Лекція11

Введення в стандартну бібліотеку шаблонів

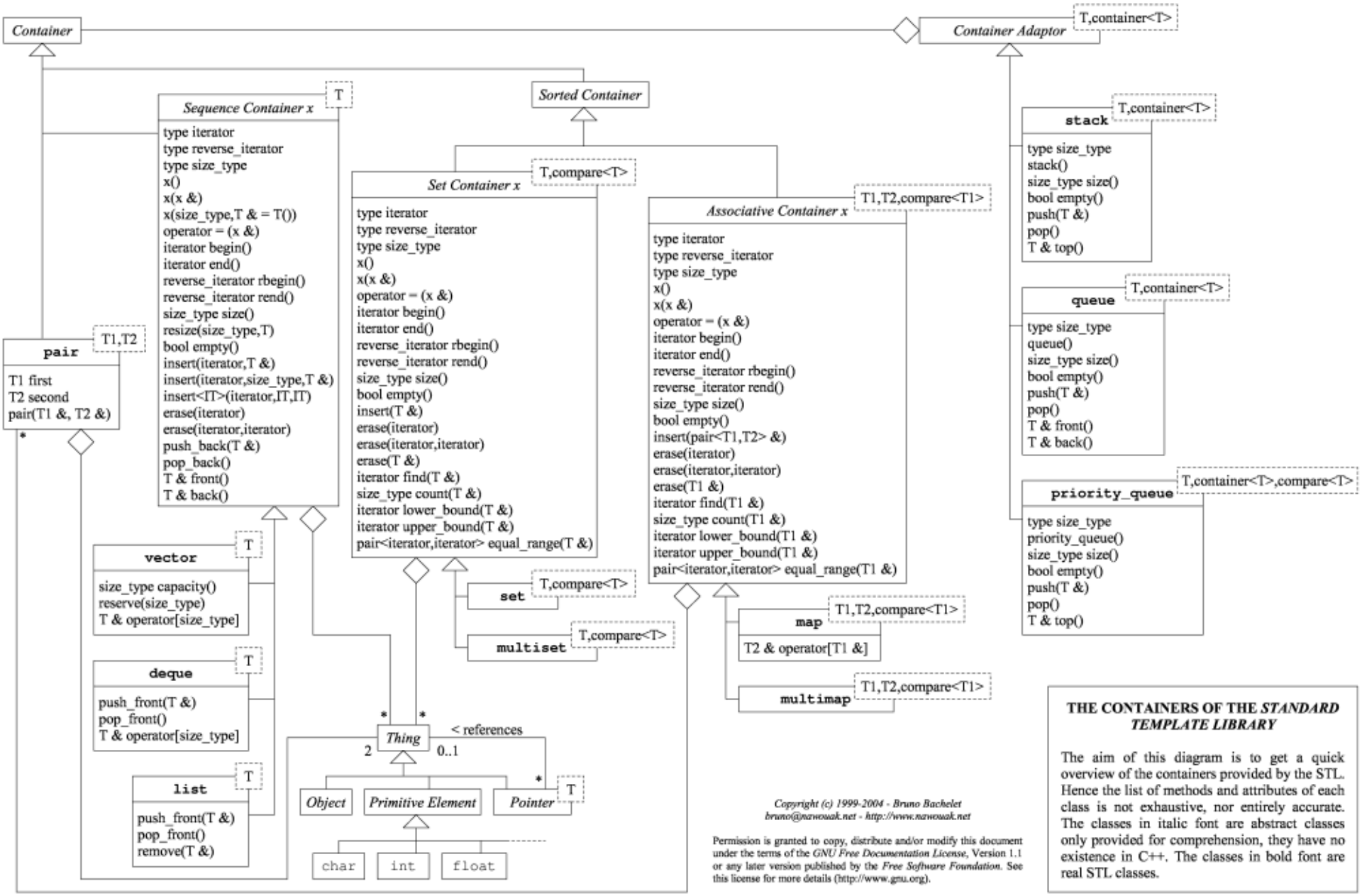
(Standard Template Library — **STL**)

Кафедра комп’ютерних наук. Ст. викл. Міловідов Ю.О.

