Зміст

[pair<type1, type2> 2](#_Toc506768793)

[vector<type1> 3](#_Toc506768794)

[stack<type1> 4](#_Toc506768795)

[deque<type1> 5](#_Toc506768796)

[map<type1, type2> 6](#_Toc506768797)

[set<type1> 7](#_Toc506768798)

[Задачі з e-olymp 8](#_Toc506768799)

# pair<type1, type2>

pair<int, int> *p*;

*p*.first - перший елемент

*p*.second - другйи елемент

Інші, можливі, приклади пар:

pair<double, double>

pair<pair<int, int>, int>

pair< pair<int, int>, pair<int, int> >

pair<long long, vector<int> >

# vector<type1>

vector<int> *m*;

*m*.push\_back(*a*); - добавити в кінець елемент *a*

*m*[*i*] – звернутия до *i*-ого елемента

*m*.size() - розмір вектора

*m*.back() - останній елемент вектора

*m*.clear(); - очистити вектор

*m*.empty() - true якщо вектор пустий, інакше *false*

*m*.begin() і *m*.end() - почак і кінець вектора. Необхідні для сортування

*m*.erase(m.begin()+*i*); - видалити *i*-ий елемент

*m*.erase(m.begin()+*i*, m.begin()+*j*); - видалити елементи від *i*-ого до *j*-ого

Видалення працює за лінійний час, тобто доволі довго.

# stack<type1>

stack<int> *s*;

*s*.push(*a*); - добавити елемент *a*

*s*.size() - розмір стека

*s*.top() - останній елемент стека

*s*.pop() - видалити останній елемент стека

*s*.empty() - true якщо стек пустий, інакше false

# deque<type1>

deque<int>*d*;

*d*.push\_front(*a*); - добавити елемент *a* на початок дека

*d*.push\_back(*a*); - добавити елемент *a* в кінці дека

*d*.front() - перший елемент дека

*d*.back() - останній елемент дека

*d*.pop\_front(); - видалити перший елемент

*d*.pop\_back(); - видалити останній елемент

*d*.size() - розмір дека

*d*.empty() - true якщо дек пустий, інакше false

*d*.clear(); - очистити дек

*d*[*i*] - *i*-ий елемент в деці

# map<type1, type2>

map<int, int> *m*;

map<int, int>::iterator *it*; - ітератор(вказівник) на певний елемент мапи

*m*[*i*] = *a*; - присвоєння елементу мапи з індексом *i* значення *a*

*m*.count(*i*) - перевірка чи використовувався елемент з індексом *і*

*m*.clear(); - очистити мапу

*m*.begin() - початок мапи і заодно вказівник на перший елемент

*m*.end() - вказівник на кінець мапи. Розташований на 1 позицію дальше від останнього елмента мапи

*m*.empty() - true якщо мапа пуста, інакше false

*m*.size() – розмір мапи(кількість елементів, які використовувалися в мапі)

*it*->first - індекс елемента мапи на який вказує *it*

*it*->second - значення елемента мапи на який вказує *it*

# set<type1>

set<int>*s*;

set<int>::iterator *it*; - ітератор(вказівник) на певний елемент сета

multiset<int>*s2*; - сет в якому можуть повторюватися значення

*s*.insert(*a*); - вставити елемент *a* в сет

*s*.begin() - вказівник на початок сета і одночасто і на перший елемент

*s*.end() - вказівник на кінець сета. Розташований на 1 позицію дальше від останнього елмента сета

*s*.erase(*a*) - видалити елемент(всі елементи) *a* в сеті(мультисеті)

*s*.erase(*it*) - видалити елемент, на який вказує вказівник *it*

*s*.find(*a*) - вказівник на елемент *a* в множині, або s.end() якщо такого елемента в сеті немає

*s*.upper\_bound(*a*) - вказівник на перший елемент, який строго більший *a*

*s*.lower\_bound(*a*) - вказівник на перший елемент, який більший або рівний *a*

*s*.size() - розмір сета

*s*.clear(); - очистити сет

*s*.empty() - true якщо сет пустий, інакше false

\**it* - значення самого елемента в сеті

*it*++; - перенести вказівник на наступний елемент в сеті

*it*--; - перенести вказівник на попередній елемент в сеті

Приклад проходу по всьому сетові(знаходення суми всіх елементів) :

for(it=s.begin();it!=s.end();it++)

{

 sum+= \*it;

}

# Задачі з e-olymp

291 – set + deque

555 – set

693 – stack

694 – deque

790 – set

1211 – map

1225 – set

1226 – set

1227 – set

1228 – set

1776 – stack

1868 – map

1871 – stack

1872 – vector

2040 – map(можна простими масивами, але для розуміння краще зробити використовуючи map)

2479 – stack

2661 – set

3004 – set

Примітка: set підрозуміває використання set або multiset, в залежності від задачі