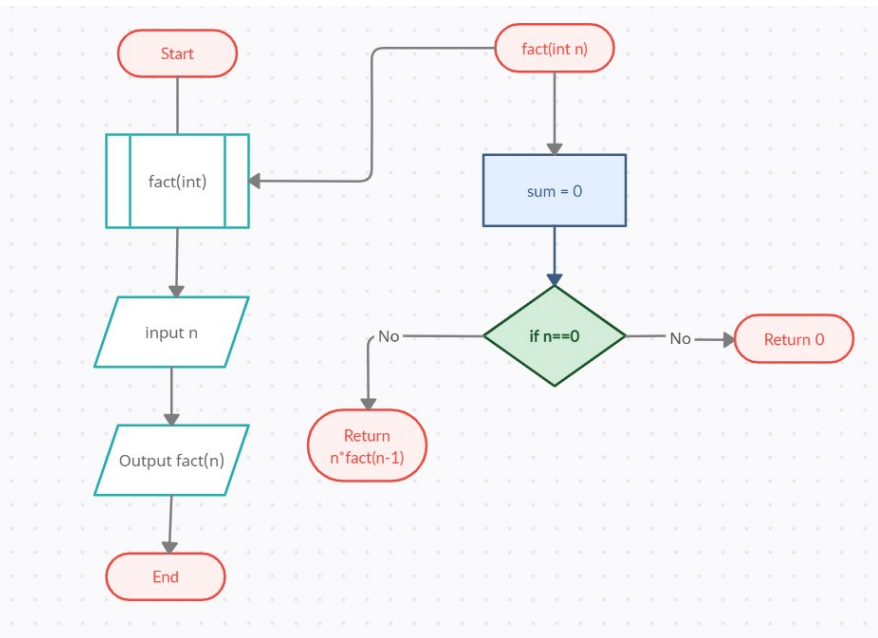
Name	**Hatim Yusuf Sawai**
UID no.	**2021300108**
Experiment No.	4

AIM:	Apply the concept of recursion to solve a given problem.
Program 1	
PROBLEM STATEMENT:	Write a recursive function to find the factorial of a number and test it.
ALGORITHM:	1. START (fact) 2. If n==0 Return 1 3. Return n*fact(n-1) 1. START (main) 2. Input n 3. Output fact(n) 4. STOP
FLOWCHART:	

PROGRAM:

```
#include<stdio.h>
int fact(int);
int main()
{
   int n;
   printf("Enter a number:\n");
   scanf("%d",&n);
   printf("%d! = %d",n,fact(n));
   return 0;
}
int fact(int n)
{
   if(n==0)
      return 1;
   return n*fact(n-1);
}
```

RESULT:

```
PS D:\C Programming\C Practicals-SPIT\Experiment-4> cd
rog1 } ; if ($?) { .\prog1 }
Enter a number:
5
5! = 120
PS D:\C Programming\C Practicals-SPIT\Experiment-4>
```

Program 2

PROBLEM STATEMENT:

Write a recursive function which returns the nth term of the fibonacci series. Call it from main() to find the 1st n numbers of the fibonacci series.

ALGORITHM:

1. START
2. Input n
3. For i=0
4. Output fib(i)
5. Repeat step 4 till i<n
6. STOP

	1. START (fib) 2. If n==0 3. return 0 4. else if n==1 5. return 1 6. else return fib(n-1)+fib(n-2);
FLOWCHART:	

PROGRAM:	#include<stdio.h>
	int fib(int n);
	int main()
	{
	int n,i;
	printf("Enter the range of the series:\n");
	scanf("%d",&n);
	for(i=0;i<n;i++)
	{
	printf("%d ",fib(i));
	}
	return 0;
	}
	int fib(int n)
	{
	if(n==0)
	return 0;
	else if(n==1)
	return 1;
	else return fib(n-1)+fib(n-2);
	}

RESULT:

```
PS D:\C Programming\C Practicals-SPIT\Experiment-4> cd
rog2 } ; if ($?) { .\prog2 }
Enter the range of the series:
10
0 1 1 2 3 5 8 13 21 34
PS D:\C Programming\C Practicals-SPIT\Experiment-4>
```

Program 3	
PROBLEM STATEMENT:	Given a number n, print following a pattern without using any loop. Example: Input: n = 16 Output: 16, 11, 6, 1, -4, 1, 6, 11, 16 Input: n = 10 Output: 10, 5, 0, 5, 10
ALGORITHM:	1. START 2. Intput n 3. Series(n) 4. STOP 1. START (series) 2. If n<=0 3. Output n 4. return 0; 5. Output n 6. series(n-5) 7. Output n
FLOWCHART:	

PROGRAM:	#include<stdio.h>
	int series(int);
	int main()
	{
	int n;
	printf("Enter a number:\n");
	scanf("%d",&n);
	series(n);
	return 0;
	}
	int series(int n)
	{
	if(n<=0)
	{
	printf("%d ",n);
	return 0;
	}
	printf("%d ",n);
	series(n-5);
	printf("%d ",n);
	}

RESULT:

```
PS D:\C Programming\C Practicals-SPIT\Experiment-4> cd "
rog3 } ; if ($?) { .\prog3 }
Enter a number:
16
16 11 6 1 -4 1 6 11 16
PS D:\C Programming\C Practicals-SPIT\Experiment-4>
```

