动态分配 const 对象（C++）

创建

在 C++中，允许动态创建 const 对象，格式如下：

const int *p = new const int(128);

与其他常量一样，动态创建的 const 对象必须在创建时初始化，并且初始化后，其值不能改变。

删除

尽管不能改变 const 对象的值，但可以删除动态创建的 const 对象，格式如下：

delete p;

这个和普通的对象一样，可以对其进行删除操作。

应用场景举例

1、加载配置文件

从配置文件读入的数据可以用来初始化 const 对象，供后续程序使用。

伪代码如下：

```
int num;
... //读取配置文件，并将配置数据填充到 num
const int *pNum = new const int(num); // 用 num 初始化 const 对象
cout<<*pNum<<endl; //使用 const 对象
...
delete pNum;
```

2、创建数组

当数组的大小依赖于某些动态因素时（比如配置文件等），可以考虑用 const 对象。

伪代码如下：

```
int num;
... //获取 num 的值
const int *pNum = new const int(num); // 用 num 初始化 const 对象
unsigned char _data[*pNum]; //创建数组
```

...

delete pNum

示例代码如下：

```
#include <iostream>

using namespace std;

int main()
{
	int num;
	cin>>num;
	const int *pNum = new const int(num);
	int arr[*pNum];
	for(int i=0;i<*pNum;++i) arr[i] = i;
	for(int i=0;i<*pNum;++i) cout<<arr[i]<<" ";
	cout<<endl;
	return 0;
}
```

当然还有很多其它场景，我暂时想到了这些，这里记录下来，方便以后查阅。