第7章　数据的抽象与封装

1、实体、对象与类的概念	2、类的定义	3、对象声明与引用	4、私有、公有与保护
5、日期类的设计	6、两种程序设计思想	7、汽车类的设计	8、几何图形圆类的设计
9、构造函数的定义	10、重载构造函数	11、析构函数的定义	12、整数翻译函数
13、实际意义的析构函数	14、Person类的设计	15、对象与指针	16、this指针

能翻译整数的机器人类的修改

- 机器人的特征
  - 姓名
  - 型号
  - 整数：待翻译的整数
  - 翻译的英文句子字符串：字符指针，句子有长有短
  - ……
- 机器人的功能
  - 翻译整数函数：形成英文字符串，并将字符串指针返回
  - 构造函数
  - 设置函数
  - 输出英文句子函数
  - 析构函数
  - ……

机器人类的修改定义

```
class robot
{
        char name[20];      //机器人姓名
        char type[20];      //机器人型号
        int num;        //待翻译的整数
        char *ps;       //指向英文字符串
public:
        robot()//构造函数
        {       strcpy(name,"XXXXXX");
                strcpy(type,"XXXXXX");
                num=0;
                ps=new char[5];
                strcpy(ps,"zero");
        }
        void set(char n[],char t[],int m);          //设置修改数据
        char *out(int a);//英文中每三位数读法相同，所以定义out函数翻译小于1000的整数
        char *tran_int(int n);//将1至1999999999的整数翻译成英文句子
        void print_num();//输出整数及其英文句子
        ~robot(){ delete [ ] ps; };    //析构函数 释放构造函数和set函数中动态申请的空间
};
```

```
//英文中每三位数读法相同，所以定义out函数对每组三位数进行处理
char *robot::out(int a)
{   char k[1000]="";
    int b=a%100;
    //若百位不为零，输出百位数加hundred，若此时十位个位均为，不加and
   if(a/100!=0)
   {                    strcat(k,num1[a/100]);
                        strcat(k,"hundred ");
                        if(b!=0)
                        strcat(k,"and ");
   }
                //当后两位在以内时，直接调用num1[n]，输出
                if(b<20)
                        strcat(k,num1[b]);
                //当后两位大于时
                else
                {       //先调用num10，输出十位数
                        strcat(k,num10[b/10]);
                        //个位不为时应输出"-"个位数
                        if(b%10!=0)
                        {       strcat(k,"\b-");
                        strcat(k,num1[b%10]);
                        }
                }
                char *p=new char[strlen(k)+1];
                strcpy(p,k);
                return p;
}
```

```
char *robot::tran_int(int n)
{               char *p;
                char kk[1000]="";
                if(n>1999999999)
                {               //cout<<"dev C++平台无法处理太大的数！"<<endl;
                                strcpy(kk,"dev C++平台无法处理太大的数！\n");
                }
                else
               {   //三位三位取出，存入abcd中
                 int a=n/1000000000,b=(n%1000000000)/1000000,c=(n%1000000)/1000,d=n%1000;
                //当abcd不等于0时，输出并加上billion ,million或thousand
                if(a!=0)
                {               p=out(a);
                                strcpy(kk,p);
                                strcat(kk,"billion ");
                                delete [ ] p; //释放在out函数中动态申请的空间
                  }
                if(b!=0)
                {               p=out(b);
                                strcat(kk,p);
                                strcat(kk,"million ");
                                delete [ ] p; //释放在out函数中动态申请的空间
                }
```

```
        if(c!=0)
        {           p=out(c);
                    strcat(kk,p);
                    strcat(kk,"thousand ");
                    delete [ ] p;          //释放在out函数中动态申请的空间
          }
          if(d!=0)
          {         //据英文语法规则，最后两位前一定有and
                    if(d<100&&(a!=0||b!=0||c!=0))
                              strcat(kk,"and ");
                    p=out(d);
                    strcat(kk,p);
                    delete [ ] p;    //释放在out函数中动态申请的空间
        }
        }  //end  of  if(n>1999999999)  else
        p=new char[strlen(kk)+1];
        strcpy(p,kk);
        return p;
}
```

类外定义设置函数

```
void robot::set(char n[],char t[],int m)          //设置修改数据
{               strcpy(name,n);
                strcpy(type,t);
                if(num==m)      //待翻译的整数没有变
                                return;
                else
                {               num=m;
                                delete [ ] ps;  //删除已有的英文句子
                }
                if(num>0)
                {               char *tp=tran_int(num);
                                ps=new char[strlen(tp)+1];
                                 strcpy(ps,tp);
                                delete [ ] tp;   //释放在trans_int中动态申请的空间
                }
                else if(num==0)
                {               ps=new char[5];
                                strcpy(ps,"zero");
                }
                else
                {               ps=new char[13];
                                strcpy(ps,"负数不能翻译");
                }
}
```

