Data Structures

Prerequisites

- very often Data structures deals with recursion and pointers, so it is important to revise them.

What is recursion??????

- Recursion is a process in which a function calls itself as subroutine.This process continues when it meets with a specific condition(if ,else ,else if,etc) and if the base condition is satisfied the function loops back to the beginning of itself.

->Now we will revise recursion with an example.

Let us see the factorial problem.

➔ **Factorial Problem**

```
int factorial (int n)
{
    if (n == 1)   // Base Case
    {
        return 1;
    }
    else   // Recursive Case
    {
        return n * factorial(n-1);
    }
}
```

```
int factorial (int n) {
if (n == 1) // Base Case
return 1;
else // Recursive Case
return n * factorial(n-1);
}
```

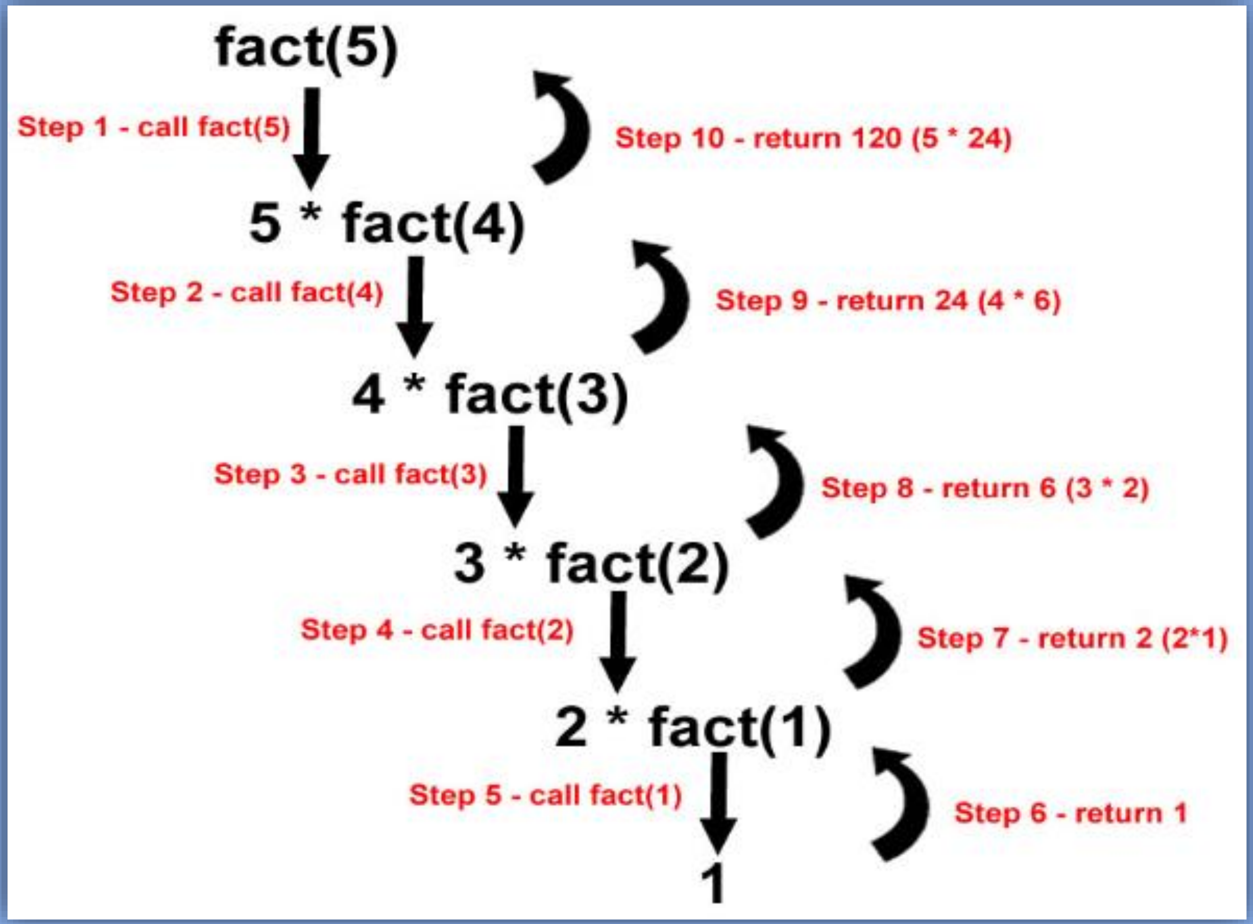
Consider n = 5

Factorial (5) = 5 * Factorial (4)

