SSN COLLEGE OF ENGINEERING (Autonomous)
(Affiliated to Anna University, Chennai)
DEPARTMENT OF CSE
UCS 1211 PROGRAMMING IN C LABORATORY
A5: Pointers in C

Register Number:185001050
Name : M.Gayathri
Class : CSE-A

1.Program Name: Frequency of a word in a line of text

Program:

```
#include<stdio.h>
#include<string.h>
void main()
{
char str[81],sub[81],temp[81];
printf("\nEnter a string:");
scanf(" %[^\n]",str);
printf("\nenter the sub string:");
scanf(" %[^\n]",sub);
int i,k=0,count=0;
for(i=0;i<=strlen(str);i++)
        {
        if(*(str+i)!=' '&&*(str+i)!='\0')
                *(temp+k++)=*(str+i);
        else
                {
                *(temp+k)='\0';
                if(strcmp(sub,temp)==0)
                        count++;
                k=0;
                }

        }
printf("The no of times \'%s\' occurs is %d \n",sub,count);
}
```

Output:

```
root@jtl-13:~/A5# gcc freq.c -o freq
root@jtl-13:~/A5# ./freq
Enter a string:fun is fun to have fun with friends
enter the sub string:fun
The no of times 'fun' occurs is 3
```

2.Program Name: To parse the text to separate the tokens

Program:

```
#include <stdio.h>
#include <string.h>
void main()
{
	char str[10][200];
	int row=0,i,l,k=0,j=0;
	char *pointer[10];
	printf("enter multiple lines of text and enter END at the end \n");
	while(1)
	{
		scanf(" %[^\n]",str[row]);
		l=strlen(str[row]);
		pointer[row]=str[row];
		if (str[row][l-3]=='E' && str[row][l-2]=='N' && str[row][l-1]=='D')
		{
			str[row][l-4]='\0';
			break;
		}
		row++;
	}

	for(i=0;i<=row;i++)
	{
		j=0;
		while(*(pointer[i]+j)!='\0')
		{
			if (*(pointer[i]+j)==' ')
			{
				k++;
			}
			j++;
```

```
                }
                k++;
        }

        printf("\nnumber of tokens : %d",k);
}
```

Output:

```
root@jtl-13:~/A5# gcc end.c -o end
root@jtl-13:~/A5# ./end
enter multiple lines of text and enter END at the end
types of comment statement
single line comment
multiple line commentEND

number of tokens : 10
```

3.Program Name: To create table c is the larger of the corresponding elements in tables a and b

Program:

```
#include <stdio.h>
#include<stdlib.h>

#define rmax 20

void input(int *a[rmax], int r, int nc)
{
    int row,col;
    printf("\nEnter the elements:\n");
    for(row=0;row<r;row++)
    {
        printf("\nRow : %2d:\n",row+1);
        for(col=0;col<nc;col++)
            scanf("%d",(*(a+row)+col));
        printf("\n");
    }
```

```
}

void larger(int *a[rmax], int *b[rmax], int *c[rmax], int r, int nc)
{
   int row,col;

   for(row=0;row<r;row++)
        for(col=0;col<nc;col++)
            if(*(*(a+row)+col)>*(*(b+row)+col))
                        *(*(c+row)+col)=*(*(a+row)+col);
                 else
                        *(*(c+row)+col)=*(*(b+row)+col);
}

void display(int *c[rmax],int r, int nc)
{
   int row,col;
   for(row=0;row<r;row++)
        {
        printf("\n");
        for(col=0;col<nc;col++)
                        printf("%4d",*(*(c+row)+col));
        }
        printf("\n");
}


void main()
{

   int r,nc,row;
   int *a[100],*b[100],*c[100];



   printf("\nEnter number of rows:");
   scanf("%d",&r);
   printf("\nEnter number of columns:");
   scanf("%d",&nc);




   for(row=0;row<r;row++)
   {
       a[row]=(int *) malloc (nc*sizeof(int));
       b[row]=(int *) malloc (nc*sizeof(int));
       c[row]=(int *) malloc (nc*sizeof(int));
   }

   input(a,r,nc);
   input(b,r,nc);
   larger(a,b,c,r,nc);
   display(c,r,nc);

}
```

