C Programming Assignment

1). Write a C program to find sum of proper division of a number using recursion.

```
#include<stdio.h>

int sumdiv(int num, int x);

int main()

{

    int num;

    printf("Enter a number:");

    scanf("%d",&num);

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

    printf("Sum of divisors=%d\n",sumdiv(num,num/2));

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

    return 0;

}

sumdiv(int num, int x)

{

    if(x==1)

    {

        printf("%d  \n",x);

         return 1;

    }

    if(num%x==0)

    {

         printf("%d + ",x);

         return x + sumdiv(num,x-1);

    }

    else

         return sumdiv(num,x-1); }
```

----

2). Write a C program to display a number in words using recursion.

```
#include <stdio.h>
void printWords(int n);
int main()
{
	int n ;
	printf("Enter any number:");
	scanf("%d",&n);
	printWords(n);
	return 0;
}
void printWords(int n)
{
	int m;
	if(n==0)
		return;

	printWords(n/10);
	m=n%10;
	switch(m)
	{
		case 0: printf("Zero");
			break;
		case 1: printf("One");
			break;
		case 2: printf("Two");
			break;
		case 3: printf("Three");
			break;
```

```
            case 4: printf("Four");
                   break;
            case 5: printf("Five");
                   break;
            case 6: printf("Six");
                   break;
            case 7: printf("Seven");
                   break;
            case 8: printf("Eight");
                   break;
            case 9: printf("Nine");
                   break;
       }
}
```

----

3). Write a C program to perform multiplication by Russian Peasant method.

```
#include<stdio.h>
int RussianPeasant(int a, int b);
int main()
{
       int a,b;
       printf("Enter 1st number:\n");
       scanf("%d",&a);
       printf("Enter 2nd number:\");
       scanf("%d",&b);
       printf("Product of two numbers is: ");
       printf("%d\n",RussianPeasant(a,b));
       return 0;
}
```

```
int RussianPeasant(int a, int b)
{
        if(a==0)
        return 0;
        if(a%2!=0)
        return b+RussianPeasant(a/2,b*2);
    return RussianPeasant(a/2,b*2);
}
```

----

4). Write a C program to calculate log to the base 2 and base n using recursion.

```
#include<stdio.h>
int logbase2(int num);
int logbasen(int num,int base);
int main()
{
        int num,base;
        printf("Enter a number for Base-2 logarithm:\n");
        scanf("%d",&num);
        printf("Value of log base2 of %d=%d\n",num,logbase2(num));
        printf("\nEnter any number:\n");
        scanf("%d",&num);
        printf("Enter base to %d:",num);
        scanf("%d",&base);
        printf("Value of log base %d of %d=%d\n",base,num,logbasen(num,base));
        return 0;
}
int logbase2(int num)
{
      if(num==1)
```

```
        return 0;
     return 1 + logbase2(num/2);
}
int logbasen(int num,int base)
{
     if(num<base)
        return 0;
    return 1 + logbasen(num/base,base);
}
```

----

5). Write a C program to calculate binomial coefficient using recursion.

```
#include<stdio.h>
int binomialCoef(int n,int m);
int main()
{
       int n,m;
       printf("Enter n and m:\n");
       scanf("%d%d",&n,&m);
       printf("Binomial coefficient\n",binomialCoef(n,m));
       printf("%d\n",binomialCoef(n,m));
       return 0;
}
int binomialCoef(int n,int m)
{
       if(m==0||m==n)
              return 1;
       return binomialCoef(n-1,m-1)+binomialCoef(n-1,m);
}
```

----

6). Write a C program to implement Ackermann Function using Recursion.

```
#include <stdio.h>
int ackermann(int m,int n)
{
    if (m == 0){
        return n+1;
    }
    else if((m > 0) && (n == 0)){
        return ackermann(m-1, 1);
    }
    else if((m > 0) && (n > 0)){
        return ackermann(m-1, ackermann(m, n-1));
    }
}
 int main()
 {
    int x;
    x= ackermann(1, 2);
    printf("%d", x);
    return 0;
}
```

----

7). Write a C program to find frequency of vowels in string using recursion.

```
#include<stdio.h>
#include<string.h>
int count_vowels(char *str);
int main()
{
        char str[100];
        printf("Enter a string:");
        gets(str);
        printf("Number of vowels in the string is %d\n",count_vowels(str));
        return 0;
}
int count_vowels(char *str)
{
   if(*str == '\0')
            return 0;
   switch(*str)
   {
          case 'A': case 'a':
            case 'E': case 'e':
            case 'I': case 'i':
            case 'O': case 'o':
            case 'U': case 'u':
          return 1 + count_vowels(str+1);
            default:
          return count_vowels(str+1);
   }
}
```

