i+1].city,tcity);

tpop=c[i].pop;

c[i].pop=c[i+1].pop;

c[i+1].pop=tpop;

tarea=c[i].area;

c[i].area=c[i+1].area;

c[i+1].area=tarea;

}

}

void sortpop()
```

```
{
int i,j;
char tcity[20];
long tpop;
float tarea;
for(j=0;j<5;j++)
for(i=0;i<5-1;i++)
if(c[i].pop>c[i+1].pop)
//swap(c[i],c[i+1]);
{
strcpy(tcity,c[i].city);
strcpy(c[i].city,c[i+1].city);
strcpy(c[i+1].city,tcity);
tpop=c[i].pop;
c[i].pop=c[i+1].pop;
c[i+1].pop=tpop;
tarea=c[i].area;
c[i].area=c[i+1].area;
c[i+1].area=tarea;
}
}
void display()
{
int i;
printf("CITY NAME \t POPULATION \t AREA\n");
for(i=0;i<5;i++)
printf("%s \t\t\t %ld \t\t\t %f\n",c[i].city,c[i].pop,c[i].area);
}
```

```
void largest()
{ int i;
int l=0;
for(i=0;i<5-1;i++)
if(c[i].area<c[i+1].area)
l=i+1;
else
l=i;
printf("The largest city is %s and its details are\n",c[l].city);
printf("CITY NAME \t POPULATION \t AREA\n");
printf("%s \t\t\t %ld \t\t\t %f\n",c[l].city,c[l].pop,c[l].area);
}
```

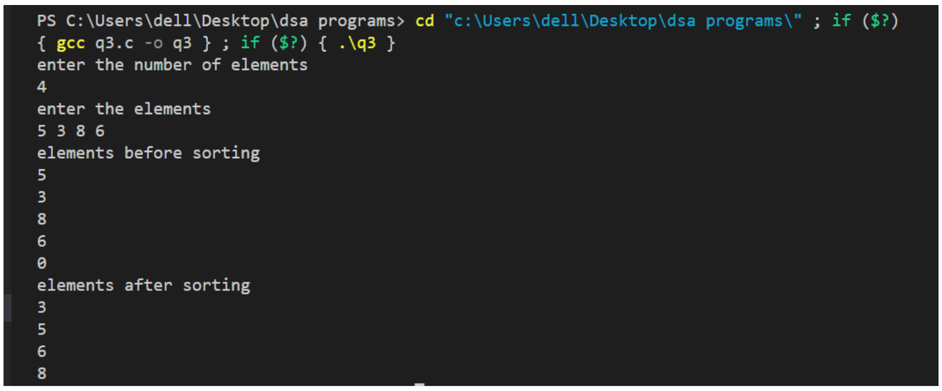


```
PS C:\Users\dell\Desktop\dsa programs> cd "c:\Users\dell\Desktop\dsa programs\" ; if ($?)
{ gcc q2.c -o q2 } ; if ($?) { .\q2 }
ENTER THE DETAILS OF CITY 1
ENTER THE CITY NAME
a
ENTER THE CITY Population
8
ENTER THE CITY AREA
1
ENTER THE DETAILS OF CITY 2
ENTER THE CITY NAME
b
ENTER THE CITY Population
9
ENTER THE CITY AREA
2
ENTER THE DETAILS OF CITY 3
ENTER THE CITY NAME
c
ENTER THE CITY Population
7
ENTER THE CITY AREA
3
DETAILS IN ALPHABETIC ORDER
CITY NAME        POPULATION      AREA
a                        8                     1.000000
b                        9                     2.000000
c                        7                     3.000000
d                        8                     4.000000
e                        9                     3.000000
DETAILS IN POPULATION ORDER
CITY NAME        POPULATION      AREA
c                        7                     3.000000
a                        8                     1.000000
d                        8                     4.000000
b                        9                     2.000000
e                        9                     3.000000
The largest city is e and its details are
CITY NAME        POPULATION      AREA
e                        9                     3.000000
PS C:\Users\dell\Desktop\dsa programs>
```

**3) Develop a C program to implement Selection sort for an integer array element and also demonstrate the same for the given input and also specify the number of passes.**

Program:

```
#include<stdio.h>

int main()

{

int i,j,a[20],n,temp,small;

