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：

构造函数

对象的初始化

- 在建立一个对象时，常常需要作某些初始化的工作

```
int i=5, count(0);
double pi=3.1415926;
char s[]={'I',' ','l','o','v','e',' ','u'};
string str="I love u!";
struct Student
{
   int no;
   string name;
   char sex;
   int age;
};
Student stu1={10001,"Zhang Xi",'M', 19};
```

```
error: in C++98 't1' must be initialized by
      constructor, not by '{...}'
error: could not convert '{14, 56, 30}'
      from '<brace-enclosed initializer list>'
      to 'Time'
```

```
#include<iostream>
using namespace std;
class Time
{
private:
   int hour;
   int minute;
   int sec;
};
int main( )
{
   Time t1={14,56,30};
   return 0;
}
```

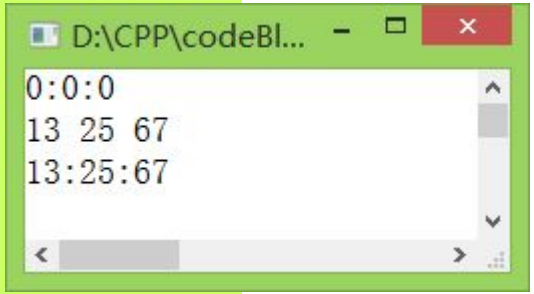
用构造函数初始化对象

构造函数(constructor)是一种特殊的成员函数

- 构造函数不需要用户来调用它，而是在建立对象时自动执行。
- 构造函数的名字必须与类名同名，而不能由用户任意命名。
- 构造函数不具有任何类型，不返回任何值。
- 构造函数的功能是由用户定义的，用户根据初始化的要求设计函数体和函数参数。

```
#include <iostream>
using namespace std;
class Time时间类
{
public:
   Time( ) //类内定义构造函数
   {
      hour=0;
      minute=0;
      sec=0;
   }
   void set_time( );
   void show_time( );
private:
   int hour;
   int minute;
   int sec;
};
```

```
void Time::set_time( )
{
   cin>>hour;
   cin>>minute;
   cin>>sec;
}
void Time::show_time( )
{
   cout<<hour<<":"<<minute<<":"<<sec<<endl;
}
int main( )
{
   Time t1;
   t1.show_time( );
   Time t2;
   t2.set_time( );
   t2.show_time( );
   return 0;
}
```



在类外定义构造函数

```
class Time
{
public:
      Time( );
      void set_time( );
      void show_time( );
 private:
      int hour;
      int minute;
      int sec;
};
```

```
Time::Time( )
{
  hour=0;
  minute=0;
  sec=0;
}
```

带参数的构造函数

```
class Time
{
public:
   Time();
   Time(int,int,int);
   void show_time();
private:
   int hour;
   int minute;
   int sec;
};
```

```
Time::Time()
{
   hour=0;
   minute=0;
   sec=0;
}

Time::Time(int h,int m,int s)
{
    hour=h;
    minute=m;
    sec=s;
}
```

```
int main()
{
   Time t1; //不可t1();
   t1.show_time();

   Time t2(15,39,59);
   t2.show_time();
   return 0;
}
```

构造函数的重载

- 函数名相同；
- 函数参数个数或参数类型不同。

用参数初始化表对数据成员初始化

- 方便、简练的写法：用参数初始化表

```
class Time
{
public:
  Time( ):hour(0),minute(0),sec(0){};
  Time(int h,int m,int s):hour(h),minute(m),sec(s) {};
  ……
};
```

- 也可以在类外用参数初始化表定义构造函数。

```
Time::Time(int h,int m,int s):hour(h), minute(m), sec(s){};
```

- 不在函数体内对数据成员初始化，而是在函数首部实现。
- 当需要初始化的数据成员较多时更显其优越性

讨论一个方案

```
class Time
{
public:
   Time( );
private:  ……
};
Time::Time( )
{
   char c1,c2;
   cout<<"请输入时间(格式hh:mm:ss)";
   while(1)
   {
      cin>>hour>>c1>>minute>>c2>>sec;
      if(c1!=':'||c2!=':')
         cout<<"格式不正确，请重新输入"<<endl;
      else if (!is_time(hour,minute,sec))
         cout<<"时间非法，请重新输入"<<endl;
      else
         break;
   }
}
```

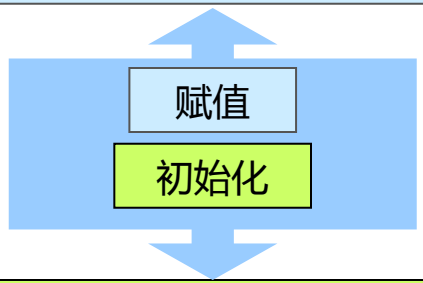
- 行不行？
- 好不好？

- 在构造函数的函数体中不仅可以对数据成员赋初值，而且可以包含其他语句；
- 一般不提倡在构造函数中加入与初始化无关的内容，以保持程序的清晰。
- 在成员函数中慎用输入输出

类的声明和对象的定义

```
class Student
{
private:
   int num;
   char name[20];
   char sex;
public:
   void set_data(int n,  char *p,char s)
   {
      num=n;
      strcpy(name,p);
      sex=s;
   }
   void display( )
   {
      cout<<"num: "<<num<<endl;
      cout<<"name: " <<name<<endl;
      cout<<"sex: " <<sex<<endl<<endl;
   }
};
```

```
int main()
{
   Student stud1,stud2;
   stud1.set_data(1,"He",'f');
   stud2.set_data(2,"She",'m');
   stud1.display();
   stud2.display();
   return 0;
}
```




```
Student stud1={1,"He",'f'};
Student stud2(2,"She",'m');
```

用构造函数初始化对象再例

```
class Student
{
private:
   int num;
   char name[20];
   char sex;
public:
   Student(int n, char *p,char s)
   {
      num=n;
      strcpy(name,p);
      sex=s;
   }
   void display( )
   {
      cout<<"num: "<<num<<endl;
      cout<<"name: " <<name<<endl;
      cout<<"sex: " <<sex<<endl<<endl;
   }
};
```

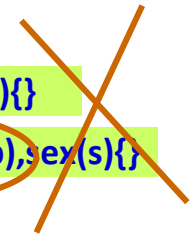
```
int main()
{
   Student stud1(1,"He",'f');
   Student stud2(2,"She",'m');
   stud1.display();
   stud2.display();
   return 0;
}
```

```
Student::Student(int n, char *p,char s)
{
   num=n;
   strcpy(name,p);
   sex=s;
}
```

```
Student::Student(int n, char *p,char s):num(n),name(p),sex(s){}
```

```
Student::Student(int n, char *p,char s):num(n),strcpy(name,p),sex(s){}
```

```
//改进方案
class Student
{
private:
   int num;
   string name;
   char sex;
... ...
}
```



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/