Бібліотека STL - це набір шаблонних класів і функцій загального призначення

Ядро стандартной библиотеки шаблонов включает три основных элемента: ***контейнеры, алгоритмы и итераторы***. Они работают совместно один с другим, предоставляя тем самым готовые решения различных задач программирования.

Структура бібліотеки



Контейнери - це об'єкти, що містять інші об'єкти.

Існує кілька різних типів контейнерів. Наприклад,

клас **vector** визначає динамічний масив,
клас **queue** створює двосторонню чергу,
клас **list** забезпечує роботу з лінійним списком

Ці контейнери називаються послідовними контейнерами і є базовими в STL.

Крім базових, бібліотека STL визначає асоціативні контейнери, які дозволяють ефективно знаходити потрібні значення на основі заданих ключових значень (ключів).

Наприклад, клас **map** забезпечує зберігання пар "ключ-значення" і надає можливість знаходити значення за заданим унікальному ключу.

Кожен контейнерний клас визначає набір функцій, які можна застосовувати до даного контейнера. Наприклад, контейнер списку включає функції, призначені для виконання вставки, видалення та об'єднання елементів. А стек включає функції, які дозволяють поміщати значення в стек і витягувати їх з стека

Алгоритми обробляють вміст контейнерів. Їх можливості включають засоби ініціалізації, сортування, пошуку і перетворення вмісту контейнерів. Багато алгоритми працюють з заданим діапазоном елементів контейнера

Ітератори

У бібліотеці STL для доступу до елементів в якості посередника використовується узагальнена абстракція, іменована ітератором. Кожен контейнер підтримує «свій» вид ітератора, який являє собою «модернізований» інтелектуальний вказівник, «знає» як отримати доступ до елементів конкретного контейнера. Стандарт С ++ визначає п'ять категорій ітераторів, описаних в таблиці нижче:

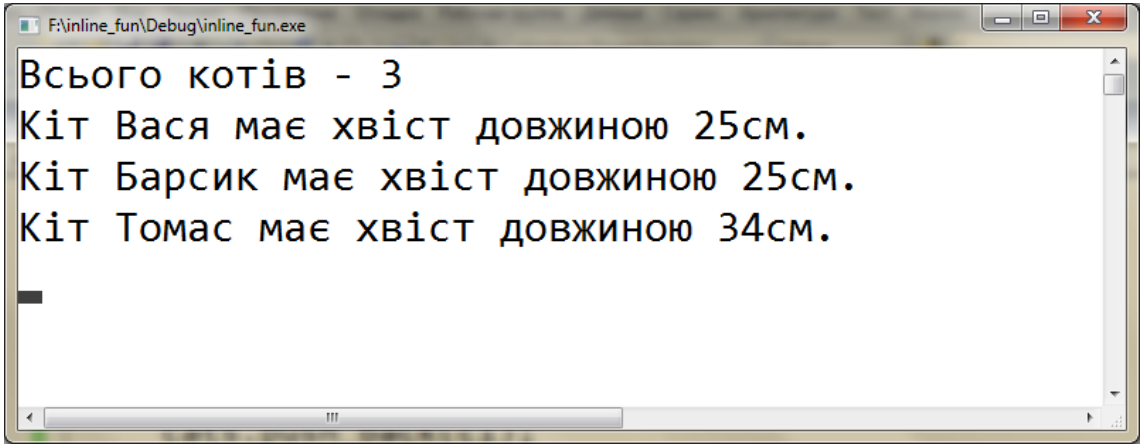
Категорія	Підтримувані операції	Примітка
вхідні	operator ++, operator *, operator->, конструктор копії, operator =, operator ==, operator! =	Забезпечують доступ для читання в одному напрямку. Дозволяють виконати присвоювання або копіювання за допомогою оператора присвоювання і конструктора копії.
вихідні	operator++, operator*, конструктор копії	Забезпечують доступ для запису в одному напрямку. Їх не можна порівнювати на рівність.
однонаправлені	operator ++, operator *, operator->, конструктор копії, конструктор за замовчуванням, operator =, operator ==, operator! =	Забезпечують доступ для читання і запису в одному напрямку. Дозволяють виконати присвоювання або копіювання за допомогою оператора прісваіваіванія і конструктора копії. Їх можна порівнювати на рівність.
двонаправлені	operator ++, operator--, operator *, operator->, конструктор копії, конструктор за замовчуванням, operator =, operator ==, operator! =	Підтримують всі функції, описані для однонапрямлених ітераторів (див. Вище). Крім того, вони дозволяють переходити до попереднього елемента.
довільного доступу	operator ++, operator--, operator *, operator->, конструктор копії, конструктор за замовчуванням, operator =, operator ==, operator! =, operator +, operator-, operator + =, operator- =, operator <, operator>, operator <=, operator> =, operator []	Еквівалентні звичайним вказівниками: підтримують арифметику покажчиків, синтаксис індексації масивів і всі форми порівняння.

