约瑟夫问题

问题：约瑟夫问题是个有名的问题：N 个人围成一圈，从第一个开始报数，第 M 个将被杀掉，最后剩下一个，其余人都将被杀掉。例如 N=6，M=5，被杀掉的顺序是：5，4，6，2，3，1。现在输入 N，M，选择合适的存储结构，建立线性表，按被杀掉顺序输出被杀的人的编号。

【问题分析】

数据对象：两个从键盘输入的数

需要功能：

1、接受键盘输入两个的数;
2、创建一个与输入数对应大小的集合，存储在内存中
3、模拟约瑟夫的淘汰过程

结果显示：通过屏幕输出每次淘汰的编号

举例

6　5→N　M
(1，2，3，4，5，6) →List
List=(1，2，3，4，6)　out<<5
List= (1，2，3，6)　out<<4
List= (1，2，3)　out<<6
List= (1，3)　out<<2
List= (1)　out<<3
out<<1

【算法设计】

1、输入两个整数，并建立对应大小的约瑟夫环;
2、设计一个计数器，记录当前所报的数;
3、不断遍历约瑟夫环，删除报到对应数的元素
4、输出删除的元素的编号;

【ADT 定义】

数据对象：一组连续整数

数据关系：人与人之间的编号是连续的，因此满足线性关系。

基本操作：

准备能够存储一组整数的空间
删除元素
遍历空间，获取整数值

【线性表 ADT 的设计】

ADT IntegerSet {
数据对象：D={ ai |ai ∈整数，i = 1 ， 2，……，n，n>=0}
数据关系：R={<a(i-1),ai> | a(i-1) , ai ∈D}
基本操作
Link* head():
void init ()
bool delete(Elem&)
bool append(Elem&)
void next()
void getvalue(Elem)
void getlen()
}

【详细设计】

该程序有 2 个模块组成：

- **输入模块：**设计一个输入模块 void input(List<int> L)，输入两个整数 M，N，创建一个线性表，大小为 N，线性表中的元素从 1 增长到 M；
- **计算与输出模块：**设计一个模拟约瑟夫过程的模块，void josephus (List<int> &L),并输出删除的元素值

【物理数据类型的实现】

抽象数据类型：线性表

物理数据类型：集合元素是整数，物理数据类型选用整形变量 int，约瑟夫问题会删除其中元素，因此选定基于链表实现的线性表

输入模块的算法步骤：

1、从键盘读取整数；
2、新建链表；
Linklist<int> L

3、初始化链表插入元素；
append();

```
void input(List<int> L){
    int m,n;//输入整数
    cin>>m>>n;//创建链表
    Linklist<int> L;
    For(int i=1;i<=m;i++)
        L.append(i);//给链表元素赋值
    return;
}
```

计算模块的算法步骤：

1、遍历链表元素；
next()，getval（）,getlen(),head()
2、当计数器数值为 n 的倍数时，输出该元素编号，删除该节点；
delete（）

```
void josephus (List<int> &L){
    int i=0;                    //   初始化计数器
    Link* p=L.head();           //指到头节点
    while(L.getlen()>0)
    {
        i++;
        if(i%n==0)              //判断是否是对应人
        {
            cout<<L.getval(p);//输入编号
            L.delete(p);        //删除节点
        }
        p=L.next();             //继续遍历
        if(p==L.tail())         //形成环
            p=L.head();
    }
    return;
}
```

【算法分析】

程序由 2 个模块：
输入模块：void input(List<int> L)，时间复杂度为 O(n);
计算模块：void josephus (List<int> &L)，时间复杂度为 O(n)

【测试样例】

1、边界值测式：输入 M=1,N=1；
2、压力测试：M=10000,N=6；
3、一般情况测试：M=6,N=5；

【如何设计测试样例】①

1、显式功能性需求

对于显式功能性需求我们最长用到的方案主要有三种:
等价类划分法
边界值分析法
错误推断法

1.1、 等价类划分法

我们如果想测试一个功能的最傻的办法就是穷举。比如说一个密码验证功能，我们把所有的可能的密码都尝试一遍，自然就可以覆盖掉到所有的问题与可能。但是这种穷举的方法明显是做不到的。因此我们要用到等价类划分法。

等价类划分法就是说我们将所有可能的输入数据或操作分为多组不同的子集,每个子集中的数据与操作对发现程序中的潜在错误都有同等的效应。这样我们就将一个子集称为一个等价类。比如输入各种与用户名不相符的密码，是一个等价类；输入各种不存在的用户名是另一组等价类。这样在测试的时候，我们只要在每个等价类中选择一个典型操作，就可以达到较好的测试覆盖度。

