**Лабораторна робота №9**

**Тема:** Особливості розробки класів для роботи з динамічними масивами довільної розмірності з використанням контейнерів vector та algorithm стандартної бібліотеки шаблонів STL.

**Мета:** Набуття навичок студентами при роботі з контейнерами стандартної бібліотеки шаблонів STL.

**Порядок виконання роботи**

1. Проаналізувати структуру контейнеру vector та обґрунтувати вибір методів контейнера для розв’язання задачі відповідно Завдання 1.
2. Розробити метод з використанням контейнера vector як зберігання даних заданого типу відповідно до Завдання 1.
3. Розробити метод з використанням контейнера vector як сховища об’єктів заданого типу відповідно до Завдання 1.
4. Розробити тест для перевірки вірності даної програми.
5. Розробити програму з використанням алгоритмів контейнеру algorithm бібліотеки STL відповідно до Завдання 2.
6. Оформити звіт до лабораторної роботи.

**Завдання 1**

1.Створити об'єкт-контейнер відповідно до варіанта завдання і заповнити його даними, тип яких визначається варіантом завдання.

2.Переглянути контейнер.

3.Змінити контейнер, видаливши з нього одні елементи і замінивши інші.

4.Переглянути контейнер.

5.Створити другий контейнер цього ж класу і заповнити його даними того ж типу, що і перший контейнер.

6.Змінити перший контейнер, видаливши з нього n елементів після заданого і додавши потім у нього всі елементи з другого контейнера.

Варіант 1. типу char з використанням ітератора

Варіант 2. типу long без використання ітератора

Варіант 3. типу int з використанням ітератора

Варіант 4. типу int без використання ітератора

Варіант 5. типу long з використанням ітератора

Варіант 6. типу float без використання ітератора

Варіант 7. типу long з використанням ітератора

Варіант 8. типу float без використання ітератора

Варіант 9. типу double з використанням ітератора

Варіант 10. типу double без використання ітератора

Варіант 11. типу long з використанням ітератора

Варіант 12. типу char без використання ітератора

Варіант 13. типу int з використанням ітератора

Варіант 14. типу int без використання ітератора

Варіант 15. типу float з використанням ітератора

Варіант 16. типу float без використання ітератора

Варіант 17. типу long з використанням ітератора

Варіант 18. типу float без використання ітератора

Варіант 19. типу int з використанням ітератора

Варіант 20. типу double без використання ітератора

Варіант 21. типу char з використанням ітератора

Варіант 22. типу char без використання ітератора

Варіант 23. типу int з використанням ітератора

Варіант 24. типу double без використання ітератора

Варіант 25. типу float з використанням ітератора

Варіант 26. типу double без використання ітератора

Варіант 27. типу long з використанням ітератора

Варіант 28. типу float без використання ітератора

Варіант 29. типу long з використанням ітератора

Варіант 30. типу float без використання ітератора

**Завдання 2**

##### Варіант 1. В одномірному масиві, що складається з n елементів, обчислити:

* суму негативних елементів масиву;
* добуток елементів масиву, розташованих між максимальним і мінімальними елементами;
* упорядкувати елементи масиву по зростанню.

##### Варіант 2. В одномірному масиві, що складається з n елементів, обчислити:

* суму позитивних елементів масиву;
* добуток елементів масиву, розташованих між максимальним по модулю мінімальним по модулю елементами;
* упорядкувати елементи масиву по убуванню.

##### Варіант 3. В одномірному масиві, що складається з n цілих елементів, обчислити:

* добуток елементів масиву з парними номерами;
* суму елементів масиву, розташованих між першим і останнім нульовими елементами;
* перетворити масив таким чином, щоб спочатку розташовувалися всі позитивні елементи, а потім — усі негативні (елементи, рівні 0, вважати позитивними).

