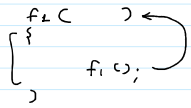
Day 12

29 March 2025 08:27 AM

Recursive function :-



5! = 5 * 4*3*2*1

5! = 5 * 4! 24 → 120

4! = 4 * 3! 6

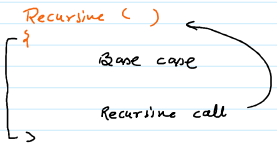
3! = 3 * 2! 2

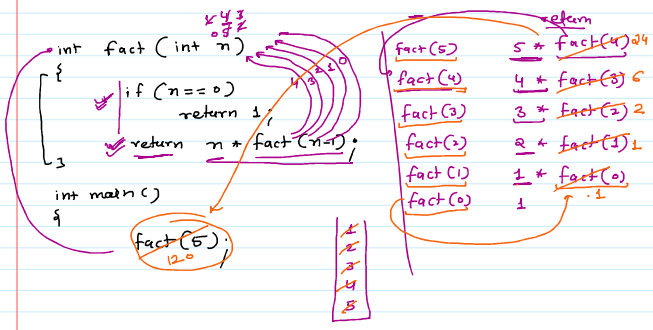
2! = 2 * 1! 1

1! = 1 * 0! 1

0! = 1

n * fact(n-1)





stack → LIFO list

```
#include<iostream>
using namespace std;
int fact(int n)
{
    if(n==0)
        return 1;
    return n*fact(n-1);
}
int main()
{
    int n;
    cout<<"Enter a number:";
    cin>>n;
    cout<<fact(n);
    return 0;
}
```

W.A.R.F to calculate power of a number-

n p n^p

$2^5 \rightarrow 2*2*2*2*2$

p or

$n^p = n * n^{p-1}$

$2^5 = 2 * 2^4$

$2^4 = 2 * 2^3$

$2^3 = 2 * 2^2$

$2^2 = 2 * 2^1$

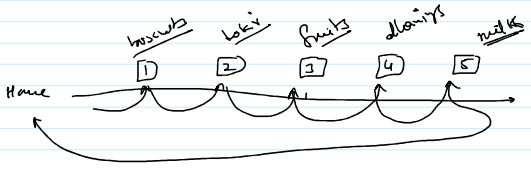
$2^1 = 2 * 2^0$

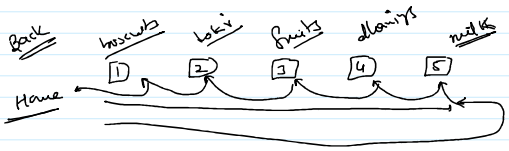
$2^0 = 1$

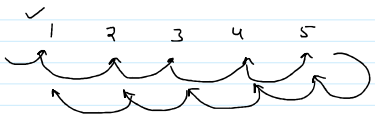
```
#include<iostream>
using namespace std;
int merapower(int n,int p)
{
```

```
    if(p==0)
        return 1;
    return n*merapower(n,p-1);
}
int main()
{
    int n,p;
    cout<<"Enter a number and its power:";
    cin>>n>>p;
    cout<<merapower(n,p);
    return 0;
}
```

1 2 3 4 5 ... n → Recursive func







5 4 3 2 1

```
#include<iostream>
using namespace std;
void series(int st, int n)
{
    if(st>n)
        return;
    cout<<st<<" ";
    series(st+1,n);
}
int main()
{
    int n;
    cout<<"Enter num of terms:";
    cin>>n;
    series(1,n);
    return 0;
}
```

```
#include<iostream>
using namespace std;
void series(int n)
{
    if(n==0)
        return;
    series(n-1);
    cout<<n<<" ";
}
int main()
{
    int n;
    cout<<"Enter num of terms:";
    cin>>n;
    series(n);
    return 0;
}
```

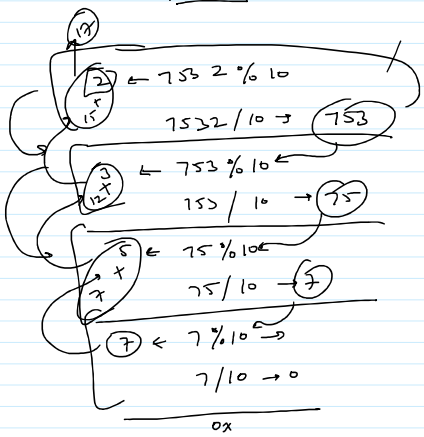
n ... 5 4 3 2 1 2 3 4 5 ... n

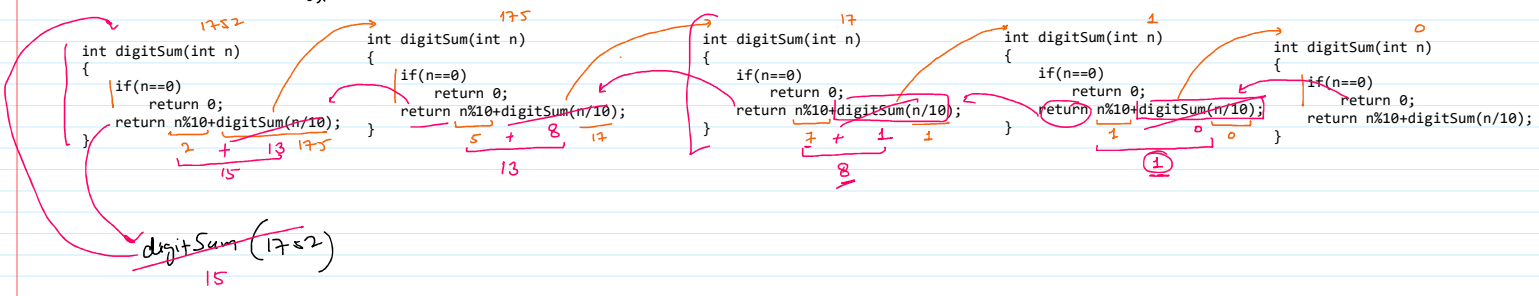
```
#include<iostream>
using namespace std;
void series(int n)
{
    if(n==1){
        cout<<n<<" ";
        return;
    }
    cout<<n<<" ";
```

```
    series(n-1);
    cout<<n<<" ";
}
int main()
{
    int n;
    cout<<"Enter num of terms:";
    cin>>n;
    series(n);
    return 0;
}
```