等价类还会分为有效等价类和无效等价类两种。有效等价类指的是合理的、有意义的输入，主要用来验证功能是否实现了某个功能。无效等价类与有效等价类相反，指的是无意义的，超过软件规格的，不合理的输入，主要用来测试功能的健壮性，看是否考虑了如何处理不合理的情况。

1.2、边界值分析法

在我们在测试合理与不合理的数据的时候，往往最容易出现问题的就是合理与不合理的边界，这时我们就需要使用边界值分析法了。边界值分析法，就是对恰好大于、小于和等于边界的值进行测试，来验证程序是否做到了合适的处理。边界值分析法一般是作为等价类的补充，来加强测试功能实现的程度与健壮性保障的程度，是否符合规格。

1.3 错误推测法

在测试的时候就算我们使用了等价类划分法和边界值分析法,也很可能会遗漏一些需求中没有清晰提出，技术上比较隐蔽的错误。这种错误就需要测试人员通过已有的经验、对功能实现可能的方法的理解或直觉，来推测出软件中可能存在的各种错误与场景，然后编写测试用例来进行验证，这就叫做错误推测法。比如，登录超时后，某个需要权限操作的功能在使用的时候，是否跳到了登录页，还是直接报错，甚至说依旧可以操作。这种错误是需要测试人员一定的经验、技术积累与直觉的。

虽然说功能性测试往往是黑盒测试,但是如果测试工程师对于功能的实现有一定的理解——比如说是否用了缓存、是否使用了消息队列、是否某个地方会消耗极大的性能等等——将会更容易的推断出哪些地方会产生错误。

2、非功能性需求

在测试工程师测试完显示功能需求之后，还要考虑到系统的非功能性需求。这种需求可能在文档中有明确提到，也可能并没有明确的提出。但是我们的测试工程师在测试的时候却必须

要关注到。

2.1 兼容性测试

兼容性指的是开发的软件是否在各种平台都可以使用。比如我们开发一个网站，我们的用户可能会用到各种不同的浏览器访问我们的网站。这样我们在测试的时候，就不能只考虑到某一种浏览器。我们需要考虑到多种浏览器的兼容性。

兼容性测试可能会涉及到：

不同厂商的浏览器及相同厂商不同版本的浏览器。

不同的设备终端及操作系统

不同的屏幕分辨率

不同的用户软件环境（比如是否禁用了 cookie、是否可以连接外网等）

2.2 安全性测试

我们的测试人员还需要关注到开发软件的安全性。这涉及到：

用户隐私信息是否加密

需要权限的资源是否有没有权限也可以被拿到的风险

会不会受到跨站脚本的攻击

会不会受到 sql 注入攻击

2.3 压力测试

测试人员也需要考虑的软件是否能够承载其需求所需的压力，例如：

软件是否能在合理的时间内响应用户行为

软件是否可能承载足够的请求

软件在处理大数据量时会不会产生资源锁死

【如何设计并撰写算法思想】

1、模块化思考：将整个问题分解为几个模块，再设计每个模块的算法；

2、基本操作化思考：在每个模块内部，将需求通过调用基本操作实现；

【如何设计和表述数据结构】

1、写出当前的数据类型，与数据间的逻辑关系

2、根据数据间逻辑关系选择合适的 ADT；

3、根据需要的基本操作，定义合适的基本操作

【如何设计并表示算法步骤】

1、将每个模块再次分解为一些简单逻辑块

2、写出每个逻辑块内需要的基本操作、

【如何表示伪代码】②

1、安排任务序列并编写相应地伪代码。

2、从伪代码的声明开始，确定该伪代码主要目标。

3、通常用连续的数字或字母来标示同一模块中的连续语句，可省略标号。

4、在程序中缩进方式，这样有助于理解决策控制和执行机制，可以很大程度上提高了可读

性。
5、详细说明实际代码中将要发生的一切，不要将伪代码抽象化。
6、变量不需声明，但变量局部于特定过程，不能不加显示的说明就使用全局变量；
7、检查伪代码的所有部分是否完整，有限且清晰，以便理解。
8、不要以完整的编程方式编写伪代码；必须易于理解，因此不需要包含太多技术术语。

【如何写实验日志】

1、记录实验日期，地点
2、记录实验内容；
3、记录实验的过程
4、记录实验过程中遇见的问题；
5、记录实验中遇见问题的解决办法；

【参考文献】

[1] 小新文 929，https://wenku.baidu.com/view/fd0a350ca76e58fafbb00301.html，初学者功能性测试用例编写方法 [DB/OL].
[3] 青灯夜游，https://m.php.cn/csharp-article-415083.html，伪代码是什么？如何写一个伪代码？[DB/OL].
[3] meibaorui，http://www.imooc.com/article/47811，如何设计功能测试用例[DB/OL].