= 5 * 4 * Factorial (3) … and so on

➔ **Factorial (5) = 5 * 4 * 3 * 2 * 1 = 120**

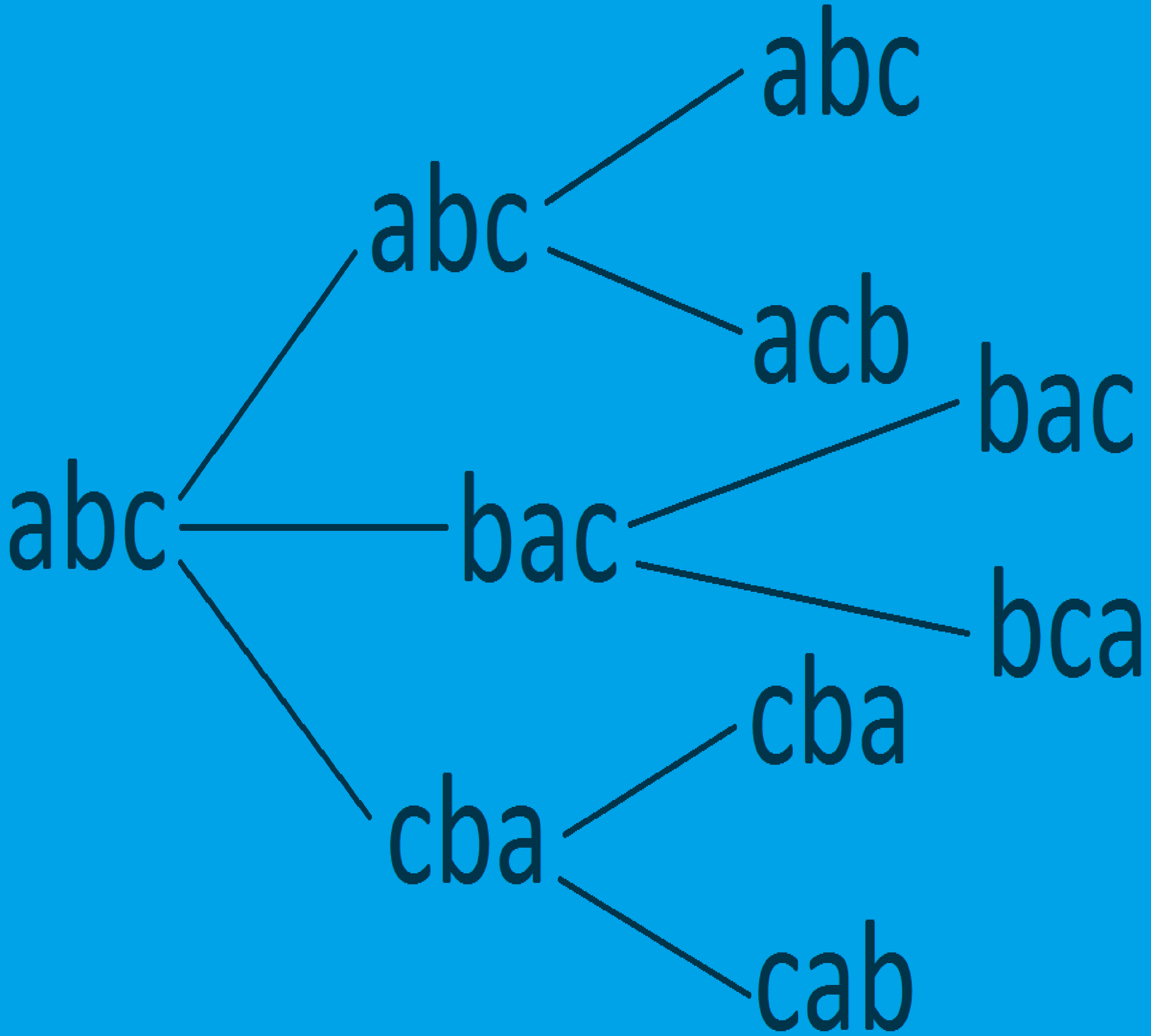
5 * factorial (4) → 120



String Permutations

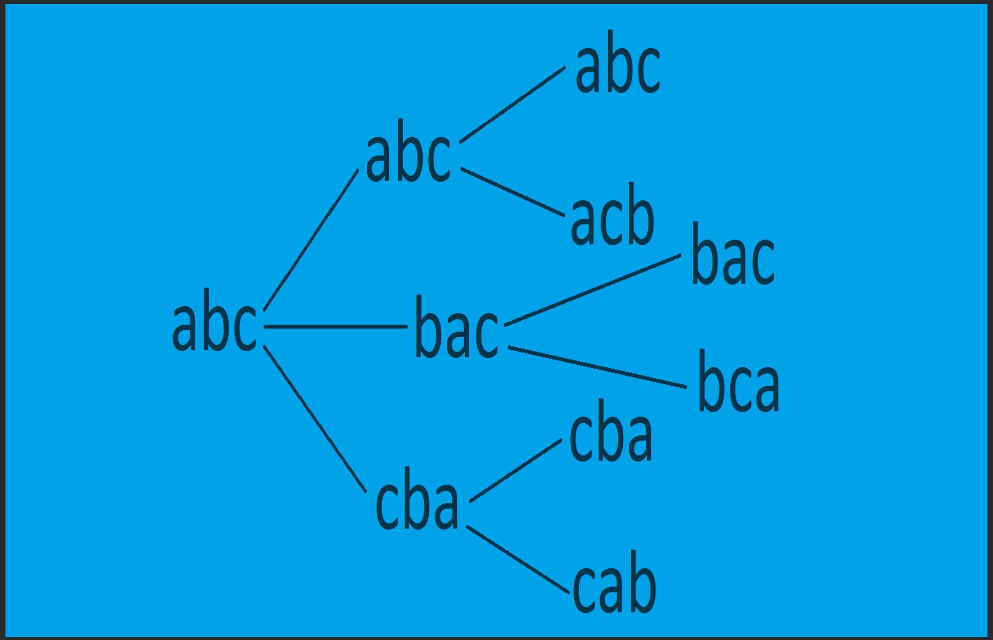
- **Input: “abc“**
- **begin: 0;**
- **end: 3**
- **Number of permutations = 1 x 2 x 3 = 6**
- **Output: abc,acb,bac,bca,cba,cab.**

How this occurs???


abc
abc
bac
cba
abc
acb
bac
bca
cba
cab

```c
void swap(char *a,char *b) {
    char temp;
    temp = *a;
    *a=*b;
    *b=temp;
}

void permute(char *str,int start,int end)
{
    int i,range;
    range=end-start;
    if(range==1){
        printf("%s\n",str);
    }
    else {
        for (i = 0; i < range; i++) {
            swap(&str[start], &str[start + i]);
            permute(str, start + 1, end);
            swap(&str[start], &str[start + i]);
        }
    }
}
```



Pointers

What is a Pointer????

- A **pointer** is a variable whose value is the address of another variable(implies direct address of the memory location).Like any variable or constant, you must declare a pointer before using it to store any variable address.

Declaration of a pointer:

- Syntax -> datatype *variable name;
- It is good habit to point to NULL;

- Integer pointer array -> int *variablename[size of array];
- Character pointer array -> char *variablename={“string1”,”string”,”string3”}

Example:

```c
int main() {
    char string[]={'a','b','c','d','\0'};
    char *str;
        str=&string[1];
        *str='z';
        printf("%x ",str);
        printf("%s ",str);
        str++;
        printf("%x ",str);
        printf("%s ",str);
        printf("%s ",string);
    return 0;
}
```

Output:

```
60ff28   zcd   60ff29   cd   azcd
```