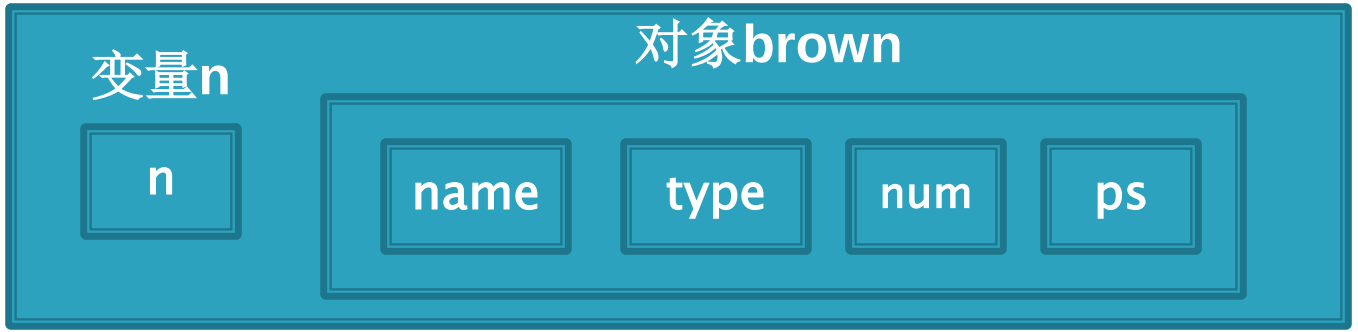
机器人翻译测试

```
int main()
{
        robot brown;
        brown.print_num();
        int n;
        cout<<"请输入n：";
        cin>>n;
        brown.set("brown","800#",n);
        brown.print_num();

    return 0;
}
```

存储空间的实际分配

- 当运行**int n;**与**robot brown;**后
- 存储空间分配如下：

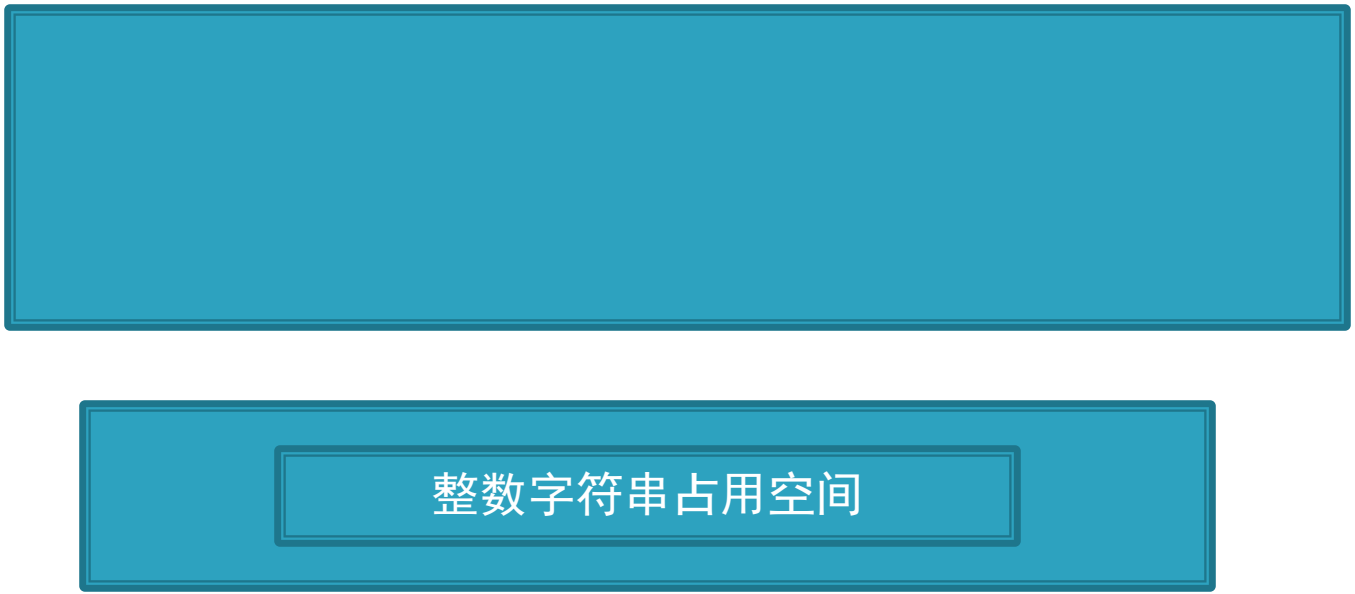


new算符操作结果

- 当运行**brown.set(“brown”, “800#”,n);**后
- 堆区有字符串空间被占用，具体空间分配如下：

空析构函数的实际作用

- 当析构函数为空函数时：**~robot() { }**
- 运行**return**语句后，空间状态如下：



析构函数的实际作用

- 当析构函数为：**~robot() { delete [ ] ps; }**
- 运行**return**语句后，空间状态如下：

感谢收看！