vector 的六种 创建和初始化方法_veghlreywg的博客-CSDN博客_vector初始化

https://blog.csdn.net/veghlreywg/article/details/80400382

None Mon Nov, 23 17:53

C++的初始化方法很多，各种初始化方法有一些不同。

(1): vector<int> ilist1;

默认初始化，vector为空， size为0，表明容器中没有元素，而且 capacity 也返回 0，意味着还没有分配内存空间。这种初始化方式适用于元素个数未知，需要在程序中动态添加的情况。

(2): vector<int> ilist2(ilist);

vector<int> ilist2 = ilist;

两种方式等价，ilist2 初始化为ilist 的拷贝，ilist必须与ilist2 类型相同，也就是同为int的vector类型，ilist2将具有和ilist相同的容量和元素

(3): vector<int> ilist = {1,2,3.0,4,5,6,7};

vector<int> ilist {1,2,3.0,4,5,6,7};

ilist 初始化为列表中元素的拷贝，列表中元素必须与ilist的元素类型相容，本例中必须是与整数类型相容的类型，整形会直接拷贝，其他类型会进行类型转换。

(4): vector<int> ilist3(ilist.begin()+2,ilist.end()-1);

ilist3初始化为两个迭代器指定范围中元素的拷贝，范围中的元素类型必须与ilist3 的元素类型相容，在本例中ilist3被初始化为{3,4,5,6}。注意：由于只要求范围中的元素类型与待初始化的容器的元素类型相容，因此迭代器来自不同的容器是可能的，例如，用一个double的list的范围来初始化ilist3是可行的。另外由于构造函数只是读取范围中的元素进行拷贝，因此使用普通迭代器还是const迭代器来指出范围并没有区别。这种初始化方法特别适合于获取一个序列的子序列。

(5): vector<int> ilist4(7);

默认值初始化，ilist4中将包含7个元素，每个元素进行缺省的值初始化，对于int，也就是被赋值为0，因此ilist4被初始化为包含7个0。当程序运行初期元素大致数量可预知，而元素的值需要动态获取的时候，可采用这种初始化方式。

(6):vector<int> ilist5(7,3);

指定值初始化，ilist5被初始化为包含7个值为3的int

内容主要来自于===> C++primer习题集（第五版）P198