

इतना मत गिरो कि उठने का यकीन टूट जाए,
इतना उठो कि गिरने वाला भी सोच में पड़ जाए।



Standard Template Library

STL is a collection of pre-built classes and functions

Components of STL



Containers

Hold and organize the data.



Algorithms

Perform actions like sorting or searching on the data.



Iterators

Helps go through the data in containers one by one.

Container

Containers are the data structures used to store objects and data according to the requirement.

Containers can be further classified into 4 types:

1. Sequence Containers : Vector, Deque, List
2. Associative Containers : Set, MultiSet, Map, Multimap
3. Unordered Associated Containers : Unordered set, unordered Map,



Algorithms

STL algorithms offer a wide range of functions to perform common operations on dat

`Sort()`

`reverse()`

`min_element()`

`max_element()`

`count()`

#include <iostream>

#include <bits/stdc++.h>

#include <string.h>

#include <stdio.h>

#include <math.h>
#include <stdlib.h>

vector

header: #include <vector>

Syntax:

vector < data type > v

vector < data type > v(size)

vector < int > v(5)

capacity = 5

size() = 3



* `v.push_back(data)`

(10)

(20)

(30)

(40)

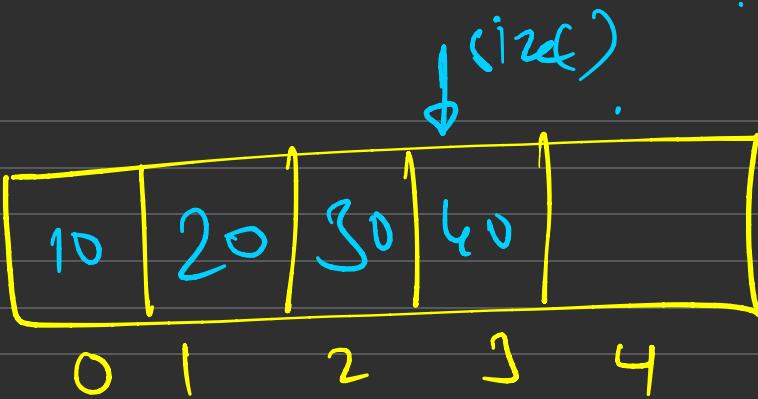
10

[10 | 20] [30]

* `v.pop_back()` → Remove last element

* `v.size()` → size of vector

* `v.empty()` { T if empty
F }



~~size = 0~~

2

pop-back

vector <int> v (5)

v.push-back(20)

v.size() = 6

size++

Pairs

$\langle \text{Val}_1, \text{Val}_2 \rangle$
 $\langle \text{Val}_2, \text{Val}_3 \rangle$

header = include < pair >

Syntax:

pair < dataT₁, dataT₂ > p

p. first = dataT₁

p. second = dataT₂

pair < int, string > p[5]

(Roll, Name)

p[0].first = 1

p[0].second = "Umeshali"

| | | | |
|---|---|---|----|
| 7 | 8 | 9 | 10 |
| 0 | 1 | 2 | 3 |

p[1].first = 5

p[1].second = "Sanchita"

| | | | | |
|------|--------|--------|---|---|
| 1, V | (5, S) | (2, E) | | |
| 0 | 1 | 2 | 3 | 4 |

p[2].first = 2

p[2].second = "Eshita"

| |
|--------------|
| (data, data) |
|--------------|

| | | | | |
|---|---|---|---|----|
| 8 | 9 | 0 | 9 | 10 |
|---|---|---|---|----|

```
for( i=0 ; i < p.size() ; i++ )
```

```
}
```

```
    cout << p(i).first
```

```
    cout << p(i).second
```

```
4
```

Set

- mainly used to store data
- data is in sorted order (increasing)
- duplicate not allowed

Unordered Set

- data can be in any order
- duplicate not allowed

Set

header = include <set>

Syntax :

- set <datatype> s (increasing)
- unordered_set <datatype> us

- `s.size()` → mainly gives no. of elements
 - `s.empty()` → Either Empty
- X —————
- `s.insert(data)` → Insert data in set