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

scanf("%d",&n);

printf("Enter the array of elements\n");

for(i=0;i<=n-1;i++)

scanf("%d",&a[i]);

for(i=0;i<=n-1;i++)

{

small=i;

for(j=i+1;j<=n;j++)

{

if(a[j]<a[small])

small=j;

}

temp=a[i];

a[i]=a[small];

a[small]=temp;

}

printf("Sort elements are :\n");

for(i=0;i<=n-1;i++)

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

return 0; }
```



```
PS C:\Users\dell\Desktop\dsa programs> cd "c:\Users\dell\Desktop\dsa programs\" ; if ($?)
{ gcc q3.c -o q3 } ; if ($?) { .\q3 }
enter the number of elements
4
enter the elements
5 3 8 6
elements before sorting
5
3
8
6
0
elements after sorting
3
5
6
8
```

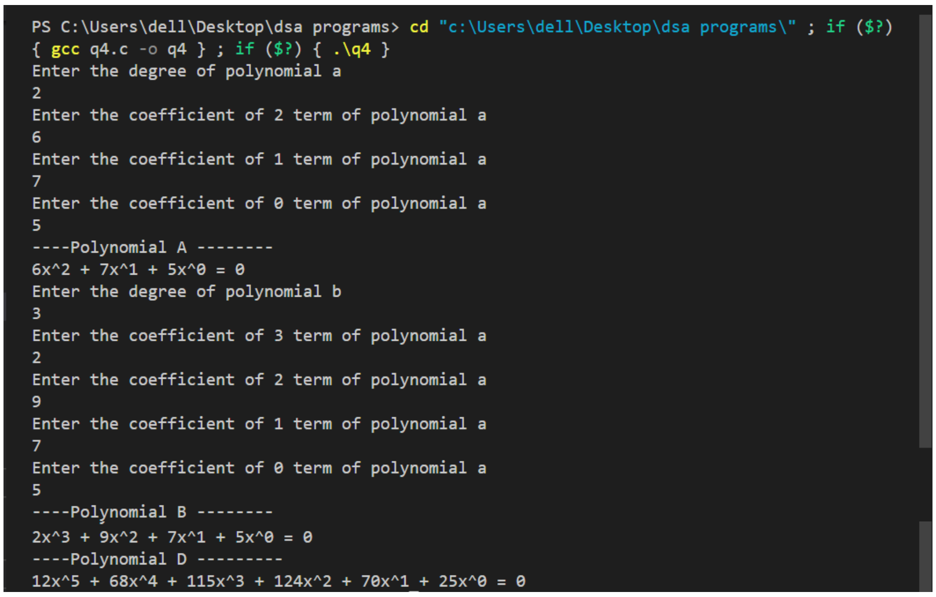
## 4) Develop a C program to multiply two polynomials A(x) and B(x) and store the resultant in C(x).

<u>Program:</u>

```
#include<stdio.h>

#include<stdlib.h>

#define MAX_TERMS 100

typedef struct

{

int coef;

int expon;

}polynomial;

polynomial terms[MAX_TERMS];

int avail = 0;

void pmul(int, int, int, int, int *, int * );

void simplifyPoly(int *, int * );

int main()

{ int startA, startB, finishA, finishB, x, y;

int *startD, *finishD;

startD = &x;

finishD = &y;

int cf, i, degree1, degree2;

// Reading Polynomial A
```

```
printf("Enter the degree of polynomial a\n");

scanf("%d", &degree1);

startA = avail;

for(; degree1>=0; degree1--)

{

printf("Enter the coefficient of %d term of polynomial a\n",

degree1);

scanf("%d", &cf);

if(cf != 0)

{

terms[avail].coef = cf;

terms[avail++].expon = degree1;

}

}

finishA = avail-1;

printf("----Polynomial A --------\n");

for(i=startA; i<finishA; i++)

printf("%dx^%d + ", terms[i].coef, terms[i].expon);

printf("%dx^%d = 0\n",terms[i].coef, terms[i].expon);

//Reading Polynomial B

printf("Enter the degree of polynomial b\n");

scanf("%d", &degree2);

startB = avail;

for(; degree2>=0; degree2--)