##### Варіант 4. В одномірному масиві, що складається з n елементів, обчислити:

* суму елементів масиву з непарними номерами;
* суму елементів масиву, розташованих між першим і останнім негативними елементами;
* стиснути масив, видаливши з нього всі елементи, модуль яких не перевищує 1. Елементи, що звільнилися в кінці масиву, заповнити нулями.

##### Варіант 5. В одномірному масиві, що складається з n елементів, обчислити:

* максимальний елемент масиву;
* суму елементів масиву, розташованих до останнього позитивного елемента;
* стиснути масив, видаливши з нього всі елементи, модуль яких знаходиться в інтервалі [a,b]. Елементи, що звільнилися в кінці масиву, заповнити нулями.

##### Варіант 6. В одномірному масиві, що складається з n елементів, обчислити:

* номер мінімального по модулю елемента масиву;
* суму модулів елементів масиву, розташованих після першого негативного елемента;
* стиснути масив, видаливши з нього всі елементи, величина яких знаходиться в інтервалі [a,b]. Елементи, що звільнилися в кінці масиву, заповнити нулями.

##### Варіант 7. В одномірному масиві, що складається з n цілих елементів, обчислити:

* номер максимального елемента масиву;
* добуток елементів масиву, розташованих між першим і другим нульовими елементами;
* перетворити масив таким чином, щоб у першій його половині розташовувалися елементи, що стояли в непарних позиціях, а в другій половині — елементи, що стояли в парних позиціях.

##### Варіант 8. В одномірному масиві, що складається з n елементів, обчислити:

* номер мінімального елемента масиву;
* суму елементів масиву, розташованих між першим і другим негативними елементами;
* перетворити масив таким чином, щоб спочатку розташовувалися всі елементи, модуль яких не перевищує 1, а потім — всі інші.

##### Варіант 9. В одномірному масиві, що складається з n елементів, обчислити:

* кількість елементів масиву, менших З;
* суму цілих частин елементів масиву, розташованих після останнього негативного елемента;
* перетворити масив таким чином, щоб спочатку розташовувалися всі елементи, що відрізняються від максимального не більше ніж на 20%, а потім — всі інші.

##### Варіант 10. В одномірному масиві, що складається з n цілих елементів, обчислити:

* мінімальний по модулю елемент масиву;
* суму модулів елементів масиву, розташованих після першого елемента, рівного нулю;
* перетворити масив таким чином, щоб у першій його половині розташовувалися елементи, що стояли в парних позиціях, а в другій половині — елементи, що стояли в непарних позиціях.

##### Варіант 11. В одномірному масиві, що складається з n елементів, обчислити:

* суму негативних елементів масиву;
* добуток елементів масиву, розташованих між мінімальним і максимальним елементами;
* упорядкувати елементи масиву по зростанню.

##### Варіант 12. В одномірному масиві, що складається з n елементів, обчислити:

* суму позитивних елементів масиву;
* добуток елементів масиву, розташованих між максимальним по модулю мінімальним негативними елементами;
* упорядкувати елементи масиву по убуванню.

##### Варіант 13. В одномірному масиві, що складається з n цілих елементів, обчислити:

* добуток елементів масиву з непарними номерами;
* суму елементів масиву, розташованих між першим і останнім нульовими елементами;
* перетворити масив таким чином, щоб спочатку розташовувалися всі позитивні елементи, а потім — усі негативні (елементи, рівні 0, вважати негативними).

##### Варіант 14. В одномірному масиві, що складається з n елементів, обчислити:

* суму елементів масиву з непарними номерами;
* суму елементів масиву, розташованих між першим і останнім негативними елементами;
* стиснути масив, видаливши з нього всі елементи, модуль яких перевищує 1. Елементи, що звільнилися в кінці масиву, заповнити нулями.

##### Варіант 15. В одномірному масиві, що складається з n елементів, обчислити:

* максимальний елемент масиву;
* суму елементів масиву, розташованих до останнього негативного елемента;
* стиснути масив, видаливши з нього всі елементи, модуль яких знаходиться в інтервалі [a,b]. Елементи, що звільнилися в кінці масиву, заповнити нулями.

