第一周直播课（下午）
基础线性数据结构与 STL

—扶苏—
上海财经大学交叉科学研究院
上海洛谷网络科技有限公司

洛谷
www.luogu.com.cn

课前声明

上课的时候专心听讲解。

不直接使用 AI 做题，AI 会做不等于自己会。

不抄袭题解（含对照题解抄一遍），抄对不等于会做。

看完题解后，关闭题解独立练习。

练习中途遇到问题，应当分析题目及自己的思路，而非回忆题解或再次参考题解。

关于 STL 的声明

目前是否可以在正式比赛中使用 STL?

是。目前阶段（CSP-J/S），大多数情况下，可以放心使用 STL。

STL 会不会比我手写的要慢?

目前的大环境下（`-std=c++14`，`-O2`），绝大多数情况下不会。

指针

概念

变量的在物理内存中的存放地址。

地址也是数据。存放地址所用的变量类型有一个特殊的名字，叫做「指针变量」，有时也简称做「指针」。

日常见到的 `int*`，`char*`，某个结构体* 都是指针。它们都指向某一个内存地址，只是后续有效的数据大小不同。

指针

引用

通过在变量名前加 & 获取变量地址。常见于 scanf。

解引用

访问指针变量地址所对应的空间（又称指针所指向的空间），使用 * 符号。

对结构体内变量访问也需要 *，不过有一种取巧的办法是直接使用 -> 符号。

指针

偏移量

数组是一块连续的空间。C/C++中直接访问数组名，得到的是数组的起始地址。

常用 `[]` 来访问数组中某个元素，这个过程间接施加了偏移量。指针允许使用 `+/-` 施加偏移量，一个偏移量就是一个指针对应的数据的大小。

对数组 `a`，`a + 4` 和 `&a[4]` 是等价的。

常见于 `std::sort(a + 1, a + n + 1);`。

空指针

C++11 之后用 `nullptr` 表示空指针，代表指针不指向任何有效内存地址。

指针

new

`type* t = new type();` 建立一个变量，并以指针的形式发送给 `t`。

也可以用于构建数组。`type* t = new type[12345];`

做了解即可。

delete / delete[]

释放一块内存，做了解即可。

迭代器

迭代器是 STL 容器的更抽象的指针。不是指针但和指针类似。

可以用 .begin() 等方法获取指向容器特定位置的迭代器。

可以用 * 对迭代器解引用或用 -> 访问成员变量。

迭代器大都支持自增（++）、自减少（--）操作，部分连续存储容器的迭代器（比如 vector，称为随机访问迭代器）支持 +k 的操作。

C++ string 类

概念复习

方法	含义	例子
size / length	字符串长度	int len = s.length();
clear	清空字符串	s.clear();

+= / append / push_back	向后插入内容	s += "aaa";
		s.append("aaa");
		s.push_back('a');

insert	向指定位置插入字符串	s.insert(3, "aaa");
replace	替换指定位置的字符串	s.replace(3, 17, "aaa");
erase	删去指定区间的字符串	s.erase(3, 7);
substr	拿到某个位置起的子串	s.substr(3, 2);

vector

概念复习

中文名为向量，但是实际上是一个动态数组。

运算符/方法	含义	例子
size	动态数组大小	v.size();
empty	判断数组是否为空	if (v.empty()) ...
push_back	向后插入内容	v.push_back(3);
operator[]	访问某位置元素	int a = v[3];
pop_back	弹出最后一个元素	v.pop_back();
clear	清空动态数组	v.clear();
insert	在某位置插入一个元素（时间复杂度是 $O(n)$ 的）	v.insert(3, 4);
erase	删除某个/某一些元素	
begin, end	迭代器	
front, back	首尾元素	

vector 的遍历

```
std::vector<int> a(n);
for (auto &i : a) std::cin >> i;
for (auto i : a) std::cout << i;
for (auto i = a.begin(); i != a.end(); ++i)
  std::cin >> *i;
for (auto i = a.begin(); i != a.end(); ++i)
  std::cout << *i;
```