Демонстрація базової поведінки вектора.

```
#include "stdafx.h"
#include <iostream>
#include <fstream>
#include <vector>
#include <list>
#include <string>
#include <map>
#include <conio.h>
using namespace std;
```

```
class Cat
{
    string name;
    int tail;
public:
    Cat(string name="Вася", int tail=25)
    {
        this->name=name;
        this->tail=tail;
    }
    void Print()
    {
        cout<<"Кіт "<<name<<" має хвіст довжиною "<<tail<<"см."<<endl;
    }
};
```

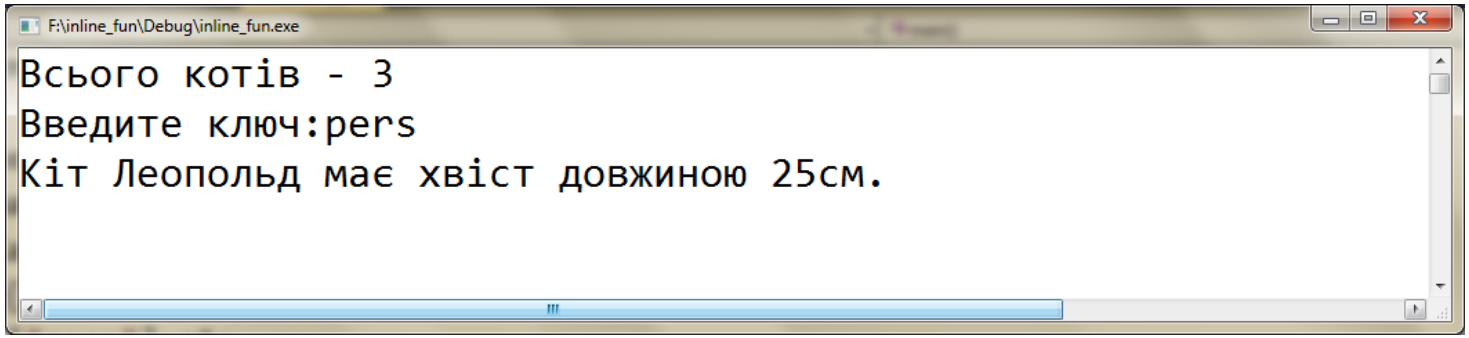
```
void main()
{
    setlocale(LC_ALL, "Russian");
    vector<Cat> cats;
    Cat c1, c2("Барсик"), c3("Томас", 34);
    cats.push_back(c1);
    cats.push_back(c2);
    cats.push_back(c3);
    cout<<"Всього котів - "<<cats.size()<<endl;
    for(int i=0; i<cats.size(); i++)
        cats[i].Print();
    _getch();
}
```


F:\inline_fun\Debug\inline_fun.exe
Всього котів - 3
Кіт Вася має хвіст довжиною 25см.
Кіт Барсик має хвіст довжиною 25см.
Кіт Томас має хвіст довжиною 34см.

Демонстрація асоціативного контейнера

```
void main()
{
    setlocale(LC_ALL, "Russian");
    map<string,  Cat> m;
    Cat c1("Леопольд"), c2("Барсик"), c3("Томас", 34);
    m["brit"]=c3;
    m["pers"]=c1;
    m["siam"]=c2;

    cout<<"Всього котів - "<<m.size()<<endl;
    string key;
    cout<<"Введите ключ:";
    cin>>key;
    m[key].Print();
    _getch();
}
```


F:\inline_fun\Debug\inline_fun.exe
Всього котів - 3
Введите ключ:pers
Кіт Леопольд має хвіст довжиною 25см.