第 12 课
实现与效率

薛浩

xuehao0618@outlook.com

阅读

- Programming Abstraction in C++ *Chapter 11,12*

今日话题

- ADT 实现
- ADT 效率

回顧

重点

学习常见 ADT 的实现算法及其权衡，

不是高级的 C++ 语法

静态内存

- Stack 区域
- 操作系统管理

动态内存

- Heap 区域
- 程序员管理
  - 分配 new
  - 释放 delete

小试牛刀

利用内存检查工具测试程序

动态内存管理

- 在运行时，获取底层存储空间
- 可以按照程序员的意图，随意使用这块区域
- 使用结束后，必须明确释放这块区域

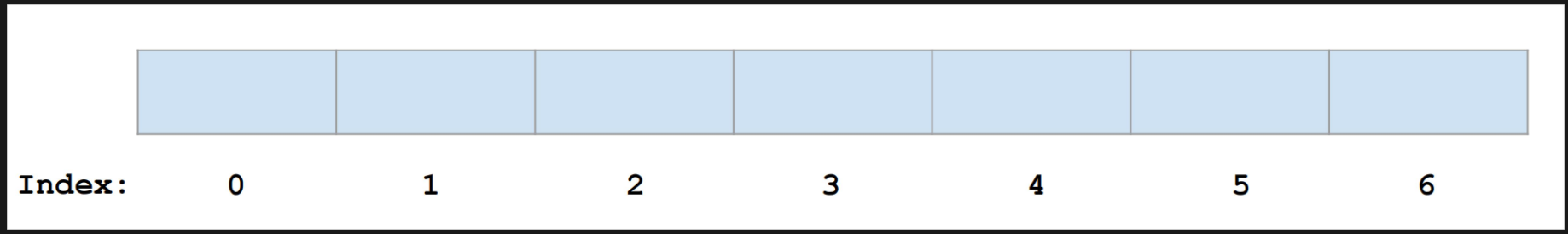
动态内存管理

动态内存管理

```
int * arr = new int[7]; // 获取
```

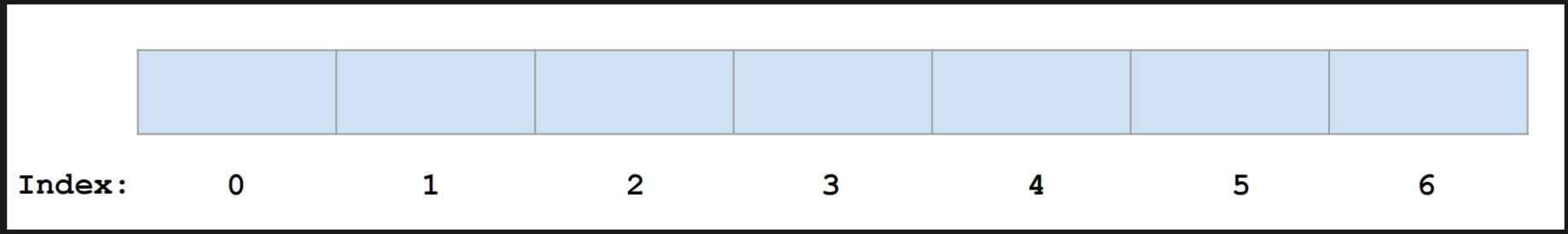
动态内存管理

```
int * arr = new int[7]; // 获取
```



动态内存管理

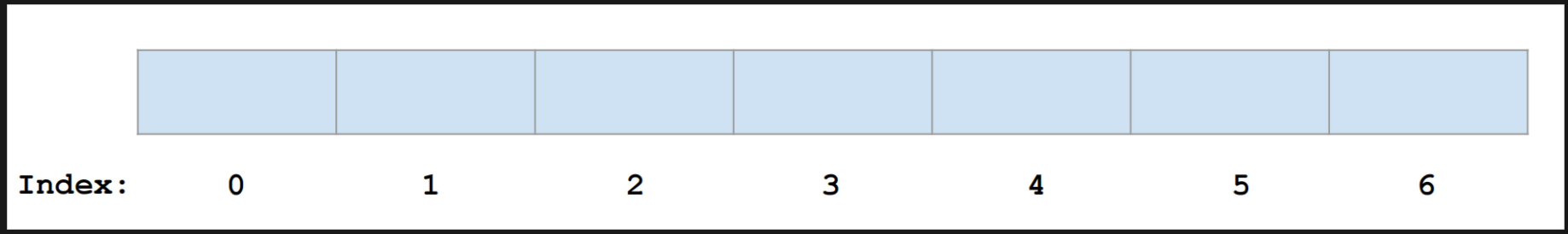
```
int * arr = new int[7]; // 获取
```




```
delete[] arr; // 释放
```

动态内存管理

```
int * arr = new int[7]; // 获取
```




```
delete[] arr; // 释放
```

类型+大小

扩容策略

扩容策略

- 备份原数组指针

扩容策略

- 备份原数组指针
- 重新分配一个两倍容量的新数组

扩容策略

- 备份原数组指针
- 重新分配一个两倍容量的新数组
- 原数组数据拷贝到新数组

扩容策略

- 备份原数组指针
- 重新分配一个两倍容量的新数组
- 原数组数据拷贝到新数组
- 释放原数组内存

类的设计

- 成员变量
  - 类封装的信息，一般不可以被外界直接访问
- 成员函数
  - 对象可以调用的操作逻辑，例如 vec.add(), vec.size() 等
- 构造器
  - 在定义类的变量，即对象时，都是通过构造器来创建的

类的接口与实现

- 接口：定义了什么样的操作可以用于对象修改状态信息
- 实现：定义了这些操作具体的执行逻辑

接口 .H

```
#pragma once
class Point {
public:
    Point();
    Point(int x, int y);
    int getX();
    int getY();
    std::string toString() const;
private:
    int x;
    int y;
    friend bool operator!=(Point p1, Point p2);
    friend bool operator==(Point p1, Point p2);
};
```

实现 .CPP

```
Point::Point(int x, int y) {
    this->x = x;
    this->y = y;
}
int Point::getX() {
    return x;
}
std::string Point::toString() const {
    return "(" + integerToString(x) + "," + integerToString(y)
}
bool operator!=(Point p1, Point p2) {
    return p1.x != p2.x || p1.y != p2.y;
}
```

斯坦福 STACK

小试牛刀

常见一个 PointStack 抽象数据类型

析构函数

- 特殊成员函数，用于清理对象的内存空间
- 对象生命周期结束后，自动执行
- 语法：`~ClassName()`

今日话题

- ~~ADT 实现~~
- ADT 效率

ADT 效率

ADT 效率

不同的 ADT 实现策略，对它操作的时间复杂度有很大影响

优先队列

- 一种类似于队列的抽象数据类型
- 每个元素有一个与之关联的优先级值
- 优先级决定了元素的操作顺序

优先队列接口

- enqueue()
- dequeue()
- peek()

优先队列实现

- 基于数组
- 基于二叉堆

优先队列数组操作

- 父节点 index 对应的子节点

$$左\ 2 * index + 1$$

$$右\ 2 * index + 2$$

- 子节点 index 对应的父节点

$$(index - 1)/2$$

Priority Queue Visualization

今日话题

- ~~ADT 实现~~
- ~~ADT 效率~~

下一次课

- 指针和链表

THE END