C++知识点

1. Namespace 命名空间

在指定命名空间中定义

|  |
| --- |
| 1.命名空间将全局作用域分成不同的部分， |
| 2.不同命名空间中的标识符可以同名而不会发生冲突 |
| 3.命名空间可以发生嵌套 |
| 4.全局作用域也叫默认命名空间 |

1. Extern关键字

extern可以置于变量或者函数前，以标示变量或者函数的定义在别的文件中，提示编译器

遇到此变量和函数时在其他模块中寻找其定义。此外extern也可用来进行链接指定。

也就是说extern有两个作用，第一个,当它与"C"一起连用时，如: extern "C" void fun(int a, int b);则告诉编译器在编译fun这个函数名时按着C的规则去翻译相应的函数名而不是C++的。

第二，当extern不与"C"在一起修饰变量或函数时，如在头文件中: extern int g; 它的作用就是声明函数或全局变量的作用范围的关键字，其声明的函数和变量可以在本模块活其他模块中使用，记住它是一个声明不是定义!也就是说B模块(编译单元)要是引用模块(编译单元)A中定义的全局变量或函数时，它只要包含A模块的头文件即可,在编译阶段，模块B虽然找不到该函数或变量，但它不会报错，它会在连接时从模块A生成的目标代码中找到此函数。

1. Ifdef/ifndef 和 endif

宏命令条件编译，如果当满足或不满足条件时，才对其中语句进行编译。

1. This语句

this 是 [C++](http://c.biancheng.net/cplus/) 中的一个关键字，也是一个 const [指针](http://c.biancheng.net/c/80/)，它指向当前对象，通过它可以访问当前对象的所有成员。只能在类的内部使用。

1. C++中的callback机制

回调函数

1. C++的assert函数

assert宏的原型定义在<assert.h>中，其作用是如果它的条件返回错误，则终止程序执行。

1. 获取Linux系统时间

struct tm {

int tm\_sec; /\* seconds \*/

int tm\_min; /\* minutes \*/

int tm\_hour; /\* hours \*/

int tm\_mday; /\* day of the month \*/

int tm\_mon; /\* month \*/

int tm\_year; /\* year \*/

int tm\_wday; /\* day of the week \*/

int tm\_yday; /\* day in the year \*/

int tm\_isdst; /\* daylight saving time \*/

};

//int tm\_sec 代表目前秒数，正常范围为0-59，但允许至61秒

//int tm\_min 代表目前分数，范围0-59

//int tm\_hour 从午夜算起的时数，范围为0-23

//int tm\_mday 目前月份的日数，范围01-31

//int tm\_mon 代表目前月份，从一月算起，范围从0-11

//int tm\_year 从1900 年算起至今的年数

//int tm\_wday 一星期的日数，从星期一算起，范围为0-6

//int tm\_yday 从今年1月1日算起至今的天数，范围为0-365

//int tm\_isdst 日光节约时间的旗标

struct timeval {

time\_t tv\_sec; /\* seconds (秒)\*/

suseconds\_t tv\_usec; /\* microseconds（微秒） \*/

};

struct timezone {

int tz\_minuteswest; /\* minutes west of Greenwich \*/

int tz\_dsttime; /\* type of DST correction \*/

};

int tz\_minuteswest; /\* 格林威治时间往西方的时差 \*/

int tz\_dsttime; /\* 时间的修正方式\*/

在linux下，常用的获取时间的函数有如下几个： **asctime,  ctime, gmtime, localtime, gettimeofday ，mktime, asctime\_r, ctime\_r, gmtime\_r, localtime\_r**

**time()** 函数获取当前时间，**localtime\_r() localtime()**取得当地目前时间和日期。**asctime()  asctime\_r()** 将时间和日期以字符串格式返回。**ctime()，ctime\_r()** 将时间和日期以字符串格式表示。**mktime()** 将时间结构体struct tm的值转化为经过的秒数。**gettimeofday()** 获取当前时间

8．堆和栈的区别

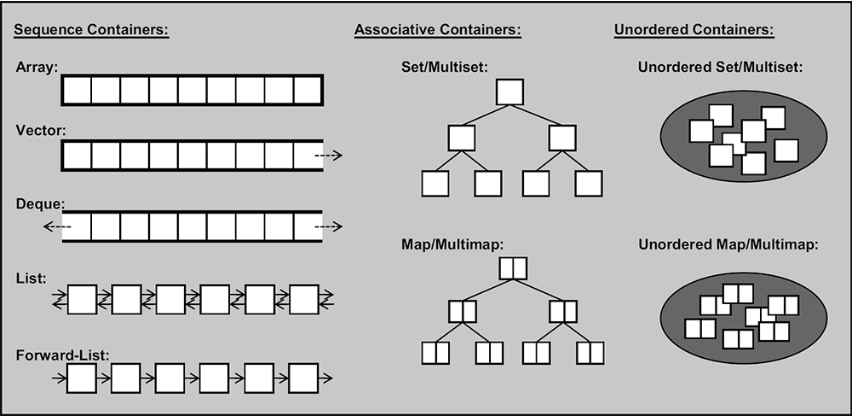
进程在内存中占用的空间包括栈、堆、数据段和代码段。

栈是程序执行前静态分配的地址空间，栈会向下增长。

堆是程序执行时，根据需要动态分配的空间，包括malloc、calloc、realloc函数分配的空间。堆内存由所有线程共享，并会向上增长。

1. linux指令 shell
2. C++11的标准模板库STL（Standard Template Library）

STL容器：均可使用迭代器（Iterator）进行遍历访问和检查，迭代器主要支持两个运算符：自增 ( ++ ) 和解引用（单目 \* 运算符），其中自增用来移动迭代器，解引用可以获取或修改它指向的元素。