Output:

```
root@jtl-13:~/A5# gcc larger.c -o larger
root@jtl-13:~/A5# ./larger

Enter number of rows:2

Enter number of columns:3

Enter the elements:

Row :  1:
4 7 9

Row :  2:
5 19 -5

Enter the elements:

Row :  1:
5 6 10

Row :  2:
3 12 -3


   5   7  10
   5  19  -3
```

4.Program Name: To determine the number of days between two dates

Program:

```
#include<stdio.h>
#include<stdlib.h>
long nodays(int *date)
{
	long ndays;
	int yy=*(date+2);
	ndays=(long)(30.42*(*(date+1)-1)+*date);
	if (*(date+1)==2)
		ndays+=1;
	else if(*(date+1)>2 && *(date+1)<8)
		ndays-=1;
	yy-=1900;
	if(yy%4==0 &&*(date+1)>2)
		ndays+=1;
	ndays+=(yy/4)*1461;
	ndays+=(yy%4)*365+1;
	if(ndays>59)
		ndays-=1;
	//printf("%ld\n",ndays%7);
	return ndays;
}
void main()
{
	int *date1,*date2;
	long n1,n2,diff;
	date1=(int*)malloc(3*sizeof(int));
	date2=(int*)malloc(3*sizeof(int));
	printf("\n\nFINDING DIFFERENCE BETWEEN TWO DAYS");
	printf("\n\nEnter a date(mm dd yyyy): ");
	scanf("%2d %2d %4d",(date1+1),(date1),(date1+2));
	n1=nodays(date1);
	printf("\nEnter another date(mm dd yyyy): ");
	scanf("%2d %2d %4d",(date2+1),(date2),(date2+2));
	n2=nodays(date2);
	diff=n1-n2;
	if (diff>0)
		printf("\n\n%d %d %d comes after %ld days of %d %d
%d",*(date1+1),*(date1),*(date1+2),diff,*(date2+1),*(date2),*(date2+2));
	else
```

```
                    printf("\n\n%d %d %d comes after %ld days of %d %d
%d\n",*(date2+1),*(date2),*(date2+2),-diff,*(date1+1),*(date1),*(date1+2));

}
```

Output:

```
root@jtl-13:~/A5# gcc date.c -o date
root@jtl-13:~/A5# ./date

FINDING DIFFERENCE BETWEEN TWO DAYS

Enter a date(mm dd yyyy): 1 5 2001

Enter another date(mm dd yyyy): 3 26 2019

3 26 2019 comes after 6654 days of 1 5 2001
```

5.Program Name: To determine compound interest table

Program:

```
#include<stdio.h>
#include<math.h>

void table(double (*fn)(double,int,float),double a,int m, float n)
{
   double i,f;
   //printf();
   for(int c=5;c<=10;++c)
   {
      i=0.01*c;
      f=a*(*fn)(i,m,n);
      printf("  %.2f",f);
   }

}

double ci1(double i,int m,float n)
{
   double x;
   x=12*(pow((1+i/m),m*n)-1)/i;
   return x;
}
```

```
double ci2(double i,int m,float n)
{
   double x;
   x=(pow((1+i/m),m*n)-1)/(pow((1+i/m),m/12)-1);
   return x;
}
double ci3(double i,int m,float n)
{
   double x;
   x=(exp(i*n)-1)/(exp(i/12)-1);
   return x;
}
void main()
{
   float n;
   int m=1;
   double a;
   printf("\n\nEnter Principal amount:  ");
   scanf("%lf",&a);
   printf("\n\nEnter number of years: ");
   scanf("%f",&n);
   printf("\n\nPrincipal amount:  %.2lf",a);
   printf("\n\nNumber of years: %.2f",n);
   printf("\n\nInterest rate:   ");
   printf("5%%\t    6%%\t\t7%%\t   8%%\t\t9%%\t    10%% ");
   printf("\n\nFrequency of\ncompounding\n\nAnnually     ");
   table(ci1,a,m,n);
   m=2;
   printf("\n\nSemiannual   ");
   table(ci1,a,m,n);
   m=4;
   printf("\n\nQuarterly    ");
   table(ci1,a,m,n);
   m=12;
   printf("\n\nMonthly      ");
   table(ci1,a,m,n);
   m=360;
   printf("\n\nDaily        ");
   table(ci2,a,m,n);
   m=0;
   printf("\n\nContinuously ");
   table(ci3,a,m,n);
}
```

