Bubble sort.

```
for(int i=0 ; i< n-1 ; i++)        // n-1 → size of the array
{
 int flag=0;
for(int j=0; j < n-i-1;j++)
{

if ( arr[j] > arr [j+1])
{

int temp= arr[j];
arr[j] = arr[j+1];
arr[j+1] = temp;        flag=1;
}
if(flag==0) break;      // → means that the list is sorted and no need to traverse it again and again.
}
```

no of rounds = total no of elements − 1

with every round we need not compare the last element because with every round largest no gets placed at the end

36	19	29	12	5

⟹ Round 1 → i=0

j=0	36	19	29	12	5
j=1	19	36	29	12	5
j=2	19	29	36	12	5
j=3	19	29	12	36	5
	19	29	12	5	36

⟹ Round 2 ⇒ i= 1

j=0	19	29	12	5	36
j=1	19	29	12	5	36
j=2	19	12	29	5	36
	19	12	5	29	36

⟹ Round 3 ⇒ i= 2

j=0	19	12	5	29	36
j=1	12	19	5	29	36
	12	5	19	29	36

⟹ Round 4 ⇒ i= 3

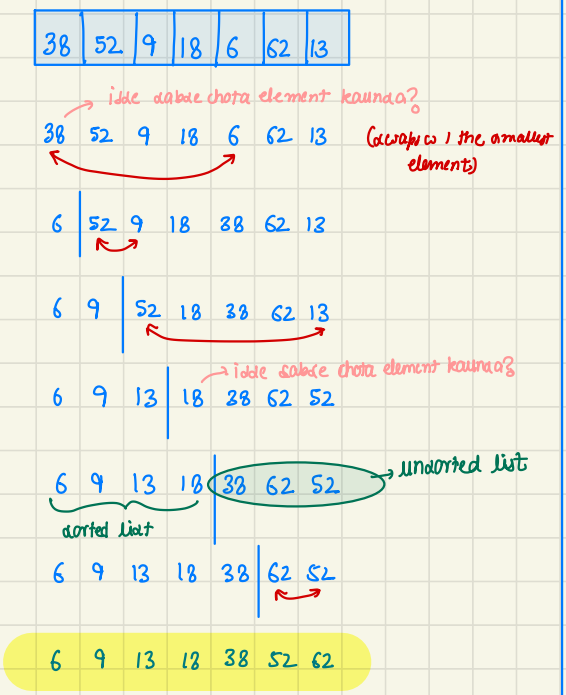
j=0	12	5	19	29	36
	5	12	19	29	36

Selection Sort

Combination of searching and sorting.

finds the smallest element in the array and puts it in the beginning

On every iteration of selection sort, the minimum element is picked from the unsorted array and moved to the sorted subarray.

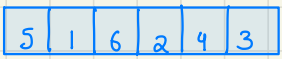


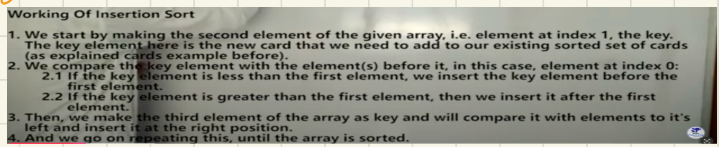

```
int min;
for(int i=0; i< n; i++)
{
 min = i;
 for(int j=i+1; j< n-1; j++)
 {
  if( a[j] < a[min])
  {
   min = j;
  }
 }
 int temp = arr[i];
 arr[i] = arr[min];
 arr[min] = temp;
}
```

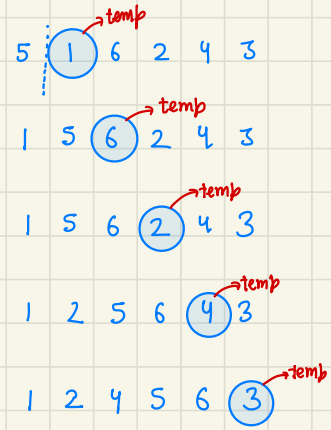
searches for smaller element than at ith index.

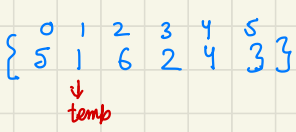
Insertion Sort

It works the way we sort playing cards in our hand. We choose one card and insert it in its position (ascending or descending)










```
int temp,j;
for(int i=1; i< n ; i++)
{
temp= arr[i];        → defects the temp variable
j=i;
while(j>0 && arr[j-1] >temp)
{                    → performs swapping till temp ae chota element nahi jaye.
arr[j]= arr[j-1];
 j=j-1;  // j--;
}
arr[j] = temp;
}
```