##### Варіант 16. В одномірному масиві, що складається з n елементів, обчислити:

* номер мінімального по модулю елемента масиву;
* суму модулів елементів масиву, розташованих після першого негативного елемента;
* стиснути масив, видаливши з нього всі елементи, величина яких знаходиться за межами інтервалу [a,b]. Елементи, що звільнилися в кінці масиву, заповнити нулями.

##### Варіант 17. В одномірному масиві, що складається з n цілих елементів, обчислити:

* номер мінімального елемента масиву;
* добуток елементів масиву, розташованих між першим і другим нульовими елементами;
* перетворити масив таким чином, щоб у першій його половині розташовувалися елементи, що стояли в непарних позиціях, а в другій половині — елементи, що стояли в парних позиціях.

##### Варіант 18. В одномірному масиві, що складається з n елементів, обчислити:

* номер мінімального по модулю елемента масиву;
* суму елементів масиву, розташованих між першим і другим негативними елементами;
* перетворити масив таким чином, щоб спочатку розташовувалися всі елементи, модуль яких не перевищує 1, а потім — всі інші.

##### Варіант 19. В одномірному масиві, що складається з n елементів, обчислити:

* кількість елементів масиву, менших 7;
* суму елементів масиву, розташованих після останнього негативного елемента;
* перетворити масив таким чином, щоб спочатку розташовувалися всі елементи, що відрізняються від максимального не більше ніж на 30%, а потім – всі інші.

##### Варіант 20. В одномірному масиві, що складається з n цілих елементів, обчислити:

* середній елемент масиву;
* суму модулів елементів масиву, розташованих після першого елемента, більшого нуля;
* перетворити масив таким чином, щоб у першій його половині розташовувалися елементи, що стояли в парних позиціях, а в другій половині — елементи, що стояли в непарних позиціях.

##### Варіант 21. В одномірному масиві, що складається з n елементів, обчислити:

* суму елементів масиву, більших 5;
* добуток елементів масиву, розташованих між максимальним і мінімальними по модулю елементами;
* упорядкувати елементи масиву по зростанню.

##### Варіант 22. В одномірному масиві, що складається з n елементів, обчислити:

* добуток позитивних елементів масиву;
* суму елементів масиву, розташованих між максимальним по модулю та першим нульовим елементами;
* упорядкувати елементи масиву по спаданню.

##### Варіант 23. В одномірному масиві, що складається з n цілих елементів, обчислити:

* добуток елементів масиву з парними номерами, більших 10;
* суму елементів масиву, розташованих між першим нульовим і максимальним елементами;
* перетворити масив таким чином, щоб спочатку розташовувалися всі позитивні елементи, а потім – усі негативні (елементи, рівні 0розташувати в кінці масиву).

##### Варіант 24. В одномірному масиві, що складається з n елементів, обчислити:

* добуток елементів масиву з непарними номерами, чия дробова частина більше 0,5;
* суму елементів масиву, розташованих між першим і останнім негативними елементами;
* стиснути масив, видаливши з нього всі елементи, модуль яких не перевищує 1. Елементи, що звільнилися в кінці масиву, заповнити нулями.

##### Варіант 25. В одномірному масиві, що складається з n елементів, обчислити:

* максимальний елемент масиву;
* суму елементів масиву, розташованих до останнього негативного елемента;
* стиснути масив, видаливши з нього всі елементи, модуль яких знаходиться за межами інтервалу [a,b]. Елементи, що звільнилися в кінці масиву, заповнити одиницями.

##### Варіант 26. В одномірному масиві, що складається з n елементів, обчислити:

* номер мінімального від'ємного елемента масиву;
* добуток модулів елементів масиву, розташованих після першого негативного елемента;
* стиснути масив, видаливши з нього всі елементи, величина яких більше 1. Елементи, що звільнилися в кінці масиву, заповнити одиницями.