Program 4	
PROBLEM STATEMENT:	Ackerman's function is defined by: A(m,n) = n+1 if m=0 = A(m-1,1) if m≠0 and n=0 =A(m-1, A(m,n-1)) if m≠0 and n≠0 Write a function which given m and n returns A(m,n). Tabulate the values of A(m,n) for all m in the range 1 to 3 and all n in the range 1 to 6.
ALGORITHM:	1. START 2. Input m, n 3. For i=1 4. For j=1 5. Output ack(j, i) 6. Repeat step 5 till j<=m 7. Repeat Steps 4-6 till i<=n 8. STOP 1. START (ack) 2. If m==0 3. Return n+1 4. Else if n==0 && m!=0 5. return ack(m-1,1) 6. Else if m!=0 && n!=0 7. return ack(m-1,ack(m,n-1))
FLOWCHART:	

PROGRAM:

```
#include<stdio.h>
int ack(int,int);
int main()
{
   int n, m, i, j;
   printf("Enter range(m,n)\n");
   scanf("%d %d", &m, &n);
   printf("Ackermann series:\n");
   printf("m,n     m=1        m=2        m=3\n");
   for (i = 1; i <= n; i++)
   {
      if(i>=10)
         printf("n=%d   ", i);
      else
         printf("n=%d    ", i);
      for (j = 1; j <= m; j++)
      {
         if(ack(j,i)>=10 && i<10)
            printf("A(%d,%d)=%d    ", j, i, ack(j, i));
         else if(ack(j, i)>=10 && i>=10)
            printf("A(%d,%d)=%d   ", j, i, ack(j, i));
         else
            printf("A(%d,%d)=%d     ", j, i, ack(j, i));
      }
      printf("\n");
   }
   return 0;
}
int ack(int m,int n)
{
   if(m==0)
      return n+1;
   else if(n==0 && m!=0)
      return ack(m-1,1);
   else if(m!=0 && n!=0)
      return ack(m-1,ack(m,n-1));
}
```

RESULT:

```
PS D:\C Programming\C Practicals-SPIT\Experiment-4> cd
rog4 } ; if ($?) { .\prog4 }
Enter range(m,n)
3 10
Ackermann series:
m,n     m=1          m=2          m=3
n=1    A(1,1)=3     A(2,1)=5     A(3,1)=13
n=2    A(1,2)=4     A(2,2)=7     A(3,2)=29
n=3    A(1,3)=5     A(2,3)=9     A(3,3)=61
n=4    A(1,4)=6     A(2,4)=11    A(3,4)=125
n=5    A(1,5)=7     A(2,5)=13    A(3,5)=253
n=6    A(1,6)=8     A(2,6)=15    A(3,6)=509
n=7    A(1,7)=9     A(2,7)=17    A(3,7)=1021
n=8    A(1,8)=10    A(2,8)=19    A(3,8)=2045
n=9    A(1,9)=11    A(2,9)=21    A(3,9)=4093
n=10   A(1,10)=12   A(2,10)=23   A(3,10)=8189
PS D:\C Programming\C Practicals-SPIT\Experiment-4>
```

Program 5	
PROBLEM STATEMENT:	Write a recursive function to return the minimum number of coins of given set of coin values that are required to produce a given amount. For example if you are given set of values {1,4,5}(indicating you had a supply of 1-cent,4-cent and 5-cent coins), and the amount 8, you should return 2, since the value 8 cents can be made with two 4-cent coins.
ALGORITHM:	1. START 2. Input n 3. For i=0 4. Input c[i] 5. Repeat step 4 till i<n 6. Input amt 7. Output mincoins(c, n, amt) 8. STOP

	1. START (mincoins) 2. If amt==0 3. Return 0 4. int res = __INT_MAX__ 5. for i=0 6. sub=0 7. if amt>=c[i] 8. sub = 1 + mincoins(c, n, amt - c[i]) 9. res = min(res, sub); 10. Repeat steps 6-9 till i<n 11. If res == __INT_MAX__ 12. Return -1 13. Return res
FLOWCHART:	

PROGRAM:	#include<stdio.h>
	#include<math.h>
	#define min(X, Y) (((X) < (Y)) ? (X) : (Y));
	int mincoins(int *,int,int);
	int main()
	{
	int c[10],amt,n,min;
	printf("Enter number of set values:\n");
	scanf("%d",&n);
	printf("Enter %d set values:\n",n);
	for(int i=0;i<n;i++)
	scanf("%d",&c[i]);
	printf("Enter the amount:\n");
	scanf("%d",&amt);
	printf("Minimum number of coins required: %d",mincoins(c,n,amt));
	return 0;
	}

```
int mincoins(int c[], int n,int amt)
{
   if (amt == 0)
      return 0;
   int res = __INT_MAX__;
   for (int i = 0; i < n; i++)
   {
      int sub=0;
      if (amt >= c[i])
      {
         sub = 1 + mincoins(c,n, amt - c[i]);
         res = min(res, sub);
      }
   }
   if (res == __INT_MAX__)
      return -1;
   return res;
}
```

RESULT:

```
PS D:\C Programming\C Practicals-SPIT\Experiment-4> cd "
prog5a } ; if ($?) { .\prog5a }
Enter number of set values:
3
Enter 3 set values:
1 4 5
Enter the amount:
8
Minimum number of coins required: 2
PS D:\C Programming\C Practicals-SPIT\Experiment-4>
```

CONCLUSION: In this experiment, we learnt the concept of recursion and how to apply recursion and make a recursive function to solve a given C program.