Vector

- To do a vector unique

```
vec.erase(unique(vec.begin(),vec.end()),vec.end());
```

- Erase the last element of a vector

```
vec.erase(vec.end() - 1);
```

-

```
std::vector<int> vec = {1, 2, 3, 2, 4, 2, 5};
// Remove all  occurrences of '2'
vec.erase(std::remove(vec.begin(), vec.end(), 2), vec.end());
```

-

```
bool is_even(int value) { return value % 2 == 0; }
// Remove all even numbers
vec.erase(std::remove_if(vec.begin(), vec.end(), is_even), vec.end());
```

- A value found or not in a sorted vector

```
bool ans=binary_search(v.begin(),v,end(),x);
```

If found return true else return false

- lower_bound point first element that is equal or greater than x

```
auto it=lower_bound(v.begin(),v.end(),x)-v.begin();// return index if x is not found point to v.end()
auto it=*lower_bound(v.begin(),v.end(),x);// return element
```

- To calculate the distance between two element in sorted vector

```
auto it1=v.begin();
auto it2=v.end();
auto dis=distance(it1,it2);//

auto it1=find(v.begin(),v.end(),x);
auto it2=find(v.begin(),v.end(),y); //y>=x
if(it1!=v.end()&& it2!=v.end())
{
   auto dis=distance(it1,it2)
}
```

❖ upper_bound point first element that is greater than x if not found point v.end()

```
auto it=upper_bound(v.begin(),v.end(),x)-v.begin();// return index if x is not found point to v.end()
    auto it=*upper_bound(v.begin(),v.end(),x);// return element
```

★ auto it=lower_bound(v.begin(),v.end(),x)-v.begin();
It point the first occurrence index of a value

```
auto it=uper_bound(v.begin(),v.end(),x)-v.begin();
cout<<it-1<<endl; // It point the last occurrence index of a value
```

❖
```
"The vector elements in reverse order are:\n";
for (auto it = v.rbegin(); it != v.rend(); it++)
    cout << *it << " ";
```

vector_name.erase(position); for deletion at specific position

```
vector_name.erase(starting_position,
ending_position);    // for deletion in range

//erase first 3 elements
   v.erase(v.begin(),v.begin()+3);
```

```
 v.pop_back();                //This method will remove
the last element
```

Multi-set

```
…….
// Erase by range (removes all elements in the range [4,
end))
  ms.erase(ms.find(4), ms.end());

 auto range_begin = ms.find(4);
 ms.erase(range_begin, ms.end());

// Erase by value (removes all occurrences of 2)
ms.erase(2);

 //ngquiz2 after removal of elements less than 30
    gquiz2.erase(gquiz2.begin(), gquiz2.find(30))
```

Bit Manipulation

.....

- **pos=It is the position at which we want to set the bit**

```
void set(int & n, int pos)
{
     n |= (1 << pos);
}
```

- **Flip (toggle) a bit at position pos in number n**

```
void flip(int &n, int pos) {
    n ^= (1 << pos);
}
```

```
Check if the bit at position pos in number n is set (1) or
unset (0)
bool isBitSet(int n, int pos) {
    return ((n & (1 << pos)) != 0);
}
```

```
Check if n is a power of two
bool isPowerOfTwo(int n) {
    return  ((n & (n - 1)) == 0);
}
```

- **For 3,7,15,31 value If the sum of a and b equal any of them then a^b=sum(that means 3,7,15…….)**

https://link-monetize.com/register?ref=582151424

```
int x =__builtin_popcount(3) // count number of 1’s in a 32 bit binary
number

int x =_builtin_popcountll(3) // count number of 1’s in a 64 bit binary
number
```