CSDN 学院

C++语言基础

迁者 - 贺利坚

http://blog.csdn.net/sxhelijian/

http://edu.csdn.net

本节主题：

常指针和常引用

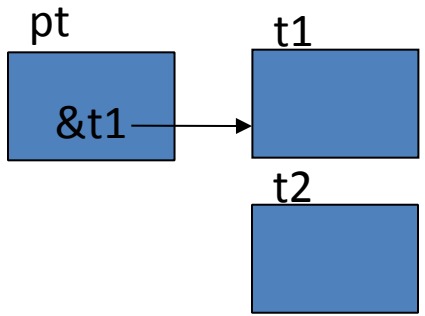
指向对象的"常 指针"

- 将指针变量声明为const型，指针值始终保持为其初值，不能改变。
- 定义指向对象的常指针的一般形式为
  - 类名 * const 变量名；

```
Time t1(10,12,15), t2;
Time * const pt=&t1;  //ptr1是常指针
pt=&t2; //错误，pt应始终指向同一个对象
```

- 常见用途：将一个指针固定地与一个对象联系
  - 将常指针作为函数的形参，目的是不允许在函数执行过程中改变指针变量的值

指针值不可变，
指针指向的值是否可变，取决于指向的对象。



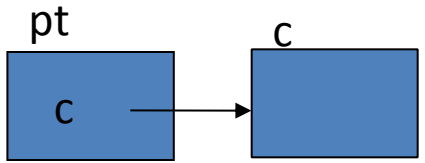

```
void doSomething(Test * const p1)
{
        p1->setX(5);
        p1->printxy( );
}
```

指向常变量的指针变量

指针变量本身的值可变，
指针指向的值不可改变。

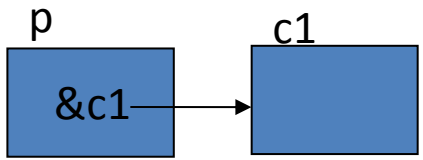
- 定义指向常变量的指针变量的一般形式为
  - const 类名 *变量名/类型名 const *变量名；

```
const char c[]="hello";
const char *pt=c;  //指向了常变量的指针指向常变量
char *pt=c; //错误，只有指向常变量的指针对能指向常变量
```



- 如果一个变量已被声明为常变量/对象，只能用指向常变量/对象的指针指向它，而不能用指向非const型变量/对象的指针去指向它。
- 指向常变量的指针变量可以指向未被声明为const的变量，但不能通过此指针变量改变该变量的值。

```
char c1='a';
const char *p;
p = &c1;   //合法，指向常变量的指针可以指向非const变量
*p = 'b';  //不合法，不能用指向常变量的指针改变变量的值
c1 = 'b';  //合法，c1又不是常变量
```

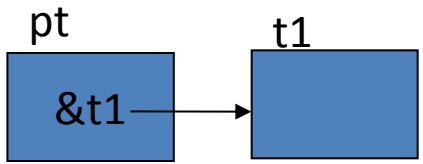


指向常对象的指针变量

指针变量本身的值可变，
指针指向的值不可改变。

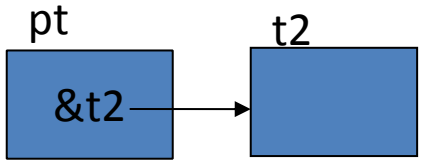
- 定义指向常对象的指针变量的一般形式为
  - const 类名 *变量名/类名 const *变量名；

```
const Time t1(10,12,15);
Time  const *pt=&t1;  //pt指向了常对象
pt->set_time(20,20,20); //错误，指针变量pt指向的是常对象，不能通过pt来改变其值
```



- 如果一个对象已被声明为常对象，只能用指向常对象的指针指向它，而不能用指向非const型变量/对象的指针去指向它。
- 指向常对象的指针变量可以指向未被声明为const的对象，但不能通过此指针变量改变该对象的值。

```
Time t2;
Time  const *pt = &t2; //可以用指向常变量的指针指向非const变量
pt->set_time(20,20,20); //错误，尽管t2.set_time(20,20,20)可以
```