序列式容器：

* **数组** ( array ) 定长的顺序表，C 风格数组的简单包装。
* **向量** ( vector ) 后端可高效增加元素的顺序表。
* **双端队列** ( deque ) 双端都可高效增加元素的顺序表。
* **列表** ( list ) 可以沿双向遍历的链表。
* **单向列表** ( forward\_list ) 只能沿一个方向遍历的链表。

关联式容器：通常用红黑树实现

* **集合** ( set ) 用以有序地存储 **互异** 元素的容器。其实现是由节点组成的红黑树，每个节点都包含着一个元素，节点之间以某种比较元素大小的谓词进行排列。
* **多重集合** ( multiset ) 用以有序地存储元素的容器。允许存在相等的元素。
* **映射** ( map ) 由 {键，值} 对组成的集合，以某种比较键大小关系的谓词进行排列。
* **多重映射** ( multimap ) 由 {键，值} 对组成的多重集合，亦即允许键有相等情况的映射。

无序（关联式）容器：基于哈希实现

* **无序（多重）集合** ( unordered\_set / unordered\_multiset ) **C++11** ，与 set / multiset 的区别在与元素无序，只关心”元素是否存在“，使用哈希实现。
* **无序（多重）映射** ( unordered\_map / unordered\_multimap ) **C++11** ，与 map / multimap 的区别在与键 (key) 无序，只关心 "键与值的对应关系"，使用哈希实现。

1. 红黑树基本操作与map和set的实现

一、红黑树的概念

* 红黑树，是一种**二叉搜索树**，但在每个结点上增加一个存储位表示结点的颜色，可以是Red或Black。
* 通过对任何一条从根到叶子的路径上各个结点着色方式的限制，红黑树确保**没有一条路径会比其他路径长出俩倍**，因而是接近平衡的。

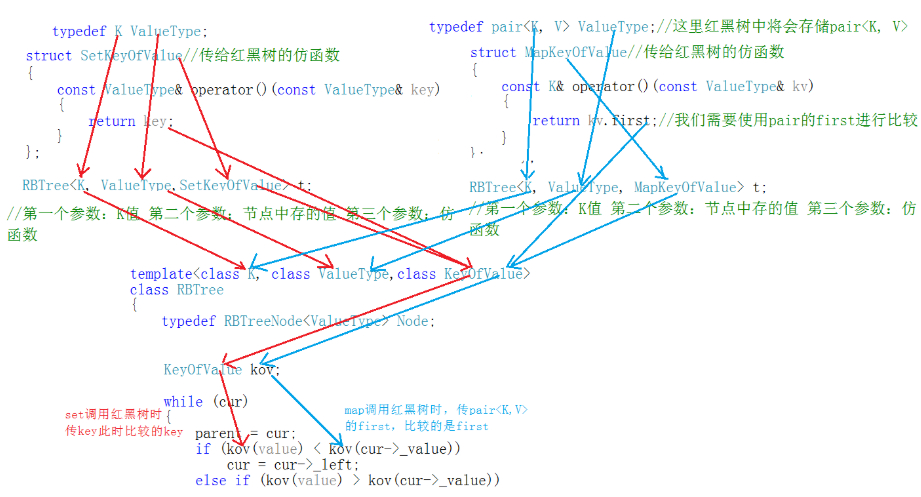
二、红黑树的性质

1. 每个结点不是红色就是黑色
2. 根节点是黑色的
3. 如果一个节点是红色的，则它的两个孩子结点是黑色的
4. 对于每个结点，从该结点到其所有后代叶结点的简单路径上，均包含相同数目的黑色结点
5. 每个叶子结点都是黑色的(此处的叶子结点指的是空结点)

三、红黑树封装实现map和set

1.改造红黑树的节点，通过Map或Set传过来的模板参数来决定节点中是存value值，还是pair<K,V>值。

2.改造红黑树的insert，第一个是插入的值不再仅仅只是键值对，而是看是MyMap使用红黑树时，就传pair<K,V>,MySet使用时就传value；第二个是创建一个仿函数对象，KeyOfValue，通过该对象的返回值来进行储存值大小的比较。



1. c++ callback类

在计算机编程中，回调函数是

* **可执行代码** 作为 **参数** 传入 **其他的可执行代码**
* 并由 **其他的可执行代码** 执行这段 **可执行代码**