{

printf("Enter the coefficient of %d term of polynomial a\n",

degree2);

scanf("%d", &cf);
```

```
if(cf)
{
terms[avail].coef = cf;
terms[avail++].expon = degree2;
}
}
finishB = avail-1;
printf("----Polynomial B --------\n");
for(i=startB; i<finishB; i++)
printf("%dx^%d + ", terms[i].coef, terms[i].expon);
printf("%dx^%d = 0\n",terms[i].coef, terms[i].expon);
// Multiplied polynomial
pmul(startA, finishA, startB, finishB, startD, finishD);
printf("----Polynomial D ---------\n");
for(i=*startD ;i<*finishD; i++)
printf("%dx^%d + ", terms[i].coef, terms[i].expon);
printf("%dx^%d = 0\n", terms[i].coef, terms[i].expon);
return 0;
}
void pmul(int startA, int finishA, int startB, int finishB, int
*startD, int *finishD)
{
*startD = avail;
for(int i=startA; i<=finishA; i++)
{
for(int j=startB; j<=finishB; j++)
{
terms[avail].coef = terms[i].coef * terms[j].coef;
terms[avail++].expon = terms[i].expon + terms[j].expon;

}

}

*finishD = avail-1;

simplifyPoly(startD, finishD);

}

void simplifyPoly(int *startD, int *finishD)

{

for(int i=*startD; i<=*finishD; i++)

for(int j=i+1; j<=*finishD; j++)

if(terms[i].expon == terms[j].expon)

{

terms[i].coef = terms[i].coef+terms[j].coef;

for(int k=j; k<=*finishD; k++)

terms[k] = terms[k+1];

*finishD = *finishD - 1;

}

}
```



```
PS C:\Users\dell\Desktop\dsa programs> cd "c:\Users\dell\Desktop\dsa programs\" ; if ($?)
{ gcc q4.c -o q4 } ; if ($?) { .\q4 }
Enter the degree of polynomial a
2
Enter the coefficient of 2 term of polynomial a
6
Enter the coefficient of 1 term of polynomial a
7
Enter the coefficient of 0 term of polynomial a
5
----Polynomial A --------
6x^2 + 7x^1 + 5x^0 = 0
Enter the degree of polynomial b
3
Enter the coefficient of 3 term of polynomial a
2
Enter the coefficient of 2 term of polynomial a
9
Enter the coefficient of 1 term of polynomial a
7
Enter the coefficient of 0 term of polynomial a
5
----Polynomial B --------
2x^3 + 9x^2 + 7x^1 + 5x^0 = 0
----Polynomial D ---------
12x^5 + 68x^4 + 115x^3 + 124x^2 + 70x^1 + 25x^0 = 0
```

**5) Develop a C program to perform the following operations on sparse matrix using arrays and functions.**

**i) To add 2 sparse matrices**

Program:

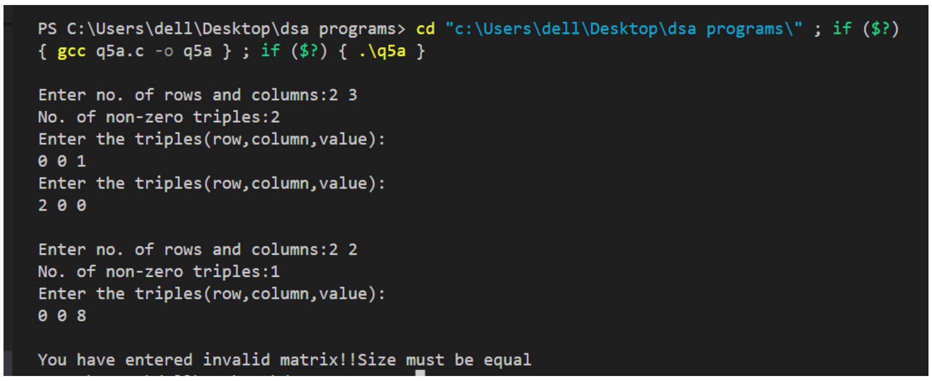
```
#include<stdio.h>

#include<stdlib.h>

#define MAX 20

void printsparse(int b[MAX][3]);

void readsparse(int b[MAX][3]);

void addsparse(int b1[MAX][3],int

b2[MAX][3],int b3[MAX][3]);

void main()