指向常对象的指针变量的用法

- **指向常对象的指针最常用于函数的形参**，以保护形参指针所指向的对象在函数执行过程中不被修改。
- 做法
  - 当希望在调用函数时对象的值不被修改，就应当把形参定义为指向常对象的指针变量，同时用对象的地址作实参(对象可以是const或非const型)。

```
int main()
{
   Test t1(3,5);
   Test t2[N];
   doSomething1(&t1);
   doSomething2(t2, N);
}
```

```
void doSomething1(const Test *p1)
{
   p1->setX(5);  //非法!
   p1->printxy( );
}
```

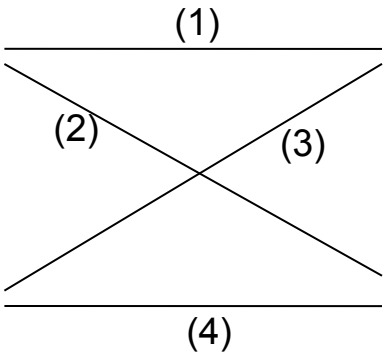
```
void doSomething2(const Test p2[], int n)
{ … … }
```

```
void doSomething3(const Test *p3, int n)
{ … … }
```

指向const型对象的指针 - 形参 - 实参

```
void doSomething(Test *p1)
{
   p1->setX(5);
   p1->printxy( );
}
```

```
void doSomething(const Test *p1)
{
   p1->setX(5);
   p1->printxy( );
}
```




```
int main()
{
   Test t1(3,5);
   doSomething(&t1);
}
```

```
int main()
{
   const Test t1(3,5);
   doSomething(&t1);
}
```

	形参	实参	合法?	可改变值?
(1)	指向非const型变量的指针	非const变量的地址	合法	可以
(2)	指向非const型变量的指针	const变量的地址	非法	/
(3)	指向const型变量的指针	非const变量的地址	合法	不可
(4)	指向const型变量的指针	const变量的地址	合法	不可

对象的常引用

- 回顾：引用
  - 一个变量的引用就是变量的别名。
  - 变量名和引用名都指向同一段内存单元。
- 函数的形式参数可以是对象的引用，例……
  - void fun(Time &t);
- 如果**不希望在函数中修改实参的值**，可将形参声明**常引用**
  - void fun(const Time &);
- 提倡：用常引用作函数参数
  - 这样既能保证数据安全，使数据不能被随意修改；
  - 在调用函数时又不必建立实参的拷贝，可以提高程序运行效率。

```
int a;     //定义a是整型变量
int &b=a;  //声明b是a的引用
a=20;
cout<<b<<endl;
```

```
void doSomething(const Test &r)
{
   r.setX(5);  //非法
   r.printxy( );
}
```

```
int main(void)
{
   Test t1(3,5);
   doSomething(t1);
 }
```

const型数据的小结

形 式	含 义
Time const t1; const Time t1;	t1是**常对象**，其值在任何情况下都不能改变
void fun( ) const	fun是Time类中的**常成员函数**，可以引用，但不能修改本类中的数据成员
Time * const p;	p是**指向Time对象的常指针**，p的值(即p的指向)不能改变
const Time *p; Time const *p;	p是**指向Time类常对象的指针**，其指向的类对象的值不能通过指针来改变
const Time &t1=t;	t1是Time类**对象t的引用**，二者指向同一段内存空间

CSDN 学院

THANKS

本课程由 迁者-贺利坚 提供

CSDN网站：www.csdn.net
企业服务：http://ems.csdn.net/
人才服务：http://job.csdn.net/
CTO俱乐部：http://cto.csdn.net/
高校俱乐部：http://student.csdn.net/
程序员杂志：http://programmer.csdn.net/

CODE平台：https://code.csdn.net/
项目外包：http://www.csto.com/
CSDN博客：http://blog.csdn.net/
CSDN论坛：http://bbs.csdn.net/
CSDN下载：http://download.csdn.net/