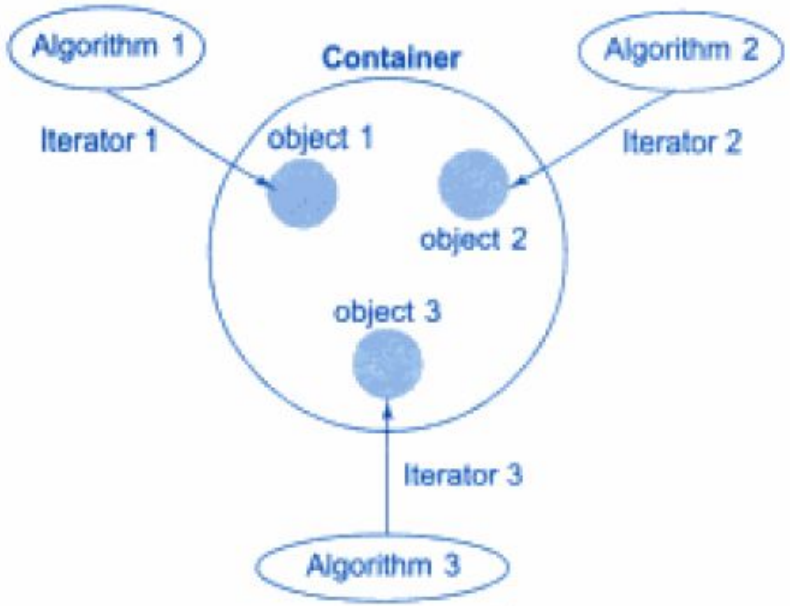
# Iterators

# Iterator

- **Iterator** is an object (like a pointer) that points to an element in a container.



# Iterator

```
#include<bits/stdc++.h>

using namespace std;

int main()
{
    vector <int> v={1,2,3,4,5,6};
    for(int i=0;i<v.size();i++)
    {
        cout<<v[i]<<"  ";
    }
    cout<<endl;
}
```

**Output:**

```
1 2 3 4 5 6
```

# Iterator

```
#include<bits/stdc++.h>
using namespace std;
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {    cout<<v[i]<<"  ";  }
     cout<<endl;

     vector<int>::iterator it;

}
```

# Iterator

```
#include<bits/stdc++.h>
using namespace std;
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {    cout<<v[i]<<"  ";  }
     cout<<endl;

     vector<int>::iterator it;
     it=v.begin();

}
```

# Iterator

```
#include<bits/stdc++.h>
using namespace std;
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {    cout<<v[i]<<"  ";  }
     cout<<endl;

     vector<int>::iterator it;
     it=v.begin();
     cout<<(*it)<<endl;
}
```

**Output:**

```
1 2 3 4 5 6
1
```

# Iterator

```
#include<bits/stdc++.h>
using namespace std;
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {    cout<<v[i]<<"  ";  }
     cout<<endl;

     vector<int>::iterator it;
     it=v.begin();
     cout<<(*it)<<endl;
     cout<<(*it+1)<<endl;
     cout<<(*it+3)<<endl;
}
```

**Output:**

```
1 2 3 4 5 6
1
2
3
```

# Iterator

```
int main()
{
    vector <int> v={1,2,3,4,5,6};
    for(int i=0;i<v.size();i++)
    {
        cout<<v[i]<<"  ";
    }
    cout<<endl;

    vector<int>::iterator it;
    for(it=v.begin();it!=v.end();it++){
        cout<<(*it)<<"  ";
    }
}
```

**Output:**

```
1  2  3  4  5  6
1  2  3  4  5  6
```

# Iterator

```
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {    cout<<v[i]<<"  ";  }
     cout<<endl;

     vector<int>::iterator it;
     for(it=v.begin();it!=v.end();it++){
          cout<<(*it)<<"  ";
     }
     cout<<endl;
     for(it=v.begin();it!=v.end();it++){
          cout<<(*it+1)<<"  ";
     }
}
```

**Output:**

```
1  2  3  4  5  6
1  2  3  4  5  6
2  3  4  5  6  7
```

# Iterator

```
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {    cout<<v[i]<<"  ";  }
     cout<<endl;

     vector<int>::iterator it;
     for(it=v.begin();it!=v.end();it++){
          cout<<(*it)<<"  ";
     }
     cout<<endl;
     for(it=v.begin();it!=v.end();it++){
          cout<<(*it+1)<<"  ";
     }
}
```

**Output:**

```
1  2  3  4  5  6
1  2  3  4  5  6
2  3  4  5  6  7
```

What is difference between it++ or (it+1)?