{

int b1[MAX][3],b2[MAX][3],b3[MAX][3];

readsparse(b1);

readsparse(b2);

addsparse(b1,b2,b3);

printsparse(b3);

}

void readsparse(int b[MAX][3])

{

int i,t,m,n;

printf("\nEnter no. of rows and columns:");

scanf("%d%d",&m,&n);

printf("No. of non-zero triples:");

scanf("%d",&t);

b[0][0]=m;

b[0][1]=n;
```

```
b[0][2]=t;
for(i=1;i<=t;i++)
{
printf("Enter the triples(row,column,value):\n");

scanf("%d%d%d",&b[i][0],&b[i][1],&b[i][2]);
}
}
void addsparse(int b1[MAX][3],int
b2[MAX][3],int b3[MAX][3])
{
int t1,t2,i,j,k;
if(b1[0][0]!=b2[0][0]||b1[0][1]!=b2[0][1])
{
printf("\nYou have entered invalid matrix!!Size must be
equal\n");
exit(0);
}
t1=b1[0][2];
t2=b2[0][2];
i=j=k=0;
b3[0][0]=b1[0][0];
b3[0][1]=b1[0][1];
while(i<=t1&&j<=t2)
{
if(b1[i][0]<b2[j][0])
{
b3[k][0]=b1[i][0];
```

```
b3[k][1]=b1[i][1];
b3[k][2]=b1[i][2];
k++;
i++;
}
else if(b2[j][0]<b1[i][0])
{
b3[k][0]=b2[j][0];
b3[k][1]=b2[j][1];
b3[k][2]=b2[j][2];
k++;
j++;
}
else if(b1[i][1]<b2[j][1])
{
b3[k][0]=b1[i][0];
b3[k][1]=b1[i][1];
b3[k][2]=b1[i][2];
k++;
i++;
}
else if(b2[j][1]<b1[i][1])
{
b3[k][0]=b2[j][0];
b3[k][1]=b2[j][1];
b3[k][2]=b2[j][2];
k++;
j++;
```

```
}
else
{
b3[k][0]=b1[i][0]; //row and column numbers are equal
b3[k][1]=b1[i][1];
b3[k][2]=b1[i][2]+b2[j][2];
k++;
i++;
j++;
}
}
while(i<=t1) //copy remaining terms from b1
{
b3[k][0]=b1[i][0];
b3[k][1]=b1[i][1];
b3[k][2]=b1[i][2];
i++;
k++;
}
while(j<=t2) //copy remaining terms from b2
{
b3[k][0]=b2[j][0];
b3[k][1]=b1[j][1];
b3[k][2]=b1[j][2];
j++;
k++;
}
b3[0][2]=k-1; //set number of terms in b3
```

```
}
void printsparse(int b[MAX][3])
{
int i,t;
t=b[0][2];
printf("nrowtcolumntvalue");
for(i=1;i<=t;i++)
{
printf("n%dt%dt%d",b[i][0],b[i][1],b[i][2]);
}
}
```



```
PS C:\Users\dell\Desktop\dsa programs> cd "c:\Users\dell\Desktop\dsa programs\" ; if ($?)
{ gcc q5a.c -o q5a } ; if ($?) { .\q5a }

Enter no. of rows and columns:2 3
No. of non-zero triples:2
Enter the triples(row,column,value):
0 0 1
Enter the triples(row,column,value):
2 0 0

Enter no. of rows and columns:2 2
No. of non-zero triples:1
Enter the triples(row,column,value):
0 0 8

You have entered invalid matrix!!Size must be equal
```

## ii) To multiply 2 sparse matrices.

<u>Program:</u>

```
#include <stdio.h>
void print(int k[3][100], int count)
{
int i, j;
//printf("%d",count);
for (j = 0; j < 3; j++)
{
for (i = 0; i < count; i++)
{
```

```
printf("%d ", k[j][i]);
}
printf("\n"); }
}
void swap(int *a, int *b)
{
int temp;
temp = *a;
*a = *b;
*b = temp;
}
void sort(int k[3][100], int count)
{
int i, j;
for (i = 0; i < count; i++)
{
for (j = 0; j < count - i - 1; j++)
{
if (k[0][j] > k[0][j + 1])
{
swap(&k[0][j], &k[0][j + 1]);
swap(&k[1][j], &k[1][j + 1]);
swap(&k[2][j], &k[2][j + 1]);
}
else if (k[0][j] == k[0][j + 1])
{
if (k[1][j] > k[1][j + 1])
{
```