`s.insert(5)`

`s.insert(2)`

`s.insert(1)`

`s.insert(4)`

`s.insert(3)`

`s.insert(2)`

ignore

| |
|---|
| 5 |
| 4 |
| 3 |
| 2 |
| 1 |

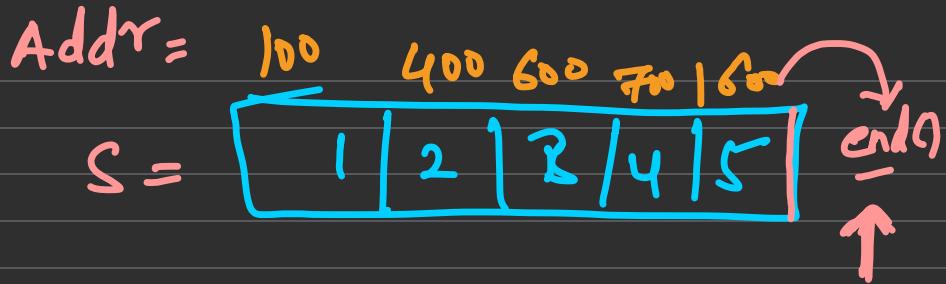
- Traversing set

```
for (auto i : s)  
    cout << i  
}
```

- $s.\text{find}(\text{data})$

```
auto x = s.find(4)
```

```
cout << (x)
```



auto y = s.find(20)

if (y == s.end())
 Not found

else

Multiset

- Allows duplicate
- sorted data

- `s.insert (val)`
 - `s.size()`
 - `s.empty()`
 - `s.find(val)`
- `{ end()`
- `{ Add`

```
for (auto i : s)  
cout << i
```

set <int> s
multiset <int> ms
unordered_set <int> us
-

Unordered Set

unordered_set <int> us

Map

Mainly used to store (key,value) pair

- * map does not allow duplicate key
- * Data is in increasing order → key

header file = #include <map>

Syntax: map<data1, data2> m

Key Value

(duplicate not
allowed)

`map < int, string > m`

`m.insert({2, "Sandali"})`

`m[1] = "Vrushali"`

`m[0] = "Samiksha"`

`(4, Prajyal)`

`(2, Sandali)`

`(1, Vrushali)`

`(0, Samiksha)`

`map`

* Traversing

for (auto i : m)

} cout << i.first → key

cout << i.second → value

}

* find :

mainly check whether given
key is present or not

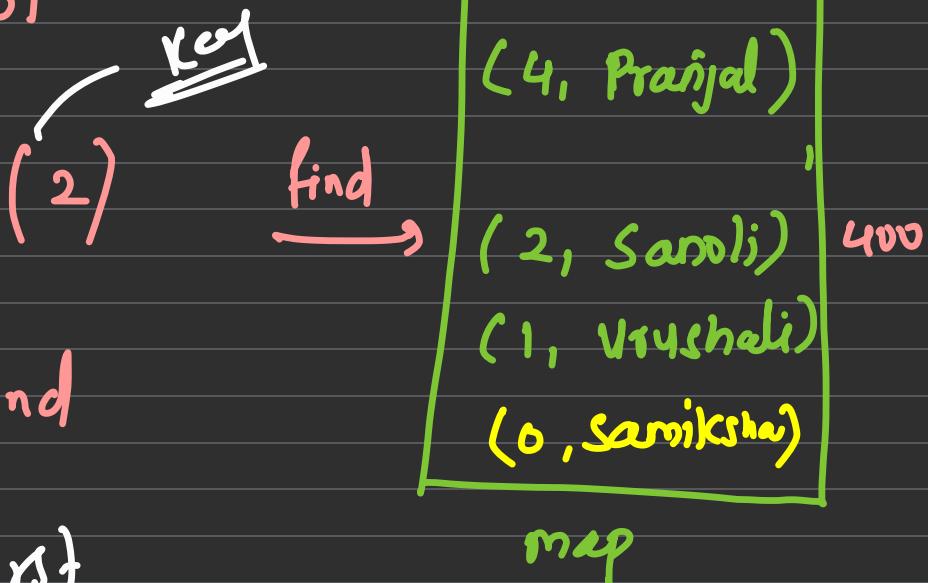
auto find = mp.find(2)

cout << find->first

cout << find->second

(* find). first

(* find). second



auto find = mp.find(20)

if (find == mp.end())

