07-06-2024

FACTORIAL USING RECURSSION =>

```
-> int fact (int n);
   main ()
   {
     int n, f;
     printf ("enter n: ");
     scanf ("%d", &n);
     f = fact (n);
     printf ("%d", f);
   }
   int fact (int n)
   {
     if (n == 0 || n == 1)
     return 1;
     else
     return n * fact (n-1);
   }
```

UP -> 1
2
3

Down -> 3
2
1

O/P :-

Enter n : 5

120

POWERS USING RECCURSION => x^y

```
-> Void pow(int, int);
   main()
   {
      int x, y, P;
      printf("enter x, y");
      scanf("%d %d", &x, &y);
      P = pow(x, y);
      printf("%d ^ %d = %d", x, y, P);
   }
int pow(int x, int y)
   {
     if (y == 0)
     return 1;
     else
     return x* pow(x, y-1);
   }
```

return :

x * Pow(1)

x * 1

1

x * Pow(2)

x * x

↓

x * Pow(3)

x * x^2

---

x

```
PAT -> int pow(int, int);
       main()
       {
          int x, y, P;
          printf("enter x, y:");
          scanf("%d %d", &x, &y);
          P = pow(x, y);
          printf("%d ^ %d = %d", x, y, P);
       }
       int pow(int x, int y)
       {
          if (y == 0)
          return 1;
          else
          return x * pow(x, y-1);
       }
```

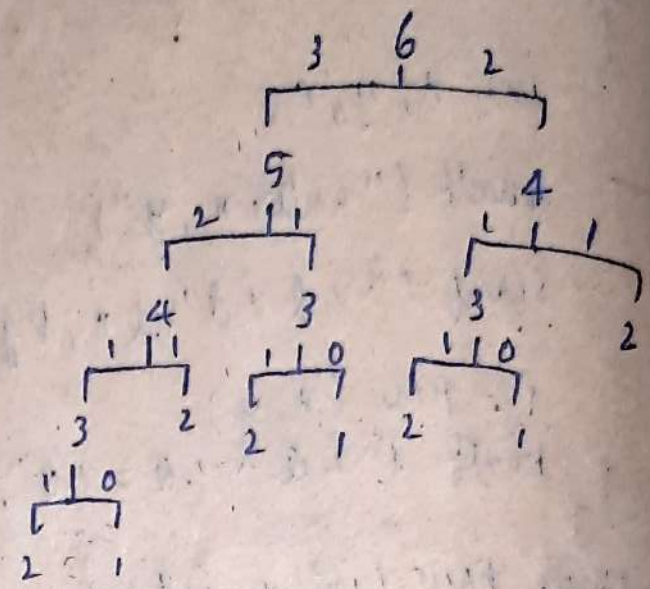
$2^3 = 8$

$3^3 = 27$

$4^5 = 1024$

FIBONACCI SERIES USING RECURSSION!:-

→
```
int fab(int);
main()
{
  int i, n, f;
  printf("enter n");
  scanf("%d", &n);
  for(i=1; i<=n; i++)
  {
    f = fab(i);
    printf("%d", f);
  }
}
int fab(int n)
{
  if (n==1)
  return 0;
  else if (n==2)
  return 1;
  else
  return fab(n-1) + fab(n-2);
}
```



FIBONACCI SERIES USING RECURSION :-

```
-> int fib (int n);
int main()
{
   int i, n, f;
   printf ("enter n: ");
   scanf ("%d", &n);
   for (i = 1; i <= n; i++)
   {
      f = fib (i);
      printf ("%d", f);
   }
}
int fib (int n)
{
   if (n == 1)
   return 0;
   else if (n == 2)
   return 1;
   else
   return fib (n-1) + fib (n-2);
}
```

O/P :-

```
Enter n: 10
0 1 1 2 3 5 8 13 21 34
```