```
swap(&k[0][j], &k[0][j + 1]);

swap(&k[1][j], &k[1][j + 1]);

swap(&k[2][j], &k[2][j + 1]);

}

}

}

}

}

void transpose(int k[3][100], int count)

{ int i, j, temp;

printf("\n");

for (j = 0; j < count; j++)

{

swap(&k[0][j], &k[1][j]);

}

sort(k, count);

// print(k,count);

}

void multiply(int k[3][100], int count, int r1, int c1)

{

int b[20][20], l[3][100], i, j, r2, c2, size = 0, kpos, lpos,

result[3][100], r, c, tempk, templ, sum, rcount = 0;

printf("Enter no of rows");

scanf("%d", &r2);

printf("Enter no of columns");

scanf("%d", &c2);

for (i = 0; i < r2; i++)

{
```

```
for (j = 0; j < c2; j++)
{
scanf("%d", &b[i][j]);
if (b[i][j] != 0)
{
l[0][size] = i;
l[1][size] = j;
l[2][size] = b[i][j];
size++;
}
}
}
if (c1 != r2)
{
printf("Not possible");
return;
}
transpose(l, size);
for (kpos = 0; kpos < count;)
{
r = k[0][kpos];
for (lpos = 0; lpos < size;)
{
c = l[0][lpos];
tempk = kpos;
templ = lpos;
sum = 0;
while (tempk < count && k[0][tempk] == r && templ < size && l[0][templ] == c)
```

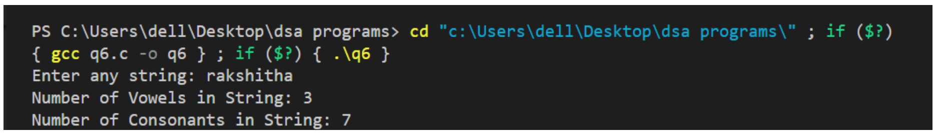
```
{
if (k[1][tempk] < l[1][templ])
{
tempk++;
}
else if (l[1][templ] > k[1][tempk])
{
templ++;
}
else
{
sum += k[2][tempk++] * l[2][templ++];
}
}
if (sum != 0)
{
result[0][rcount] = r;
result[1][rcount] = c;
result[2][rcount] = sum;
rcount++;
}
while (lpos < size && l[0][lpos] == c)
{
lpos++;
}
}
while (kpos < count && k[0][kpos] == r)
{
```

```
kpos++;
}
}
print(result, rcount);
}
int main()
{
int a[20][20], k[3][100], i, j, m, n, count = 0;
printf("Enter no of rows");
scanf("%d", &m);
printf("Enter no of coloumns");
scanf("%d", &n);
for (i = 0; i < m; i++)
{
for (j = 0; j < n; j++)
{
scanf("%d", &a[i][j]);
if (a[i][j])
{
k[0][count] = i;
k[1][count] = j;
k[2][count] = a[i][j];
count++;
}
}
}
multiply(k, count, m, n);
}
```

## 6) Develop a C program to count vowels and consonants in a string using pointer.

Program:

```
#include <stdio.h>

int main()

{

char str[100];

char *p;

int vCount=0,cCount=0;

printf("Enter any string: ");

fgets(str, 100, stdin);

p=str;

while(*p!='\0')

{

if(*p=='A' ||*p=='E' ||*p=='I' ||*p=='O' ||*p=='U'

||*p=='a' ||*p=='e' ||*p=='i' ||*p=='o' ||*p=='u')

vCount++;

else

cCount++;

p++;

}

printf("Number of Vowels in String: %d\n",vCount);

printf("Number of Consonants in String: %d",cCount);

return 0;