##### Варіант 27. В одномірному масиві, що складається з n цілих елементів, обчислити:

* номер максимального парного елемента масиву;
* добуток елементів масиву, розташованих між першим парним і останнім непарним елементами;
* перетворити масив таким чином, щоб у першій його половині розташовувалися елементи, що стояли в непарних позиціях, а в другій половині – елементи, що стояли в парних позиціях.

##### Варіант 28. В одномірному масиві, що складається з n елементів, обчислити:

* номер елемента, який зустрічається найчастіше;
* суму елементів масиву, розташованих між першим нульовим і останнім негативним елементами;
* перетворити масив таким чином, щоб спочатку розташовувалися всі елементи, модуль яких менше 1, а потім – всі інші.

##### Варіант 29. В одномірному масиві, що складається з n елементів, обчислити:

* кількість елементів масиву, більших 4;
* суму дробових частин елементів масиву, розташованих після останнього негативного елемента;
* перетворити масив таким чином, щоб спочатку розташовувалися всі елементи, що відрізняються від максимального більше ніж на 10%, а потім – всі інші.

##### Варіант 30. В одномірному масиві, що складається з n цілих елементів, обчислити:

* мінімальний по модулю елемент масиву;
* суму модулів елементів масиву, розташованих після першого елемента, більшого одиниці;
* перетворити масив таким чином, щоб у першій його половині розташовувалися елементи, що стояли в непарних позиціях, а в другій половині – елементи, що стояли в парних позиціях.

**Приклад 1 обробки vector**

Для даних типу char з використанням ітератора виконати таку послідовність дій:

* описання об'єкту класу;
* ініціалізацію VECTORу;
* визначення мінімального і максимального значення всього масиву та його заданої частини;
* сортування VECTORу;
* визначення на кожному кроці поточного розміру VECTORу;
* доповнення з заданої позиції у VECTOR;
* видалення з заданої позиції у VECTOR;
* сортування заданої частини VECTORу;
* виведення VECTORу.

vector <char> ch; //об"являємо вектор символьного типу

cout << "\nEnter number of element =";

cin >> N;

// заповнюємо вектор символами з порядковими номерами після 60

for(I=0;I<N;I++){

ch.push\_back(rand()%10+60);

}

cout << "\n\nNew vector\n";

for (I = 0; I<ch.size(); I++)

cout << ch[I] << " ";

cout << "\nCurrent size = "<<ch.size();

// сортуємо вектор

sort(ch.begin(),ch.end());

cout << "\nSorted vactor\n";

for (I = 0; I<ch.size(); I++)

cout << ch[I] << " ";

int pos\_ins;

char new\_elem;

cout << "\nEnter position when insert =";

cin >> pos\_ins;

cout << "\nEnter new element value =";

cin >> new\_elem;

vector <char>::iterator ch\_p=ch.begin();

ch\_p = ch\_p+pos\_ins;

// якщо вказано правильне місце вставки, вставляємо масив

if (pos\_ins>=0 && pos\_ins<=ch.size()) {

ch.insert(ch\_p,1,new\_elem);

}

for (I = 0; I<ch.size(); I++)

cout << ch[I] << " ";

cout << "\nEnter position when erase =";

cin >> pos\_ins;

cout << "\nCurrent size = "<<ch.size();

cout << "\nErase element\n";

// якщо вказано правильне місце вставки, вставляємо масив

if (pos\_ins>=0 && pos\_ins<=ch.size()){

ch.erase(ch.begin()+pos\_ins,ch.begin()+pos\_ins+1);

}

for (I = 0; I<ch.size(); I++)

cout << ch[I] << " ";

cout << "\nCurrent size = "<<ch.size();

return 0;

}

**Приклад 2 роботи з vector**

В одномірному масиві, що складається з n елементів, обчислити:

1. Суму негативних елементів масиву.

2.Добуток елементів масиву, розташованих між максимальним і мінімальним елементами.