---

8). Write a C program to replace a character in a string by another character.

```
#include <string.h>
#include<stdio.h>
void replacechar(char *s,char c1,char c2)
{
          int i=0;
          for(i=0;s[i];i++)
```

```
            {
                        if(s[i]==c1)
                        {
                          s[i]=c2;
                }
   }
}
int main()
{
   char str[100],c1,c2;
   printf("Enter  the string :\n ");
   gets(str);
   printf("Enter a character to replace: \n");
   c1=getchar();
   getchar();
   printf("Enter character to replace  with  %c :\n ",c1);
   c2=getchar();
   printf("Before replacing:%s\n",str);
   replacechar(str,c1,c2);
   printf("After replacing:%s\n",str);
            return 0;
 }
```

----

9). Write a C program to print pyramid of numbers using recursion.

```
#include<stdio.h>
void first(int n);
void second(int n);
void third(int n);
int main( )
```

```
{
   int n;
   printf("Enter how many lines u want to print ? ");
   scanf("%d",&n);
   printf("------------ First Pattern ------------\n");
   first(n);
   printf("------------ Second Pattern ----------- \n");
   func2(n);
   printf("------------ Third Pattern ----------- \n");
   func3(n);
   return 0;
}
void first(int n)
{
      int i;
      if(n==0)
            return;
      else
      {
            first(n-1);
            for(i=1; i<= n; i++)
                  printf("%d ",i);
             printf("\n");
      }
}
void second(int n)
{
      int i;
      if(n==0)
```

```
        return;
    else
    {
        for(i=1; i<=n; i++)
            printf("%d ",i);
        printf("\n");
        second(n-1);
    }
}
void third(int n)
{
    int i;
    if(n==0)
        return;
    else
    {
        for(i=n; i>=1; i--)
            printf("%d ",i);
        printf("\n");
        third(n-1);
    }
}
```

----

10). Write a C program to convert numbers to string using recursion.

```
#include<stdio.h>
void numtostr(long int n, char str[]);
int main()
{
    long int num;
    printf("Enter any number to convert to String:\n");
    scanf("%ld",&num);
    char str[30];
    numtostr(num, str);
    printf("\nAfter Converting Number to String:");
    puts(str);
    printf("\nEnter another number to convert to String:");
    scanf("%ld",&num);
    numtostr(num, str);
    printf("\nAfter Converting 2nd Number to String:");
    puts(str);
}
void numtostr(long int n, char s[])
{
    static int i=0;
    if(n==0)
    {
        i=0;
    return;
    }
    numtostr(n/10, s);
    s[i++] = n%10  + '0';
    s[i]='\0';
}
```

---

11). Write a C program to convert string of numbers to integer using recursion.

```
#include <stdio.h>
#include <string.h>
int num = 0;
int strtonum(char *string1) {
int i = 0;
if (string1[i] != '\0') {
        if (string1[i]<'0' || string1[i]>'9') {
           printf("Not Possible\n");


        }
        else {
            num *= 10;
            num += string1[i] - '0';
            strtonum(&string1[i+1]);
           }
    }
    return num;
}
int main(int argc, const char *argv[]) {
    printf("After string to integer : %i\n",(strtonum("99256")));
    return 0;
}
```

----

12). Write a C program to find all permutations of string by Recursion and Iteration.

```
#include<stdio.h>
#include<string.h>
void strpermute1(char str[], char* currentptr);
void strpermute2(char str[], int startIndex, int lastIndex);
void Swap(char *a, char *b);
int main()
{
        char str[20];
        printf("Enter any String:");
        scanf("%s",str);
        printf("Using Iteration:\n");
        strpermute1(str,str);
        printf("\n\n");
        printf("Using Recursion:\n");
        strpermute2(str,0,strlen(str)-1);
        printf("\n");
        return 0;
}
void strpermute1(char str[], char* currentptr)
{
        char *ptr;
        if(*(currentptr + 1) == '\0')
          printf("%s\t", str);
        else
         for(ptr=currentptr; *ptr!='\0'; ptr++)
         {
                Swap(ptr,currentptr);
                strpermute1(str, currentptr+1);
                Swap(ptr,currentptr);
         }
}
void strpermute2(char str[], int startIndex, int lastIndex)
{
        int i;
        if(startIndex==lastIndex)
        {
                for(i=0;i<=lastIndex;i++)
```

```
            printf("%c",str[i]);
        printf("\t");
    }
    else
    for(i=startIndex;i<=lastIndex;i++)
    {
        Swap(&str[startIndex], &str[i]);
        strpermute2(str,startIndex+1,lastIndex);
        Swap(&str[startIndex], &str[i]);
    }
}
void Swap(char *a, char *b)
{
    char temp = *a;  *a=*b; *b=temp;
}
```