}
```



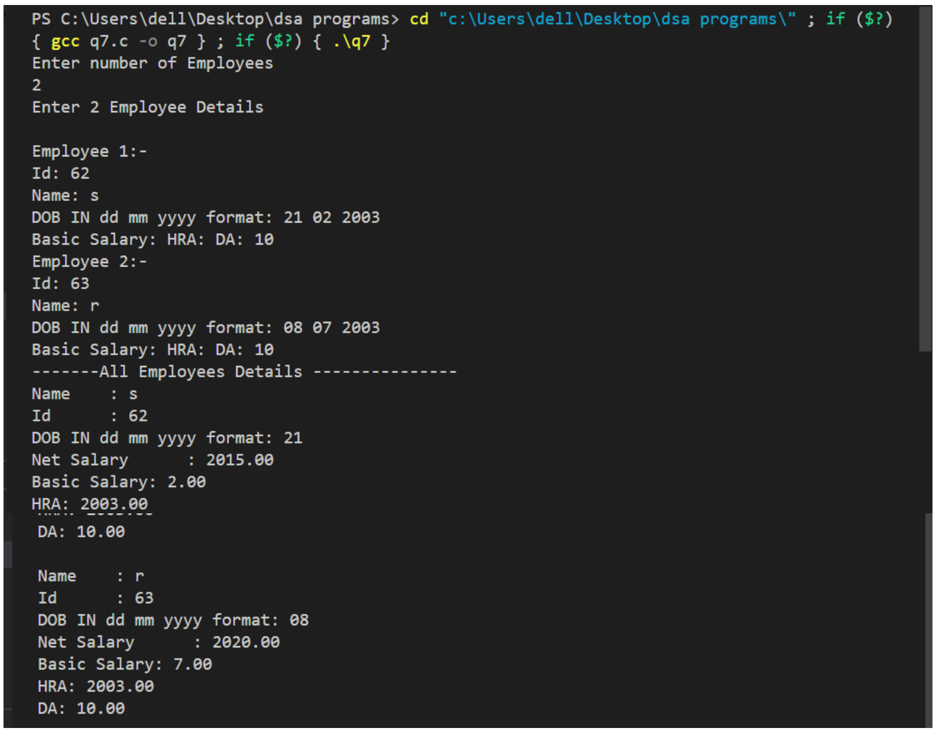
```
PS C:\Users\dell\Desktop\dsa programs> cd "c:\Users\dell\Desktop\dsa programs\" ; if ($?)
{ gcc q6.c -o q6 } ; if ($?) { .\q6 }
Enter any string: rakshitha
Number of Vowels in String: 3
Number of Consonants in String: 7
```

**7) Develop a C program to read, display and calculate total salary of N employees.The details are EMPID, ENAME, DOB (Day, Month, Year) and Salary (Basic,HRA,DA). Use appropriate structures and nested structures.**

Program:

```
#include <stdio.h>

#include <stdlib.h>

struct Salary

{

double basic_salary;

double net_salary;

double da;

double hra;

};

typedef struct emp

{ char name[30];

char id[20];

char dob[20];

struct Salary s;

} Employee;

int main()

{

int n;

Employee employees[10];

printf("Enter number of Employees\n");

scanf("%d", &n);

// Taking each employee detail as input

printf("Enter %d Employee Details \n \n", n);

for (int i = 0; i < n; i++)

{
```

```
printf("Employee %d:- \n", (i + 1));

printf("Id: ");

scanf("%s", &employees[i].id);

printf("Name: ");

scanf("%s", employees[i].name);

printf("DOB IN dd mm yyyy format: ");

scanf("%s", employees[i].dob);

printf("Basic Salary: ");

scanf("%lf", &employees[i].s.basic_salary);

printf("HRA: ");

scanf("%lf", &employees[i].s.hra);

printf("DA: ");

scanf("%lf", &employees[i].s.da);

employees[i].s.net_salary = employees[i].s.basic_salary + employees[i].s.hra +
employees[i].s.da;

}

// Displaying Employee details

printf("-------All Employees Details ---------------\n");

for (int i = 0; i < n; i++)

{ printf("Name \t: ");

printf("%s \n", employees[i].name);

printf("Id \t: ");

printf("%s \n", employees[i].id);

printf("DOB IN dd mm yyyy format: %s \n",

employees[i].dob);

printf("Net Salary \t: ");

printf("%.2lf \n", employees[i].s.net_salary);

printf("Basic Salary: %.2lf\n", employees[i].s.basic_salary);

printf("HRA: %.2lf\n", employees[i].s.hra);
printf("DA: %.2lf\n", employees[i].s.da);

printf("\n");