类模板

概念复习

类模板（class template）本身不是一个类，而是可以根据 不同数据类型 产生 不同类 的「模板」。

在使用时，编译器会根据传入的数据类型产生对应的类，再创建对应实例。

模板属于 C++ 较为高级的语言特性，仅做简单了解即可。

常常使用一对尖括号 <***> 来使用类模板，后面的例子中将会很常用这个部分。

C++ STL 算法模板库函数

概念复习

方法	含义	例子
min, max	最小/最大值	min(a, b); max(a, b);
swap	交换变量的值	swap(a, b);
sort	排序	sort(a + 1, a + n + 1);
reverse	翻转序列	reverse(a + 1, a + n + 1);
unique	去除相邻重复元素	unique(a + 1, a + n + 1);
nth_element	找出序列第 k 大元素	nth_element(a + 1, a + mid, a + n + 1);
lower_bound upper_bound	找到第一个大于（等于）指定元素的元素位置	lower_bound(a + 1, a + n + 1, k);
next_permutation	下一个排列	do while next_permutation

简单线性数据结构概念

栈：先进后出（FILO）的线性数据结构。

队列：先进先出（FIFO）的线性数据结构。

链表：一些结点穿成的链，每个结点存储了其下一个结点。

双向链表：在链表基础上每个结点存储了其上一个链表结点。

循环链表：首尾相连的链表。

栈、队列

栈：先进后出（FILO）的线性数据结构。

队列：先进先出（FIFO）的线性数据结构。

stack

概念复习

中文名为“栈”，遵循先进后出（FILO）原则。

运算符/方法	含义	例子
push	向栈中推入内容	s.push(3);
pop	弹出最后一个元素	s.pop();
top	查询栈顶元素	int x = s.top();

queue

概念复习

中文名为“队列”，遵循先进先出（FIFO）原则。

运算符/方法	含义	例子
push	向队列中推入内容	q.push(3);
pop	弹出队列头元素	q.pop();
front	查询队列头元素	int x = q.front();

deque

概念复习

中文名为双端队列，顾名思义可以 $O(1)$ 向头尾插入元素。

运算符/方法	含义	例子
size	队列大小	q.size();
empty	判断队列是否为空	if (q.empty()) ...
push_back	向后插入内容	q.push_back(3);
pop_back	弹出最后一个元素	q.pop_back();
push_front pop_front	同理	
clear	清空队列	q.clear();
insert	在某位置前插入一个元素	q.insert(it, 4);
erase	删除某个/某些元素	
front, back, begin, end	迭代器	

链表

概念

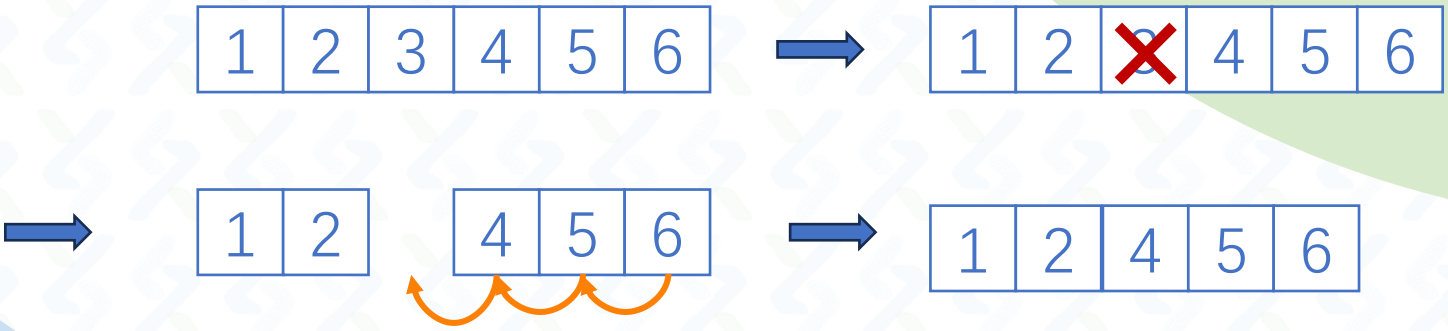
一种用于存储数据的数据结构，通过如链条一般的指针来连接元素。

链表

用途

可能会遇到一些情况，要求我们频繁地插入 / 删除一段序列中某两个元素之间的元素。

数组：（想要删除 3 号元素）非常多的时间用在了数据迁移上。



有没有一种方法可以快速的完成这样的操作?

链表

为什么数组会慢?