Впорядкувати елементи масиву по зростанню.

Лістинг програми

#include "stdafx.h"

#include <iostream.h>

#include <vector>

#include <stdlib.h>

#include <stdio.h>

#include <algorithm>

using namespace std;

int main(int argc, char\* argv[])

{

vector <int> v; //об"являємо вектор цілого типу

int I=0, N=0;

cout << "\nEnter number of element =";

cin >> N;

// заповнюємо вектор довільними числами

for(I=0;I<N;I++){

v.push\_back(rand()%10-5);

}

// виводимо вектор на екран

cout << "\n\nNew vector\n";

for (I = 0; I<v.size(); I++)

cout << v[I] << " ";

// знаходимо суму елементів вектора менше нуля

int sum=0;

for (I = 0; I<v.size(); I++)

if (v[I]<0)

sum +=v[I];

cout << "\nsum elementiv below 0 is "<<sum;

// знаходимо добуток елементів масиву,

// розташованих між максимальним і мінімальним елементами.

// об"являємо вказівник на максимальний елемент

vector<int>::iterator max\_it = max\_element( v.begin(), v.end());

// об"являємо вказівник на мінімальний елемент

vector<int>::iterator min\_it = min\_element( v.begin(), v.end());

vector<int>:: iterator p ;

long dob=1;

// визначаємо чи максимальний чи мінімальний елемент знаходиться в векторі першим

if (max\_it>min\_it){

for(p=min\_it;p<=max\_it;p++){

dob \*=\*p;

}

}

else {

for(p=max\_it;p<=min\_it;p++){

dob \*=\*p;

}

}

cout << "\nDobutoc of negative elements is "<< dob<<"\n";

// сортуємо вектор

sort(v.begin(),v.end());

// виводимо вектор на екан

cout << "Sorted vector \n";

for (I = 0; I<v.size(); I++)

cout << v[I] << " ";