}

return 0;

}
```



```
PS C:\Users\dell\Desktop\dsa programs> cd "c:\Users\dell\Desktop\dsa programs\" ; if ($?)
{ gcc q7.c -o q7 } ; if ($?) { .\q7 }
Enter number of Employees
2
Enter 2 Employee Details

Employee 1:-
Id: 62
Name: s
DOB IN dd mm yyyy format: 21 02 2003
Basic Salary: HRA: DA: 10
Employee 2:-
Id: 63
Name: r
DOB IN dd mm yyyy format: 08 07 2003
Basic Salary: HRA: DA: 10
-------All Employees Details ---------------
Name     : s
Id       : 62
DOB IN dd mm yyyy format: 21
Net Salary       : 2015.00
Basic Salary: 2.00
HRA: 2003.00
 DA: 10.00

 Name     : r
 Id       : 63
 DOB IN dd mm yyyy format: 08
 Net Salary       : 2020.00
 Basic Salary: 7.00
 HRA: 2003.00
 DA: 10.00
```

**8) Develop a C program to define a structure EMPLOYEE with its members name,Id and salary. Search a given name in the record using binary search and print the results with suitable message.**

Program :

```
#include<stdio.h>

#include<string.h>

#define MAX 50

typedef struct{

char name[30];

char id[30];

float salary;

}EMP;
```

```
EMP e[MAX],e1[MAX];

void getData(int n);

void sortData(int n);

int binSearch(int n);

void displayData(int pos);

int main(){

int n,pos;

printf("Enter the no of emp \n");

scanf("%d",&n);

getData(n);

sortData(n);

pos=binSearch(n);

displayData(pos);

return 0;

}

void getData(int n){

int i;

for(i=0;i<n;i++){

printf("Enter the name,id and salary of emp %d \n",i+1);

scanf("%s%s%f",e[i].name,e[i].id,&e[i].salary);

}

}

void sortData(int n){

int i,j;

EMP temp;

for(i=0;i<n;i++){

for(j=0;j<n-i-1;j++){

if(strcmp(e[j].name,e[j+1].name)>0){
```

```
temp= e[j];
e[j]=e[j+1];
e[j+1]=temp;
}
}
}
}
void displayData(int pos){
if(pos>=0){
printf("The employee was found at position %d \n",pos+1);
printf("________________________ \n");
printf("NAME \t EMPID \t SALARY\n");
printf("%s %s
%.2f\n",e[pos].name,e[pos].id,e[pos].salary);}
else
printf("Employee not found\n");
}
int binSearch(int n){
int i,mid,beg=0,end=n-1;
char ele[30];
printf("Enter the name to be searched \n");
scanf("%s",ele);
while(beg<=end){
mid=(beg+end)/2;
if(strcmp(e[mid].name,ele)==0){
return mid;
}
else if(strcmp(ele,e[mid].name)>0){
```

beg=mid+1;

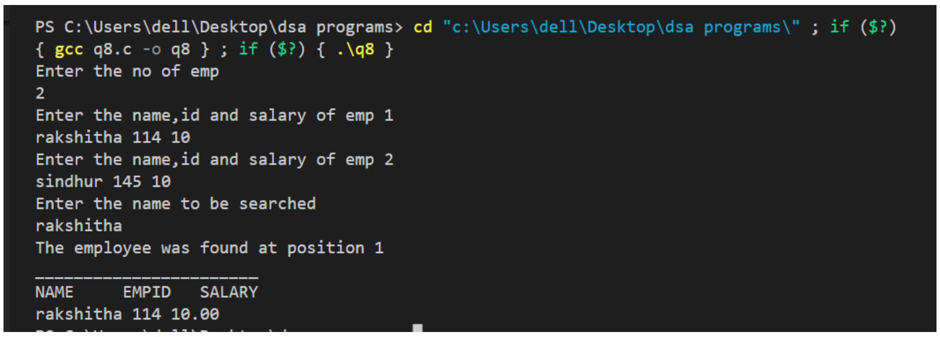
}

else

end=mid-1;

}

return -1;}