在数组中，定位元素的方式是使用下标。

例如：a[3] 代表 a 数组的第 4 个元素。这是一个完全绝对的关系，不受数组内元素之间关系的影响。

我们需要一个元素间的相对的关系。

链表借助的是元素间的相对的关系。

链表

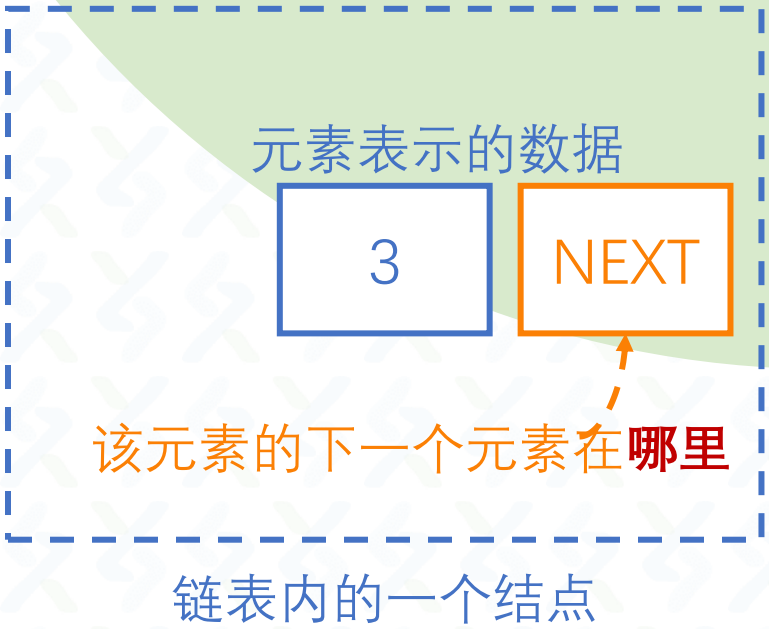
实现

算法竞赛中，一般另开数组来储存元素的相对关系。

如图。对于每一个元素，我们使用 NEXT 数组（一般写作 nxt）来存储某个元素的下一个元素。

上一个元素同理，一般用 prev。

插入，删除元素时，需要注意时序关系。



list

概念复习

中文名为“链表”，$O(1)$ 添加元素，$O(n)$ 查询元素。

注意迭代器失效

运算符/方法	含义	例子
push_back	向后插入内容	l.push_back(3);
pop_back	弹出最后一个元素	l.pop_back();
push_front pop_front	同理	
clear	清空链表	l.clear();
insert	在某位置前插入一个元素（时间复杂度是 $O(1)$ 的）	l.insert(it, 4);
erase	删除某个位置元素	
front, back, begin, end	迭代器	

map

概念复习

中文名“映射表”，$O(\log n)$ 添加元素，$O(\log n)$ 查询元素对应的值。

运算符/方法	含义	例子
operator[]	访问某位置元素	mp["XiaoMing"] = 114;
erase	删除某个元素	mp.erase("XiaoMing");
find	查找某个元素	auto it = mp.find("XiaoMing");

set

概念复习

中文名“集合”，$O(\log n)$ 添加元素，$O(\log n)$ 查询元素是否存在。

性能要求不高时，set 可以当成部分链表用。

运算符/方法	含义	例子
insert	添加元素	st.insert(123);
erase	删除某个元素	st.erase(123);
find	查找某个元素	auto it = st.find(137);
count	计算某个元素出现了多少次	int x = st.count(136);
lower_bound upper_bound	查找第一个大于（等于）给定元素的元素位置	auto it = st.lower_bound(4);

map 与 set

以上两种 STL 均有两个分支：**unordered** 和 **multi-**。

unordered：无序的，使用哈希表结构存储，一般情况下可以 $O(1)$ 执行一次操作。

multi-：多个的，允许放置多个相同元素。

其中 multimap 不常用。

priority_queue

概念复习

中文名为优先队列，是一种堆。

支持在 $O(\log n)$ 时间复杂度内插入一个数、查询最大/最小值（默认最大）、删除最大/最小值。默认是大根堆。

运算符/方法	含义	例子
size	队列大小	q.size();
empty	判断队列是否为空	if (q.empty()) ...
push	向后插入内容	q.push(3);
top	访问最小/最大元素	q.top()
pop	弹出最小/最大元素	q.pop();
clear	清空队列	q.clear();

pair

概念复习

一个存储两个变量的「对」。常用于将关联数据捆绑存储、处理的场景。

可以简单地将 `pair<type1, type2> p;` 理解成

```
struct { type1 first; type2 second; } p;
```

不同的是，`pair` 提供了更健全的赋值、构造、比较大小的方法。一定情况下可以省力。

`map` 中存储的键值对通过 `pair` 向外暴露。