1(n is a positive integral number)

(a) Recursion

```
  int factorial(int n) {
   if (n == 0)
          return 1;
 else
         return n * factorial(n-1);
}
```

$\xrightarrow{\text{For}}$

Iteration

```
int factorial(int n)
{
   int res = 1, i;
   for (i = 2; i <= n; i++)
      res *= i;
   return res;
}
```

(b) Recursion

```
void countDown(int n) {
            if(n == 0)
                   return;
            cout << n << endl;
            countDown(n-1);
            }
```

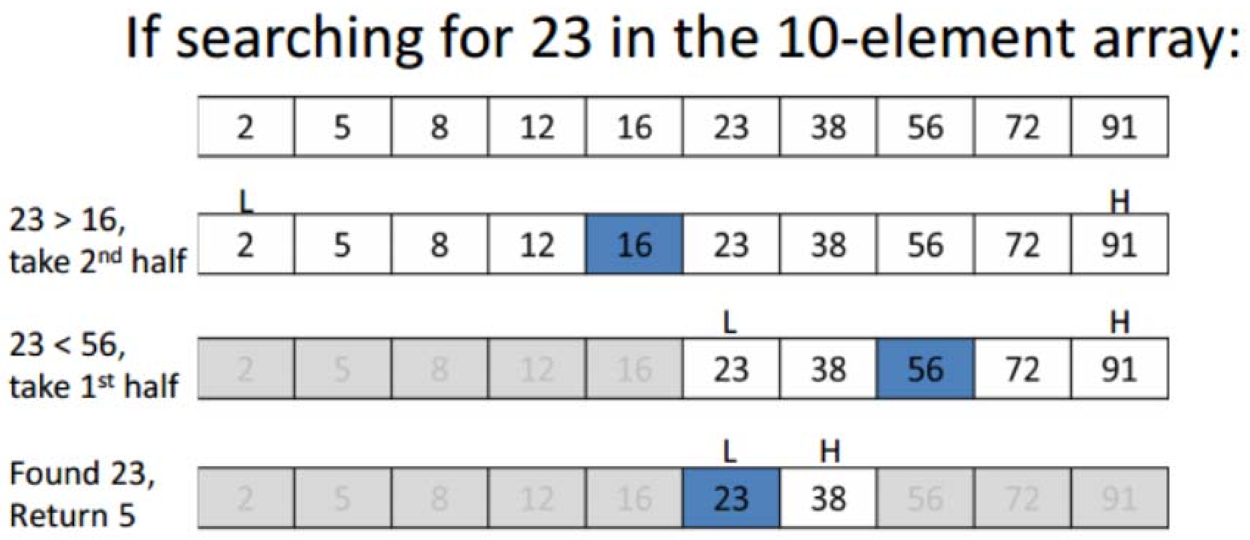
$\xrightarrow{\text{While}}$

Iteration

```
void countDown(int n) {
            while(n > 0) {
            cout << n << endl;
             n --;}
            }
```

(c) Binary Search

1. Begins by comparing the middle element of the array with the target value;
2. If the target value matches the middle element, its position in the array is returned;
3. If the target value is less than the middle element, the search continues in the lower half of the array.
4. If the target value is greater than the middle element, the search continues in the upper half of the array.



Recursion

```
 int binarySearch(int arr[], int l, int r, int x)
{
   if (r >= l)
   {
        int mid = l + (r - l)/2;
        if (arr[mid] == x)   //target in the middle position
                return mid;
         if (arr[mid] > x)
               return binarySearch(arr, l, mid-1, x); // target in a lower section
        return binarySearch(arr, mid+1, r, x); //otherwise, target has to be in a upper section
   }
   return -1; //not found
}
```