```
PS C:\Users\dell\Desktop\dsa programs> cd "c:\Users\dell\Desktop\dsa programs\" ; if ($?)
{ gcc q8.c -o q8 } ; if ($?) { .\q8 }
Enter the no of emp
2
Enter the name,id and salary of emp 1
rakshitha 114 10
Enter the name,id and salary of emp 2
sindhur 145 10
Enter the name to be searched
rakshitha
The employee was found at position 1
________________________
NAME     EMPID   SALARY
rakshitha 114 10.00
```

**9) You are given a stack of N integers. In one operation, you can either pop an element from the stack or push any popped element into the stack. You need to maximize the top element of the stack after performing exactly K operations. If the stack becomes empty after performing K operations and there is no other way for the stack to be non-empty, print -1.**

<u>Program:</u>

#include <stdio.h>

int max_element(int);

int a[50];

int main()

{

int n, k, i, tos;

printf("Enter number of elements: ");

scanf("%d", &n);

printf("Enter number of operations (push or pop): ");

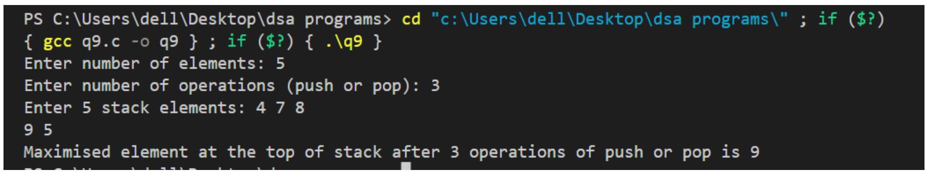
scanf("%d", &k);

printf("Enter %d stack elements: ", n);

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

```
scanf("%d", &a[i]);

if ((n == 1) && (k % 2 == 1))

printf("-1");

else if (k == n)

tos = max_element(n - 1);

else if (k < n)

{

int m = max_element(k - 1);

if (m > a[k])

tos = m;

else

tos = a[k];

}

else

tos = max_element(n);

printf("Maximised element at the top of stack after %d operations

of push or pop is %d\n", k, tos);

return 0;

}

int max_element(int n)

{

int max = a[0];

for (int i = 0; i < n; i++)

{

if (a[i] > max)

max = a[i];

}

return max;
```

```
}
```



```
PS C:\Users\dell\Desktop\dsa programs> cd "c:\Users\dell\Desktop\dsa programs\" ; if ($?)
{ gcc q9.c -o q9 } ; if ($?) { .\q9 }
Enter number of elements: 5
Enter number of operations (push or pop): 3
Enter 5 stack elements: 4 7 8
9 5
Maximised element at the top of stack after 3 operations of push or pop is 9
```

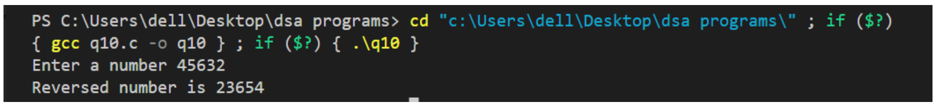
## 10) Develop a C program to reverse a given integer number using stack.

<u>Program:</u>

```
#include <stdio.h>

#define MAXSIZE 10

struct Stack

{

int top;

int array[MAXSIZE];

} st;

void push(int);

int pop();

int reverse_number(int);

int main()

{

st.top = -1;

printf("Enter a number ");

int number;

scanf("%d", &number);

printf("Reversed number is %d", reverse_number(number));

return 0;}

void push(int x)

{

if (st.top == MAXSIZE - 1)

{
```

```
printf("\nOverflow!!");
}
else
{
st.top = st.top + 1;
st.array[st.top] = x;
}
}
int pop()
{
if (st.top == -1)
{
printf("\nUnderflow!!");
}
else
{
int ele = st.array[st.top];
st.top = st.top - 1;
return ele;
}
}
int reverse_number(int number)
{
while (number != 0)
{
push((number % 10));
number = number / 10;
}
```

```
int reverse = 0;

int i = 1;

while (st.top != -1)

{

reverse = reverse + (st.array[st.top] * i);

pop();

i = i * 10;

}

return reverse;

}
```



```
PS C:\Users\dell\Desktop\dsa programs> cd "c:\Users\dell\Desktop\dsa programs\" ; if ($?)
{ gcc q10.c -o q10 } ; if ($?) { .\q10 }
Enter a number 45632
Reversed number is 23654
```