Output:

```
root@jtl-13:~/A5# gcc interest.c -o interest -lm
root@jtl-13:~/A5# ./interest

Enter Principal amount:  5000

Enter number of years: 2


Principal amount:  5000.00

Number of years: 2.00

Interest rate:       5%          6%          7%          8%          9%          10%

Frequency of
compounding

Annually         123000.00  123600.00  124200.00  124800.00  125400.00  126000.00

Semiannual     124575.47  125508.81  126448.29  127393.92  128345.73  129303.75

Quarterly       125383.32  126492.59  127612.96  128744.54  129887.43  131041.74

Monthly         125929.60  127159.78  128405.16  129665.95  130942.35  132234.58

Daily             125941.93  127177.75  128429.94  129698.72  130984.36  132287.10

Continuously   125942.36  127178.38  128430.79  129699.86  130985.81  132288.92
```

5.a.Program Name: To determine compound interest table

Program:

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<ctype.h>
void table(double (*p)(double i,int m,double n), double a,int m,double i,int t);
double percmpd(double i,int m,double n);
double dailycmpd(double i,int m,double n);
double contcmpd(double i,int m,double n);
double *val[6];

void main()
{
int m[6]={1,2,4,12,360,0},t,k;
double i,a;
char choice;
printf("\nMonthly Amount : ");
scanf("%lf",&a);
printf("Interest : ");
scanf("%lf",&i);
i*=0.01;
printf("\nFrequency of compounding :\n");
for(t=0;t<6;t++)
        val[t]=(double *) malloc (50 * sizeof(double));
for(t=0;t<6;t++)
        {
        if(t<4) table(percmpd,a,m[t],i,t);
        else if(t==4) table(dailycmpd,a,m[t],i,t);
        else table(contcmpd,a,m[t],i,t);

        }
printf("Terms   Annualy   Semiannualy   Quaterly    Monthly   Daily
        Continuous\n");
printf("(years)\n");
for(t=0;t<50;t++)
        {
        printf("%d   \t",t+1);
        for(k=0;k<6;k++)
                {printf("%.1lf",*(val[k]+t));
                 printf("\t");}
        printf("\n");
        }

}

void table(double (*p)(double i,int m,double n), double a,int m,double i,int t)
{
int j;
double f;
for(j=1;j<=50;j++)
        {
        f=a*(*p)(i,m,j);
        *(val[t]+j-1)=f;
        }

}


double percmpd(double i,int m,double n)
{
   double x;
   x=12*(pow((1+i/m),m*n)-1)/i;
   return x;
}
double dailycmpd(double i,int m,double n)
{
   double x;
   x=(pow((1+i/m),m*n)-1)/(pow((1+i/m),m/12)-1);
   return x;
}
double contcmpd(double i,int m,double n)
{
   double x;
   x=(exp(i*n)-1)/(exp(i/12)-1);
   return x;
}
```

Output:

```
root@jtl-13:~/A5# gcc interestmod.c -o interestmod -lm
root@jtl-13:~/A5# ./interestmod

