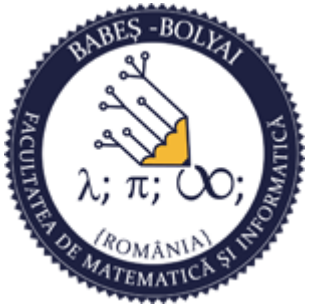
UNIVERSITATEA BABEŞ-BOLYAI
Facultatea de Matematică şi Informatică



Programare orientată obiect

Curs 06

Laura Dioşan

POO

- *Standard Template Library (STL)*
  - Containeri & Iteratori
  - Algoritmi

Containeri - *Remember*

- Un container este un obiect care stochează o colecţie de alte obiecte (elementele sale).
- Containerul:
  - gestionează spaţiul pentru elemente
  - oferă funcţii de acces la elemente, direct sau prin intermediul iteratorilor
- Unele containere au funcţii comune şi împart aceleaşi funcţionalităţi
- Alegerea unui anumit tip de container depinde de:
  - funcţionalităţile oferite de container
  - eficienţa (complexitatea) acestor funcţionalităţi

Containeri - *Remember*

- Containere secvenţiale
  - Vector
  - Listă
  - *Deque* (coadă cu 2 capete)
- Containerere adaptate
  - Stivă
  - Coadă
  - Coadă cu priorităţi
- Containere asociative
  - Multime
  - Mulţime multiplă
  - Dicţionar
  - Dicţionar multiplu

Vectori

- Header *vector*
  - *#include <vector>*
  - *using namespace std;*
- Se folosesc vectori atunci cand elementele lor trebuie parcurse de mai multe ori
- Declarare
  - *vector<elemType> vectorName;*
- Metode
  - adăugare elem: *push_back, operator[], insert, assign, swap*
  - accesare elem: *operator[], at, front, back*
  - eliminare elem: *clear, pop_back, erase*
  - capacitate: *size, empty*
  - iteratori: *begin, end, rbegin, rend*

Iteratori

- Direcţionali
  - Iteratori simpli
  - Iteratori bidirecţionali
- Iteratori de I/O (*stream iterators*)
  - Iteratori pe *stream*-uri de intrare
  - Iteratori pe *stream*-uri de ieşire
  - Iteratori pe *stream*-uri de intrare cu *buffer*
  - Iteratori pe *stream*-uri de ieşire cu *buffer*
- Iteratori de inserţie
  - Interatori de inserţie
  - Iteratori de inserţie la sfârşit (adăugare la sfârşit)
  - Iteratori de inserţie la început (adăugare la început)

Vectori şi iteratori - exemplu

- 06/containers
  - vectorOfIntegers();
  - vectorOfFlowers();
  - vectorOfFlowerAddresses();
  - vectorOfFlowersWithIterator();
  - vectorWithBidirectionalIterators();
  - vectorWithStreamIterators();

  - vectorWithIteratorsAndPredicates();
  - vectorWithIteratorsAndObjectFunctions();

Deque

- Coadă cu două capete (*Double ended queue*)
- Header *deque*
  - *#include <deque>*
  - *using namespace std;*
- Declarare
  - *deque<elemType> dequeName;*
- Metode
  - adăugare elemente: *push_front*, *push_back*, *operator[]*, *insert*, *assign*, *swap*
  - accesare elemente: *operator[]*, *at*, *front*, *back*
  - eliminare elemente: *clear*, *pop_front*, *pop_back*, *erase*
  - capacitate: *size*, *empty*
  - iteratori: *begin*, *end*, *rbegin*, *rend*

Deque - exemplu

- 06/containers
  - dequeOfInteger();
  - dequeOfFlowers();
  - dequeOfFlowersWithIterator();

Listă

- Header *list*
  - *#include <list>*
  - *using namespace std;*
- Listele se folosesc atunci când se inserează/elimină elemente de la o poziţie dată
- Declarare
  - *list<elemType> listName;*
- Metode
  - adăugare elemente: *push_front*, *push_back, insert, assign, swap*
  - accesare elemente: *operator[], at, front, back*
  - eliminare elemente: *clear, pop_front, pop_back, erase*
  - capacitate: *size, empty*
  - operaţii: *splice, remove, remove_if, unique, merge, sort, reverse*
  - iteratori: *begin, end, rbegin, rend*

Listă - exemplu

- 06/containers
  - listOfFlowersWithIterator();

