**Вектор**

Въведение в STL стандартната шаблонна библиотека

**1. STL (Standard  Template** **Library)**

Съдържа мощни компоненти, базирани върху шаблони:

**Контейнери** — шаблонни реализации на основните структури от данни.

**Итератори** — аналог на указателите. Предоставят достъп до елементите, които се съхраняват в контейнерите.

**Алгоритми** — търсене, сортиране, манипулиране на данните и т.н.

Предоставя голямо количество компоненти, които са много мощни и лесни за използване.Стандартната шаблонна библиотека предоставя три типа контейнери:

**Последователни контейнери —**линейни структури от данни(вектори, свързани списъци vector, deque, list).

**Асоциативни** **контейнери** —нелинейни структури; Дават  възможност за бързо  търсене  на елементи; Съхраняват  се двойки ключ/стойност (set, map, multiset, multimap).

**Адаптери** — класове, които позволяват модифициране  на поведението на други контейнери (stack, queue, priority\_queue)

**2.** **Вектор-определение**-структура от данни съдържаща краен брой елементи, т.е. при необходимост неговият размер може да се променя по време на изпълнение на програмата. Елементите са разположени последователно в една област от паметта. Има пряк достъп до елементите на вектора чрез оператора за индексиране [].

**3.** **Работа с вектори**

За да можем да работим с вектори трябва да включим заглавния файл vector , т.е.

#include <vector>

**а)**   **Деклариране на вектор**

vector  < тип > име;     //Празен вектор

vector  < тип > име(size);   //Вектор, който има **size** елемента

vector  < тип > име (size,value);     //Вектор, който има **size** елемента, всеки със стойност **value**

***Пример:***

**vector** <**double**> vec(10);

// създава вектор с 10 елемента от тип double , по подразбиране те са със стойност 0

**vector** <**int**> v2;

//вектор с елементи от тип int (неопределен размер)

**vector <char>  v3(10,** **'\*');**

//вектор с 10 елемента от тип char съдържащ 10 символа \*.

**б)Достъп да елементите**

Всеки елемент на вектора си има индекс, като индекса на първия елемент е 0, на втория-1 и т.н.

- Чрез индекс:     **v[i]** //Достъп до елемента с индекс i

- Чрез функцията at:      **v.at(i)** //Достъп до елемента с индекс i

- Достъп до първия елемент:    **v.front()**

-Достъп до последния елемент на вектора v: **v.back()**

vector<int> v(5);  
 for( int i = 0; i < 5; i++ ) v[i]=i;  
 cout << "The first element is " << v.front()  
      << " and the last element is " << v.back() << endl;

 The first element is 0 and the last element is 4

**в) Брой на елементите**  

**v.size()** //връща текущия размер

**г)** **Оператор за присвояване**

**v[i]=value;** //присвоява нова стойност value на елемента с индекс i

**v.front()=value;** //променя стойността на първия елемент

**v.back()=value;** //променя стойността на последния елемент

**v1=v2;**  //вектора v1 става равен на v2

**д)** **Добавяне на елемент в края**

**v.push\_back**(value); //Вмъква в края на вектора елемент със стойност value.

***Пример***: Добавяне на 10 цели числа към вектора vv:

   vector<int> vv;  
   for( int i = 0; i < 10; i++ ) vv.push\_back( i );

Вектора ще съдържа следните елементи:

 0 1 2 3 4 5 6 7 8 9

**ж)** **Изтриване на  елемент**

**v.pop\_back**(); //Изтрива последния елемент

**v.clear**(); //Изтриване на всички елементи

**з) Проверка дали вектора е празен**

**v. empty()** //Връща **true**, ако вектора е празен и **false**в противен случай.

 vector<int> v;  
  for( int i = 0; i < 5; i++ ) {  
    v.push\_back(i);  
  }  
  while( !v.empty() ) {  
    cout << v.back() << endl;  
    v.pop\_back();  
  }

**и)** **Оператори за сравнение**

**>,<,>=,<=,==,!=**

По-голям е вектора с повече елементи, ако броят на елементите е равен – сравняват се техните стойности.

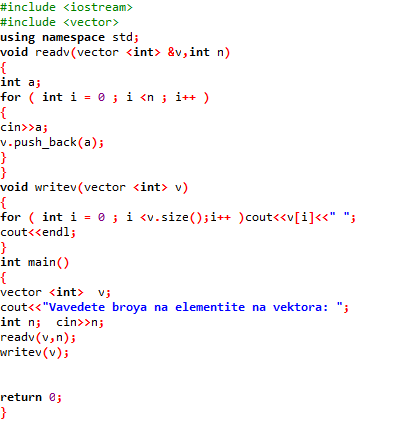
**к)Векторите като параметри на функции**

**vector** **<***тип***>** **&***име*  **//**Параметър променлива

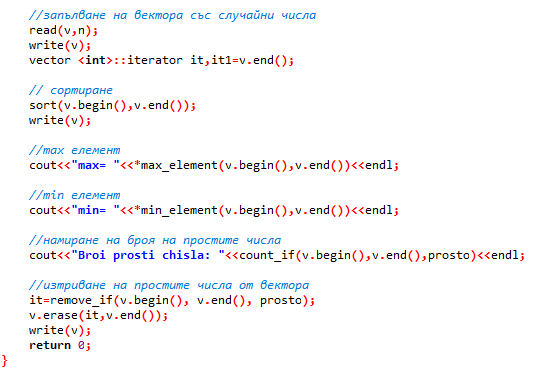
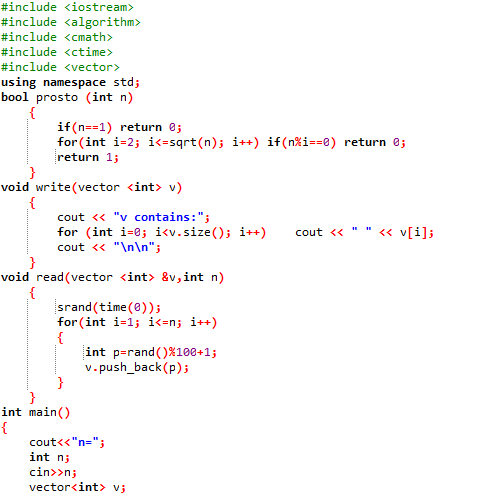
**vector** **<***тип***>** *име* //Параметър стойност

**Пример 1:**

Програма за въвеждане на вектор с **n** елемента и извеждането им на екрана.



**Пример 2:** Програма, която създава вектор от n произволни числа и сортира елементите му, извежда максималния и минималния му елемент, намира броя на простите числа във вектора и изтрива всички прости числа от вектора



Задача 1

Напишете програма, която намира броя на нечетните елементи в един вектор.

Задача 2

Да се състави програма за сливане на 2 вектора с еднаква дължина, като редувате елементите. Изведете резултатният вектор.