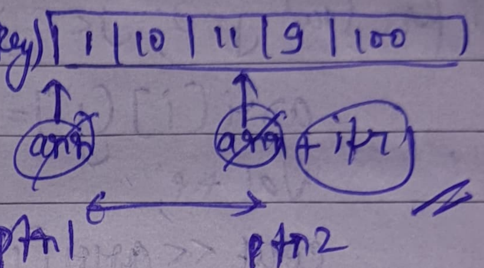
Page No.
Date :

Algorithm STL

① find function:-

find(arr, arr+n , key)
→ give iterators of the required key.

ie auto it = find(arr, arr+n, key)
int index = it - arr ;



② Binary search STL
→ when you have sorted array (log n) → time complexity [O]

bool present = binary_search (arr, arr+n, key);
↳ return true / false

→ in order to get the index of the required key, we can use
i) lower_bound(start, end, key)
→ will return the address of the first element ≥ key

ii) upper_bound(s, e, key)
→ > key

Hence,
→ upper bound - lower bound } → give frequency of the ~~el~~ key

Page No.
Date :

③ Rotate

```
rotate(arr, arr+2, arr+n) ;
```
→ do baar rotate karega.

④ Next permutation

```
next_permutation(v.begin(), v.end());
```
will get
→ lexographical next permutation.

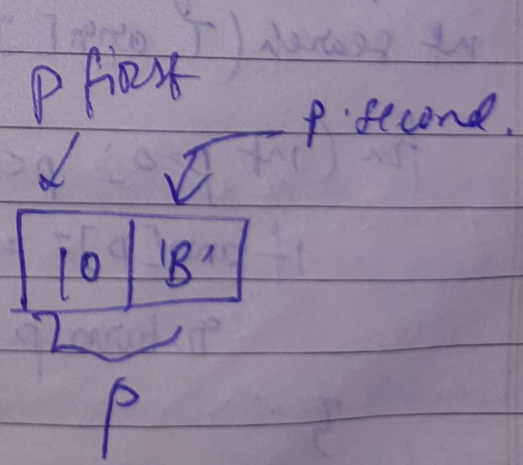
⑤
```
swap(a,b);
```
Similarly,
```
max(a,b);
min(a,b);
reverse(a, a+4);
```
→ reverse 1st 4 element of array.

⑥ Pair → bind two entities

```
pair<int, char> p;
 p.first = 10;
 p.~~char~~ second = 'B';
```
or

```
pair<int, char> p2(p);
```

→ we can use
```
pair<int, int> p4 = make_pair(100, "Audi");
```



Page No.
Date :

// pairs of pairs

```
pair <pair <int, int>, string> car;

car.second = "Audi";
car.first.first = 10;
car.first.second = 20;
```

generic programming

→ it means all the containers and all the algorithm provided by STL are generics. Generic means they can work with different datatypes

eg
```
template <typename T>                    } → bas itna change
int search (T arr[ ], int n, T key) { } → aage baaki sab same

  for (int p=0; p<n ;p++) {
      if (arr[p] = = key) {
        return p;

  }
```

// function call
```
    search( b, 3, K) <<endl;
```
← we can call above function for any data type