} Not found

γ

* erase()

auto find = mp.find(2)

mp.erase(find)

Unordered Map

Syntax:

unordered_map < data1, data2> um

header:

#include <unordered_map>

Count Occurrence of every element of array

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 3 | 1 | 3 | 6 | 2 | 4 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

map <int, int> m

for (i=0 ; i<8 ; i++)

 m[underline{p[a[i]}]]++

i = 7

m[p(4)]++
0++

(4,1)

(2,1)

(1,1)

(3,2)

(6,2)

(7,1)

map

7 → 1
6 → 2
3 → 2
1 → 1
2 → 1
4 → 1

~~Sort()~~ → mainly used to sort array / string

* int arr [] = { 3, 1, 4, 2, 0, -1 } $N = 5$

→ sort (arr, arr + N)

$O(N \cdot \log N)$

* vector < int > a = { 4, 2, 1, 0, 9 }

→ sort (a.begin(), a.end())

[4 | 2 | 1 | 0 | 9]

* String s = "Pramod"

↑
begin

→ sort (s.begin(), s.end())

end

reverse()

* int arr() = { 1, 2, 3, 4 }

reverse (arr, arr + n)

+ vector < int > a = { 4, 3, 2, 1 }

reverse (a.begin(), a.end())

count()

accumulate() → gives you sum of elements of array

int arr[] = {1, 2, 3, 4}

→ int sum = accumulate(arr, arr+n, 0)

vector<int> a = {1, 2, 3, 4}

→ int sum = accumulate(a.begin(), a.end(), 0)

`min_element()` → Returns min element from array

`int arr[] = { 3, 4, 6, 2, 0, 1 };`

`int min = *min_element(arr, arr+n)`

`int max = *max_element(arr, arr+n)`

 X

vector

`*min_element(arr.begin(), arr.end())`

max_element()

... vector.cpp X

```
vector < int > v={10 , 20 , 30, 40, 50};  
cout << v.size() << endl ;
```

return 0;

int arr[] = { 1, 2, 3, 4 };

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\pranj\Downloads\ControlFlow.cpp> cd "c:\Users\pranj\Downloads\vector"  
5  
PS C:\Users\pranj\Downloads\ControlFlow.cpp>
```

RE
int arr[-2]

```
C:\ Vector.cpp > @ main()
1 #include <iostream>
2 #include <vector>
3 using namespace std ;
4 int main ()
5 {
6     vector < int > v ;
7     v.push_back (10) ;
8     v.push_back (20) ;
9     v.push_back (30) ;
10    v.push_back (40) ;
11    v.push_back (50) ;
12    v.push_back (60) ;
13    cout << v[-1] ;
14 }
15 }
```

$$\text{Cap} = 10$$

10 | 20

10 | 20 | 30 | 40

10 | 20 | 30 | 40 | 50 | 60

0 1 2 3 4 5 C T

10 | 20 | 30 | 40 | 50 | 60

0 1 2 3 4 5

-1

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\pranj\Downloads\STL.cpp> cd "c:\Users\pranj\Downloads"
134229818
PS C:\Users\pranj\Downloads\STL.cpp>
```

40

```
1 #include <iostream>
2 #include <vector>
3 using namespace std ;
4 int main ()
5 {
6     vector < int > v ;
7     v.push_back (10) ;
8     v.push_back (20) ;
9     v.push_back (30) ;
10    v.push_back (40) ;
11    v.push_back (50) ;
12    v.push_back (60) ;
13    cout << v[10] ;
14 }
15 }
```

index = 20

```
1 #include <iostream>
2 #include <vector>
3 using namespace std ;
4 int main ()
5 {
6     vector < int > v ;
7     v.push_back (10) ;
8     v.push_back (20) ;
9     v.push_back (30) ;
10    v.push_back (40) ;
11    v.push_back (50) ;
12    v.push_back (60) ;
13    cout << v[10] ;
14 }
15 }
```

index = 10

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\pranj\Downloads\STL.cpp> cd "c:\Users\pranj\Downloads"
2
PS C:\Users\pranj\Downloads\STL.cpp>
```