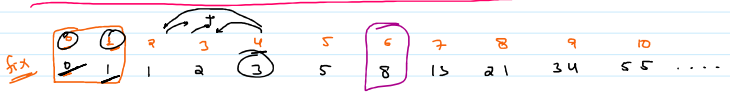
Sum of digits using Recursive func

n | 7532 7+5+3+2 → 17 (sum)



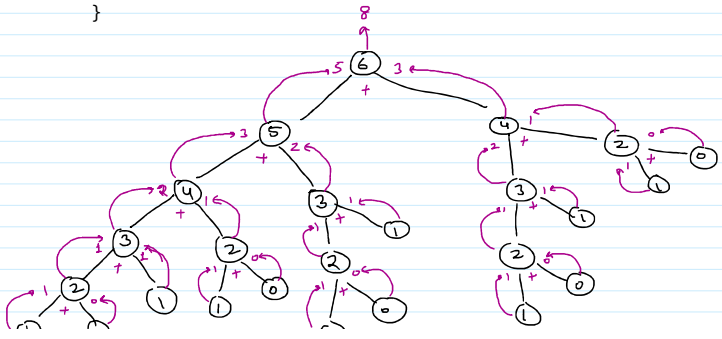


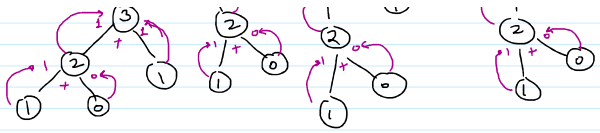
print nth term of fibonacci Series :-



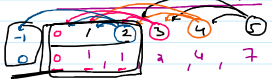

```
int fibo(int n)
{
    if (n==0 || n==1)
        return n;
    return fibo(n-1) + fibo(n-2);
}
```

```
#include<iostream>
using namespace std;
int fibo(int n)
{
    if(n==0||n==1)
        return n;
    return fibo(n-1)+fibo(n-2);
}
int main()
{
    cout<<fibo(8);
    return 0;
}
```





Tribonacci Series :- n^{th} term → Recursion

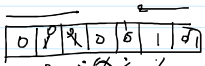


0 1 1 2 4 7 13 24 ...

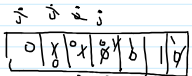
n = -1 → 0
n = 0 → 0
n = 1 → 1

n = 0 → 0
n = 1 → 1
n = 2 → 1

```
int tribo (int n)
{
    if (n == 1 || n == 0)
        return 0;
    if (n == 1)
        return 1;
    return tribo(n-1) + tribo(n-2) + tribo(n-3);
}
```



```
for(i=1, j=n-1; i<j; )
    if(arr[i] == 0 && arr[j] != 0)
        swap(arr[i++], arr[j--])
    if(arr[i] != 0)
        i++
    if(arr[j] == 0)
        j--
```



```
for(i=0; i<n; i++)
{
    if(arr[i] != 0)
    {
        swap(arr[i], arr[j]);
        j++;
    }
}
```

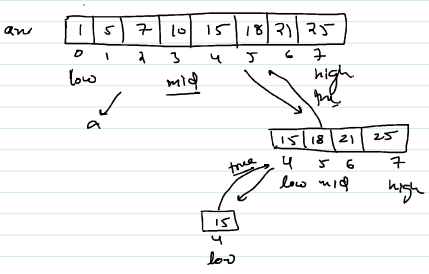
Binary Search :- Recursive

search

```
bool Binary_search (int arr[], int low, int high, int num)
{
    if(low > high)
        return false;
    int mid = (low + high)/2
    if(arr[mid] == num)
        return true;
    if(num < arr[mid])
    return Binary_search(arr, low, mid-1, num);
    else
    return Binary_search(arr, mid+1, high, num);
}
```

O(log₂n)

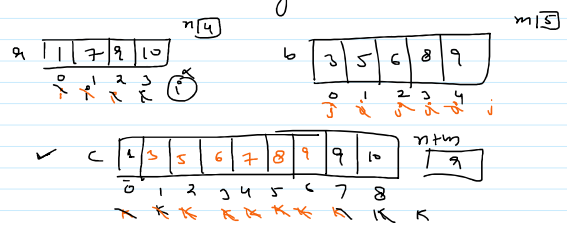
num 15



high
mid

merge Sorting :→

* merge two sorted Array




```
for(i=0,j=0,k=0; i<n&&j<m; k++)
{
    if(a[i]<b[j]
    {   c[k]=a[i];
        i++;
    }
    else
    {   c[k]=b[j];
        j++;
    }
}
```

O(n+m)

```
while(i<n)
{
    c[k]=a[i];
    i++;
    k++;
}
```

```
while(j<m)
{
    c[k]=b[j];
    j++;
    k++;
}
```

Hw

find pair with K sum →

arr | 5 | 7 | 2 | 9 | 2 | 6 | 1 | K | 16 |