Tema 3

Precizari generale

Fiecare tema trebuie insotita de un mic exemplu care sa demonstreze utilizarea functionalitatilor implementate.
Urmatoarele cerinte sunt valabile pentru oricare dintre temele listate:

- Alocarea dinamica a memoriei;
- Utilizarea unei conventii de denumire a metodelor si a variabilelor;
- Utilizarea variabilelor si metodelor const, unde este cazul;
- Utilizarea exceptiilor pentru tratarea situatiilor neprevazute;
- Utilizarea assert pentru testarea functionalitatilor;
- Indentarea si comentarea adecvata a codului;
- Tema trebuie sa compileze fara a utiliza anumite flag-uri de compilare (cu exceptia cazurilor in care pentru compilare este necesara o anumita versiune de C++) si sa respecte standardele C++ pentru sintaxa.

Termen predare: 24 mai, ora 23:59

Teme

1. Clasa template pentru array de dimensiune fixa, Array<class T, int N>, care sa ofere urmatoarele functionalitati:

- constructor cu parametru default, care initializeaza toate pozitiile unui array cu valoarea data;
- constructor de copiere si operatorul de atribuire;
- supraincarcarea operatorului +, respectiv -, pentru adunare, respectiv scadere, pe componente;
- combinarea cu un array furnizat, rezultand un nou array de elemente perechi - pe componente (Array<T> combinat cu Array<U> va rezulta Array<pair<T, U>>);
- supraincarcarea operatorului [ ] pentru accesarea elementului de pe pozitia furnizata (readwrite);
- obtinerea numarului de elemente;
- transformarea elementelor prin intermediul unui obiect-functie furnizat (template);
- supraincarcarea operatorului de afisare.

2. Clasa template pentru vector de dimensiune variabila (dar nu nelimitata), Vector<class T>, care sa ofere urmatoarele functionalitati:

- constructor fara parametri, care initializeaza un vector gol;
- constructor de copiere si operatorul de atribuire;

- adaugare de elemente (daca se depaseste capacitatea curenta, se va redimensiona conform unei dimensiuni minime de redimensionare);
- stergere de elemente;
- obtinerea capacitatii curente (capacity);
- posibilitatea rezervarii unei capacitati mai mari (dar nu mai mare decat o dimensiune maxima - max_size)
- supraincarcarea operatorului [ ] pentru accesarea elementului de pe pozitia furnizata (readwrite);
- obtinerea numarului de elemente (size);
- transformarea elementelor prin intermediul unui obiect-functie furnizat (template);
- supraincarcarea operatorului de afisare.

3. Clasa template pentru dictionar (perechi ordonate -dupa cheie- de cheie-valoare), Map<class K, class V>, care sa ofere urmatoarele functionalitati:

- constructor fara parametri, care initializeaza un dictionar gol;
- constructor de copiere si operatorul de atribuire;
- adaugare de perechi cheie-valoare (daca cheia exista deja, se suprascrie valoarea veche);
- stergere de perechi cheie-valoare (dupa cheie);
- stabilirea apartenentei unei chei in dictionar;
- supraincarcarea operatorului [ ] pentru obtinerea valorii asociate unei anumite chei;
- obtinerea numarului de perechi;
- transformarea valorilor prin intermediul unui obiect-functie furnizat (template);
- supraincarcarea operatorului de afisare.

4. Clasa template pentru multiset (multime ordonata in care un element poate aparea de mai multe ori), Multiset<class T>, care sa ofere urmatoarele functionalitati:

- constructor fara parametri, care initializeaza un multiset gol;
- constructor de copiere si operatorul de atribuire;
- adaugare si stergere de elemente din multiset (se sterge prima aparitie);
- obtinerea numarului de aparitii ale unui element;
- stabilirea apartenentei unui element in multiset;
- eliminarea tuturor aparitiilor ale unui element;
- obtinerea numarului de elemente (aparitii distincte);
- transformarea elementelor prin intermediul unui obiect-functie furnizat (template);
- supraincarcarea operatorului de afisare.