刘一兵

12-1

得分配长度是5的来存储'\0'

```
Cow::Cow(){
    name[0] = '\0';
    hobby = new char[4];
    strcpy(hobby, "null");
    weight = 0;
    } //构造函数
```

重载=运算符的时候如果不释放之前的hobby申请的内存会发生内存泄露

```
Cow & Cow::operator=(const Cow & c) //赋值运算符的重载
{
    if(this==&c) //if(*this==c)
        return *this;
    strcpy(name,c.name);
    hobby=new char(strlen(c.hobby)+1);
    strcpy(hobby,c.hobby);
    weight=c.weight;weight;
    return *this;

}
```

这个weight啥意思

```
Cow & Cow::operator=(const Cow & c) //赋值运算符的重载
{
    if(this==&c) //if(*this==c)
        return *this;
    strcpy(name,c.name);
    hobby=new char(strlen(c.hobby)+1);
    strcpy(hobby,c.hobby);
    weight=c.weight;weight;
    return *this;

}
```

12-2

有返回值但是并没有return 语句

```
std::istream & operator>>(std::istream &is,String &a)
{
    is>>a.str;
    a.len=strlen(a.str)+1;
}
```

这只分配一个空间，后面输入名字根本就存不下，直接内存溢出了

```
String::String()
{
    str=new char[1];
    len=1;
    str[0]='\0';
    num_strings++;
}
```

我理解的是根据输入的大小给a.str分配空间，并且应该先释放之前给a.str分配的内存空间避免内存泄露。然后再把输入的数据赋值过去

```
std::istream & operator>>(std::istream &is,String &a)
{
    is>>a.str;
    a.len=strlen(a.str)+1;
}
```

我记得之前有一章讲过tolower和toupper吧

```
//字符串转大小写
void String::Stringlow()
{       618vvv, last week • 12部分
    for(int i=0;i<len;i++)
    {
        if(str[i]>=65&&str[i]<=90)
        str[i]+=32;
    }
}
void String::Stringup()
{
    for(int i=0;i<len;i++)
    {
        if(str[i]>=97&&str[i]<=122)
        str[i]-=32;
    }

}
```

如果a的长度比当前类的长度要小，会出现内存溢出

```
bool String::operator==(const String &a) const
    {
        for(int i=0;i<len;i++)      618vvv, last week • 12
        {
            if(str[i]!=a.str[i])
            return false;
        }
        return true;
    }
```

12-4

题目是有默认参数的，这边没了

```
Stack::Stack(int n)
{
    size=n;
    pitems=new Item[n];
    for(int i=0;i<n;i++)pitems[i]=0;
    top=0;
}
```

不是这个意思吧，应该是根据赋值的n或者赋值运算符赋值的栈或者是复制构造函数传入的栈的大小，决定栈的大小，而不是直接是10

```
bool Stack::isfull() const
{
    return top==10;
}
```

这个size应该是栈的大小吧，不能++把

```
bool Stack::push(const Item &item)
{
    if(isfull())return false;
    if(top==size) size++;
    pitems[top]=item;
    top++;
    return true;      618vvv, 5 days ago •
}
```

pop函数没有返回值

```
bool Stack::pop()
{
    if(isempty())return false;
    top--;
}
```

12-5

这个缩进...我觉得要不就if else都用{}括起来，这个缩进很容易让人有歧义

```
bool Queue::enqueue(const Item &item) // add item to
{
    if(isfull())return false;
    Node *add=new Node;
    add->item=item;
    add->next=NULL;
    items++;
    if(front==NULL)
        front=add;
    else
        rear->next=add;
    rear=add;      618vvv, 3 days ago • 12章作业
    return true;

}
```

这边没return，应该是return false

```
bool Queue::dequeue(Item &item)
{
    if(isempty()==0)false;
    item=front->item;
    items--;
    Node *temp=front;
    front=front->next;
    delete temp;
    if(items==0)rear=NULL;
    return true;
}
```

`main.cpp`我看和程序清单12.12一样的啊。
我感觉应该是不用自己输入把，应该是固定队列大小和模拟时长，然后写for循环，人数从1开始，直到平均等待时间超过1min，客户数是当前模拟的人数-1

12-6

这个line1.isfull()判断有用吗，两个队列长度是一样的，然后如果line2>line1那么说明line1肯定没满，就算line1满了，那也不能往满了的里面加

```
else if(line2>line1||line1.isfull())
{
    customers++;
    temp.set(cycle);     // cycle = time of arrival
    line1.enqueue(temp); // add newcomer to line
}else{
    customers++;
    temp.set(cycle);      618vvv, 3 days ago • 12章作
    line2.enqueue(temp);
}
```

两个队列一个顾客随机出现的概率也不会变，为什么要<2

```
bool newcustomer(double x)
{
    return (std::rand() * x / RAND_MAX < 2);      618vvv, 3 d
}
```