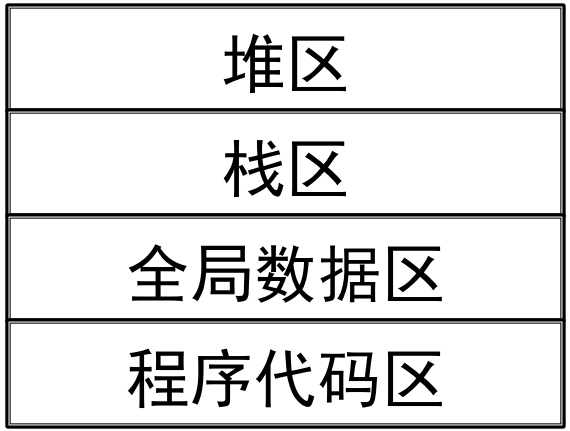
变量的存储类型

杨振平

变量的存储类型

- 不同的变量所分配的存储区域也不同，这就是变量的存储类型。

（1）C++程序运行时使用的内存区域



存放动态分配的数据

存放局部数据。如局部变量

存放全局数据和静态数据，如全局变量

存放程序的各个函数的代码

（2）变量的存储类型

变量的存储类型是变量在内存中存储的方式，根据变量的存储类型，可以知道变量的作用域和生存期。这里，介绍4个存储类型，分别是auto（自动类），register（寄存器类），static（静态类）和extern（外部类）。

在c++中定义一个变量的完整形式是：

<**存储类型**> <**数据类型**> <**变量名**>;

变量的存储类型（续）

- 自动变量–用auto修饰（默认的定义方式）。

  如：定义一个局部变量i。

  auto int i; 与 **int** i; **是**相同的。

说明：自动变量在其定义块（函数或复合语句）开始执行时分配空间，在块执行结束时释放空间。所以自动变量的生命期开始于块的执行，终止于块的结束。

- 寄存器变量–用register修饰

  将尽可能存放在CPU的寄存器中，以提高程序的运行效率

注意，仅局部变量和形参可作为寄存器变量。

变量的存储类型（续）

- 静态变量-用static修饰
  - 静态变量分配在全局数据区中，定义时系统将提供默认的初始值。
  - 静态变量在编译时分配存储空间，在整个程序执行结束后释放存储空间。所以，静态变量具有全局生命期。
  - 根据声明的位置不同，静态变量又分为静态局部变量和静态全局变量。
  - 静态局部变量是在“块”中定义的静态变量。它具有局部作用域，却有全局生命期。在“块”执行结束后，该静态局部变量并不释放（其值依旧存在），以便下次调用时可继续使用。

变量的存储类型（续）

- 外部变量-用extern修饰

  如果在一个源文件中定义的全局变量要在其它源文件中使用，则在使用前应该用extern进行声明，表示该全局变量不是在本文件中定义的。

  例如：在1.cpp文件中定义全局变量

  ```
  int  Dimension=100;
  ```

  如果在2.cpp文件中使用，这时，应在2.cpp文件中声明如下：

  ```
  extern int Dimension;
  ```

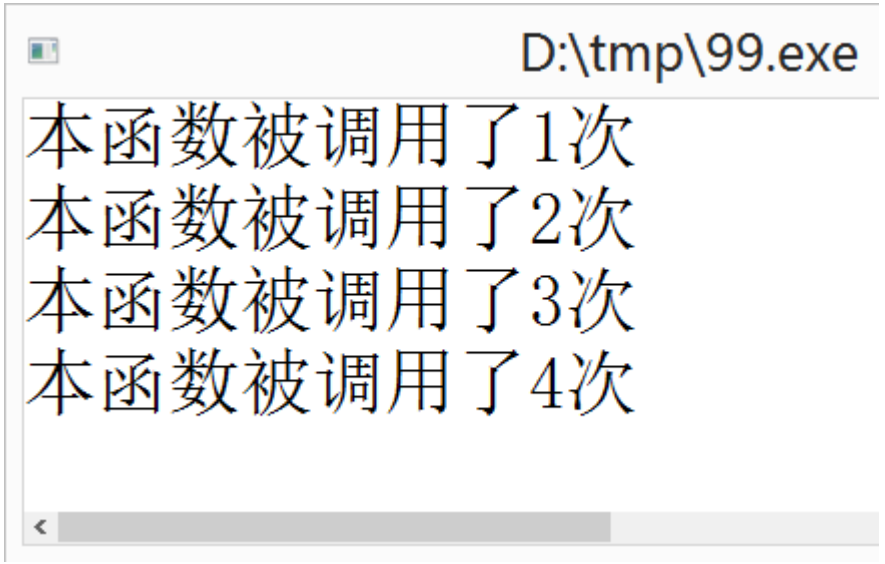
静态全局变量：

全局变量可以在其它源文件中使用。如果在全局变量前加上static修饰符，则成为静态全局变量。静态全局变量只能在本文件中使用。

例如：函数调用计数器。使用静态局部变量统计某个函数被调用的次数。

```
void fun()
{  static int n=0; //局部静态变量
   n++;
   cout<<“本函数被调用了”<<n<<“次”<<endl;
}
```

```
int main()
{
    int i;
    for(i=1;i<=3;i++)
     fun();
    fun();
    return 0;
}
```



如果将函数fun中语句static int n=0;更改为 int n=0;
程序的运行结果有何变化？

```
void fun()
{
   int n=0; //自动变量
   n++;
   cout<<“本函数被调用了”<<n<<“次”<<endl;
}
```

```
int main()
{
    int i;
    for(i=1;i<=3;i++)
     fun();
    fun();
    return 0;
}
```