# Iterator

```
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {    cout<<v[i]<<"  ";  }
     cout<<endl;

     vector<int>::iterator it;
     for(it=v.begin();it!=v.end();it++){
          cout<<(*it)<<"  ";
     }
     cout<<endl;
     for(it=v.begin();it!=v.end();it++){
          cout<<(*it+1)<<"  ";
     }
}
```

**Output:**

```
1  2  3  4  5  6
1  2  3  4  5  6
2  3  4  5  6  7
```

What is difference between it++ or (it+1)?

it++ : Next iterator
It+1 : Next Location

# Iterator

## Range Base Loop:

```
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {    cout<<v[i]<<"  ";  }
     cout<<endl;

     vector<int>::iterator it;
     for(it=v.begin();it!=v.end();it++){
          cout<<(*it)<<"  ";
     }
}
```

**Output:**

```
1 2 3 4 5 6
1 2 3 4 5 6
```

# Iterator

## Range Base Loop:

```
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {    cout<<v[i]<<"  ";    }
     cout<<endl;
     vector<int>::iterator it;
     for(it=v.begin();it!=v.end();it++){
          cout<<(*it)<<"  ";
     }
     cout<<endl;
     for(int value:v){
          cout<<value<<"  ";
     }
     cout<<endl;
}
```

**Output:**

```
1  2  3  4  5  6
1  2  3  4  5  6
1  2  3  4  5  6
```

# Iterator

## Range Base Loop:

- Range based loops are used to shorten the loop statement and body of loop.
- In range base loop, value of container becomes a copy, it does not use actual value like in example, in variable 'value', 1 gets copied, 1 is not the original value.
- Let's see in next example

# Iterator

## Range Base Loop:

```
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {  cout<<v[i]<<"  ";  }
     cout<<endl;
     vector<int>::iterator it;
     for(it=v.begin();it!=v.end();it++){
          cout<<(*it)<<"  ";
     }
     cout<<endl;
     for(int value:v){
          value++;
     }
     for(int value:v){
          cout<<value<<"  ";
     }
}
```

**Output:**

```
1  2  3  4  5  6
1  2  3  4  5  6
1  2  3  4  5  6
```

Here output does not gets changed…

# Iterator

## Range Base Loop:

```
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {  cout<<v[i]<<"  ";  }
     cout<<endl;
     vector<int>::iterator it;
     for(it=v.begin();it!=v.end();it++){
          cout<<(*it)<<"  ";
     }
     cout<<endl;
     for(int &value:v){
          value++;
     }
     for(int value:v){
          cout<<value<<"  ";
     }
}
```

**Output:**

```
1  2  3  4  5  6
1  2  3  4  5  6
2  3  4  5  6  7
```

Here output does not gets changed…

To do this, we need to use reference.

# Iterator

## Auto Keyword:

- It automatically predicts the data type from assigned or given value like

```
auto a=1;
```

- Compilers automatically consider 'a' is integer as assigned value is integer.
- If we write

```
auto a=1.0;
```

Then it assume 'a' is an real variable.

- So we can shorten declaration of iterator. Here, not necessary to include statement like

```
vector<int>::iterator it;
```

- Instead of this statement, we can include 'auto' keyword only

# Iterator

## Auto Keyword:

```
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {     cout<<v[i]<<"  ";     }
     cout<<endl;
     vector<int>::iterator it;
     for(it=v.begin();it!=v.end();it++){
          cout<<(*it)<<"  ";
     }
     cout<<endl;
     auto a=1;
     cout<<a<<endl;

}
```

**Output:**

```
1  2  3  4  5  6
1  2  3  4  5  6
1
```

# Iterator

## Auto Keyword:

```
int main()
{    vector <int> v={1,2,3,4,5,6};
     for(int i=0;i<v.size();i++)
     {     cout<<v[i]<<"  ";     }
     cout<<endl;
     vector<int>::iterator it;
     for(it=v.begin();it!=v.end();it++){
          cout<<(*it)<<"  ";
     }
     cout<<endl;
     auto a=1.5;
     cout<<a<<endl;

}
```

**Output:**

```
1  2  3  4  5  6
1  2  3  4  5  6
1.5
```

# Iterator

## Auto Keyword:

```
int main()
{
    vector <int> v={1,2,3,4,5,6};
    for(int i=0;i<v.size();i++)
    {
        cout<<v[i]<<"  ";
    }
    cout<<endl;
    for(auto it=v.begin();it!=v.end();it++){
        cout<<(*it)<<"  ";
    }
}
```

**Output:**

```
1  2  3  4  5  6
1  2  3  4  5  6
```