Monthly Amount : 2000
Interest : 3

Frequency of compounding :
Terms   Annualy    Semiannualy    Quaterly      Monthly       Daily      Continuous
(years)
1       24000.0    24180.0        24271.4       24332.8       24333.2    24333.2
2       48720.0    49090.8        49279.1       49405.6       49407.4    49407.4
3       74181.6    74754.6        75045.5       75241.1       75245.2    75245.3
4       100407.0   101194.1       101593.7      101862.4      101869.8   101870.0
5       127419.3   128432.7       128947.3      129293.4      129305.2   129305.6
6       155241.8   156494.5       157130.8      157558.8      157576.2   157576.8
```

7	183899.1	185404.6	186169.4	186683.8	186708.0	186708.9
8	213416.1	215188.4	216089.0	216694.8	216727.1	216728.2
9	243818.5	245872.5	246916.3	247618.5	247660.3	247661.7
10	275133.1	277484.0	278678.9	279482.8	279535.5	279537.3
11	307387.1	310051.0	311405.1	312316.3	312381.4	312383.7
12	340608.7	343602.2	345124.3	346148.5	346227.6	346230.4
13	374827.0	378167.6	379866.4	381009.7	381104.6	381107.9
14	410071.8	413777.7	415662.6	416931.3	417043.6	417047.5
15	446373.9	450464.2	452544.8	453945.4	454077.1	454081.7
16	483765.2	488259.5	490546.0	492085.3	492238.4	492243.7
17	522278.1	527197.1	529700.1	531385.3	531561.8	531567.9
18	561946.4	567311.6	570042.2	571880.7	572082.8	572089.8
19	602804.8	608638.6	611608.1	613607.8	613837.7	613845.7
20	644889.0	651214.7	654435.2	656604.0	656864.3	656873.3
21	688235.7	695077.7	698561.6	700908.0	701201.1	701211.2
22	732882.7	740266.4	744026.7	746559.6	746888.1	746899.5
23	778869.2	786821.0	790871.2	793599.6	793966.5	793979.1
24	826235.3	834782.6	839137.0	842070.5	842478.5	842492.6
25	875022.3	884193.9	888867.1	892015.6	892467.9	892483.5
26	925273.0	935098.7	940105.9	943479.9	943979.6	943996.8
27	977031.2	987542.1	992899.3	996509.5	997060.0	997079.0
28	1030342.1	1041570.5	1047294.5	1051152.1	1051756.9	1051777.8
29	1085252.4	1097232.0	1103339.9	1107456.6	1108119.4	1108142.3
30	1141810.0	1154575.8	1161085.7	1165473.8	1166198.5	1166223.5
31	1200064.3	1213652.9	1220583.4	1225255.5	1226046.2	1226073.5
32	1260066.2	1274515.5	1281886.3	1286855.6	1287716.4	1287746.2
33	1321868.2	1337217.8	1345049.0	1350329.3	1351264.7	1351297.0
34	1385524.2	1401815.2	1410128.1	1415733.7	1416748.3	1416783.4
35	1451090.0	1468365.0	1477181.6	1483127.3	1484226.0	1484264.0
36	1518622.7	1536926.4	1546269.4	1552570.8	1553758.7	1553799.8
37	1588181.3	1607560.0	1617453.3	1624126.5	1625408.9	1625453.2
38	1659826.8	1680328.5	1690796.9	1697858.6	1699241.0	1699288.8
39	1733621.6	1755296.4	1766365.7	1773833.4	1775321.6	1775373.0
40	1809630.2	1832530.2	1844227.1	1852119.0	1853719.1	1853774.4
41	1887919.1	1912098.5	1924450.8	1932785.7	1934504.0	1934563.4
42	1968556.7	1994071.6	2007108.5	2015906.0	2017749.1	2017812.8
43	2051613.4	2078522.4	2092273.9	2101554.5	2103529.3	2103597.6
44	2137161.8	2165525.8	2180023.1	2189808.1	2191921.8	2191994.8
45	2225276.7	2255158.8	2270434.6	2280745.9	2283006.1	2283084.2
46	2316035.0	2347501.0	2363589.1	2374449.8	2376864.2	2376947.6
47	2409516.0	2442634.2	2459569.8	2471003.7	2473580.6	2473669.7
48	2505801.5	2540642.8	2558462.6	2570494.4	2573242.3	2573337.3
49	2604975.5	2641613.7	2660355.6	2673011.2	2675939.1	2676040.3

50	2707124.8	2745636.5	2765340.0	2778646.2	2781763.3	2781871.0