Таблиця 1 - Алгоритми бібліотеки стандартних шаблонів

|  |  |
| --- | --- |
| Алгоритм | Призначення |
| adjacent\_find () | Виконує пошук суміжних парних елементів у послідовності. Повертає ітератор першої пари. |
| binary\_search () | Виконує бінарний пошук в упорядкованій послідовності. |
| copy () | Копіює послідовність. |
| copy\_backward () | Аналогічна функції copy(), за винятком того, що переміщає в початок послідовності елементи з її кінця. |
| count () | Повертає число елементів в послідовності. |
| count\_if () | Повертає число елементів в послідовності, що задовольняють деякому предикату. |
| equal () | Визначає ідентичність двох діапазонів. |
| equal\_range () | Повертає діапазон, в який можна вставити елемент, не порушивши при цьому порядок проходження елементів у послідовності. |
| fill ()  fill\_n () | Заповнює діапазон заданим значенням. |
| find () | Виконує пошук діапазону для значення та повертає перший знайдений елемент. |
| find\_end () | Виконує пошук діапазону для підпослідовності. Функція повертає ітератор кінця підпослідовності в середині діапазону. |
| find\_first\_of () | Знаходить перший елемент в середині послідовності, парний елементу в середині діапазону. |
| find\_if () | Виконує пошук діапазону для елемента, для якого певний користувачем унарний предикат повертає істину. |
| for\_earch () | Призначає функцію діапазону елементів. |
| generate ()  generate\_n () | Привласнює елементам у діапазоні значення, що повертаються породжуючою функцією. |
| includes () | Визначає, чи включає одна послідовність всі елементи іншої послідовності. |
| inplace\_merge () | Виконує злиття одного діапазону з іншим. Обидва діапазони повинні бути відсортовані в порядку зростання елементів. Результуюча послідовність сортується. |
| iter\_swap () | Міняє місцями значення, на які вказують два ітератора, що є аргументами функції. |
| lexicographical\_compare () | Порівнює дві послідовності за абеткою. |
| lower\_bound () | Виявляє перше значення в послідовності, що не менше заданого значення. |
| make\_heap () | Виконує пірамідальне сортування послідовності (піраміда, англійською мовою *heap* - повне двійкове дерево, що володіє тією властивістю, що значення кожного вузла не менше значення кожного з його дочірніх вузлів). |
| max () | Повертає максимальне із двох значень. |
| max\_element () | Повертає ітератор максимального елемента в середині діапазону. |
| merge () | Виконує злиття двох упорядкованих послідовностей, а результат розміщає в третій послідовності. |
| min () | Повертає мінімальне із двох значень. |
| min\_element () | Повертає ітератор мінімального елемента в середині діапазону. |
| mismatch () | Виявляє першу розбіжність між елементами у двох послідовностях. Повертає ітератори обох незбіжних елементів. |
| next\_permutation() | Утворює наступну перестановку (permutation) послідовності. |
| nth\_element () | Упорядковує послідовність таким чином, щоб всі елементи, менші заданого елемента Е, розташовувалися перед ним, а всі елементи, більші заданого елемента Е, - після нього. |
| partial\_sort () | Сортує діапазон. |
| partial\_sort\_copy() | Сортує діапазон, а потім копіює стільки елементів, скільки ввійде в результуючу послідовність. |
| partition () | Упорядковує послідовність таким чином, щоб всі елементи, для яких предикат повертає істину, розташовувалися перед елементами, для яких предикат повертає “неправду”. |
| pop\_heap () | Міняє місцями перший і попередній перед останнім елементи, а потім відновлює піраміду. |
| prev\_permutation() | Утворить попередню перестановку послідовності. |
| push\_heap () | Розміщує елемент на кінці піраміди. |
| random\_shuffle () | Безладно перемішує послідовність. |
| remove ()  remove\_if ()  remove\_copy ()  remove\_copy\_if () | Видаляє елементи із заданого діапазону. |
| replace ()  replace\_if ()  replace\_copy ()  replace\_copy\_if () | Заміняє елементи в середині діапазону. |
| reverse ()  reverse\_copy () | Змінює порядок сортування елементів діапазону на зворотній. |
| rotate ()  rotate\_copy () | Виконує циклічне переміщення вліво елементів у діапазоні. |
| search () | Виконує пошук підпослідовності в середині послідовності. |
| search\_n () | Виконує пошук послідовності заданого числа однакових елементів. |
| set\_difference () | Створює послідовність, яка містить ділянки, що розрізняються, двох упорядкованих наборів. |
| set\_intersection() | Створює послідовність, що містить однакові ділянки двох упорядкованих наборів. |
| set\_symmetric\_difference () | Створює послідовність, що містить симетричні ділянки, що розрізняються, двох упорядкованих наборів. |
| set\_union () | Створює послідовність, що містить об'єднання (union) двох упорядкованих наборів. |
| sort () | Сортує діапазон. |
| sort\_heap () | Сортує піраміду усередині діапазону. |
| stable\_sort () | Сортує діапазон. Однакові елементи не переставляються. |
| swap () | Міняє місцями два значення. |
| swap\_ranges () | Міняє місцями елементи в діапазоні. |
| transform () | Призначає функцію діапазону елементів і зберігає результат у новій послідовності. |
| unique () | Видаляє повторювані елементи з діапазону. |
| unique\_copy () | Виявляє останнє значення в послідовності, що не більше деякого значення. |
| upper\_bound () |

**Контрольні запитання**

1. Для чого було створено STL?
2. Що таке контейнери, алгоритми, ітератори?
3. Види контейнерів.
4. Як працюють алгоритми?
5. Види ітераторів.
6. Що таке vector
7. Назвіть відомі вам функції-члени класу vector.
8. Основні операції, які можна виконувати з vector
9. Що таке list?
10. Назвіть відомі вам функції-члени класу list.
11. Наведіть приклад простого списку.