Stivă

- Header *stack*
  - *#include <stack>*
  - *using namespace std;*
- Declarare
  - *stack<elemType> stackName;*
- Metode
  - adăugare elemente: *push*
  - accesare elemente: *top*
  - eliminare elemente: *pop*
  - capacitate: *size, empty*
  - NU EXISTĂ ITERATORI

Stivă - exemplu

- 06/containers
  - stackOfFlowers();

Coadă

- Header *queue*
  - *#include <queue>*
  - *using namespace std;*
- Declarare
  - *queue<elemType> queueName;*
- Metode
  - adăugare elemente: *push*
  - accesare elemente: *front, back*
  - eliminare elemente: *pop*
  - capacitate: *size, empty*
  - NU EXISTĂ ITERATORI

Coadă - exemplu

- 06/containers
  - queueOfFlowers();

Coadă cu priorități

- Header *queue*
  - *#include <queue>*
  - *using namespace std;*
- Declarare
  - *priority_queue<elemType> queueName;*
- Metode
  - adăugare elemente: *push*
  - accesare elemente: *top*
  - eliminare elemente: *pop*
  - capacitate: *size, empty*
  - NU EXISTĂ ITERATORI

Coadă cu priorități – exemplu

- 06/containers
  - priorityQueueOfFlowers();

Mulțime

- Header *set*
  - *#include <set>*
  - *using namespace std;*
- Stochează elemente unice
- Declarare
  - *set<elemType> setName;*
- Metode
  - adăugare elemente: *insert, swap*
  - eliminare elemente: *clear, erase*
  - capacitate: *size, empty*
  - iteratori: *begin, end, rbegin, rend*
  - observatori: *key_comp, value_comp*
  - operaţii: *find, count, …*

Mulțime – exemplu

- 06/containers
  - setOfFlowers()

Colecție

- Mulţime cu chei multiple
- Header *set*
  - *#include <set>*
  - *using namespace std;*
- Permite chei multiple cu valori egale
- Declarare
  - *multiset<elemType> multisetName;*
- Metode
  - adăugare elemente: *insert, swap*
  - eliminare elemente: *clear, erase*
  - capacitate: *size, empty*
  - iteratori: *begin, end, rbegin, rend*
  - observatori: *key_comp, value_comp*
  - operaţii: *find, count, …*

Colecție – exemplu

- 06/containers
  - multiSetOfFlowers()

Dicționar

- Header *map*
  - *#include <map>*
  - *using namespace std;*
- Container asociativ care stochează elemente formate din combinarea unei chei şi a unei valori
- Declarare
  - *map<keyType, elemType> mapName;*
- Metode
  - adăugare elemente: *insert, swap*
  - eliminare elemente: *clear, erase*
  - capacitate: *size, empty*
  - iteratori: *begin, end, rbegin, rend*
  - observatori: *key_comp, value_comp*
  - operaţii: *find, count, …*

Dicționar - exemplu

- 06/containers
  - mapOfFlowers()

Dicționar multiplu

- Dicţionar cu chei multiple
- Header *set*
  - *#include <map>*
  - *using namespace std;*

- Permite chei multiple cu valori egale

- Declarare
  - *multimap<elemType> mapName;*

- Metode
  - adăugare elemente: *insert, swap*
  - eliminare elemente: *clear, erase*
  - capacitate: *size, empty*
  - iteratori: *begin, end, rbegin, rend*
  - observatori: *key_comp, value_comp*
  - operaţii: *find, count, …*

Dicționar multiplu - exemplu

- 06/containers
  - multipleMapOfFlowers()

Algoritmi

- Sortare
  - crescător -> funcţia *sort(…)*
  - descrescător -> funcţia *reverse*(…)
- Interschimbare
  - Funcţia *swap(…)*
- Copiere
  - Funcţia *copy(…)*
    - Dintr-un
      - container
      - *stream*
    - Într-un
      - alt container
      - alt *stream*
  - Funcţia *remove_copy_if(…)*

Predicate

- Un pointer spre:
  - O funcţie booleană
  - Un obiect al cărui clasă supraîncarcă *operator()*

Algoritmi şi predicate – exemplu

- 06/containers
  - sortVectorOfIntegers();
  - sortVectorOfFlowers();
  - swapVector();
  - copyVector();
  - copyVectorCond();
  - vectorWithStreamIterators();
  - vectorWithIteratorsAndPredicates();
  - vectorWithIteratorsAndObjectFunctions();

Cursul următor

- Relația de moștenire