教学内容

- 一、算法分析基础
- 二、算法基本工具和优化技巧
  - 1、循环与递归
  - 2、利用数组提高算法质量
  - 3、算法优化基本技巧
- 三、基本的算法策略
  - 1、迭代算法
  - 2、蛮力算法
  - 3、分而治之算法
  - 4、贪婪算法
  - 5、动态规划
- 四、图的搜索算法
- 五、算法设计综合实践

课程要求及考核方式

- 分组要求：

可以选择独立完成，也可以组队（不超过3人）完成。

- 成绩组成：

（1）平时成绩（40%）

（2）完成综合设计任务（30%）

（3）综合设计任务答辩（15%）

（4）实践报告（15%）

一 、算法分析基础

- 算法是计算机学科中最具有方法论性质的核心概念，也被誉为计算机学科的灵魂。

- 算法设计作为用计算机解决问题的一个步骤，其任务是对各类具体问题设计良好的算法；作为一门课程，是研究设计算法的规律和方法。

1.1 用计算机求解问题

- 学习算法设计的重点就是把人类找到的求解问题的方法、步骤，以过程化、形式化、机械化的形式表示出来，以便让计算机执行。
- 用计算机求解问题的步骤：
  - 1. 问题分析：准确、完整地理解和描述问题是解决问题的第一步。
  - 2、数学模型建立
  - 3、算法设计与选择：算法设计是解决问题的核心，算法设计要同时结合数据结构的设计。

1.1 用计算机求解问题

- 4、算法表示：算法流程图、盒图、PAD图和伪码（类似于程序设计语言）等。
- 5、算法实现
- 6、程序调试：白盒测试对算法的各个分支进行测试；黑盒测试检验对给定的输入是否有指定输出。

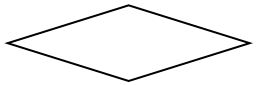
1.2 算法表示方法

1. 流程图

- 流程图可以表示任何算法的逻辑结构。
- 基本组件：

算法的
入口和出口

处理



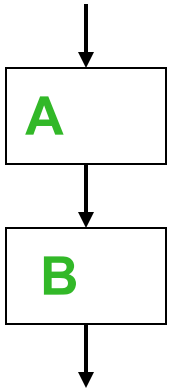
条件

控制流

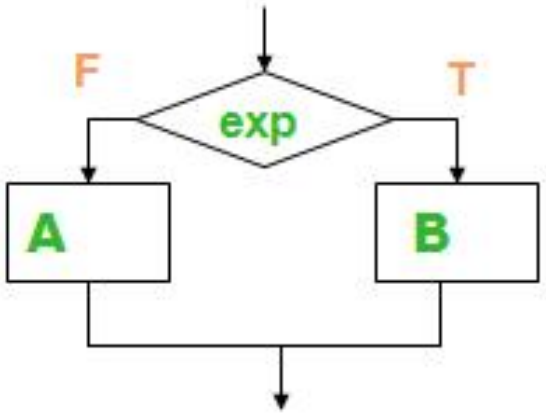
1.2 算法表示方法

- 基本控制结构：

1、顺序型



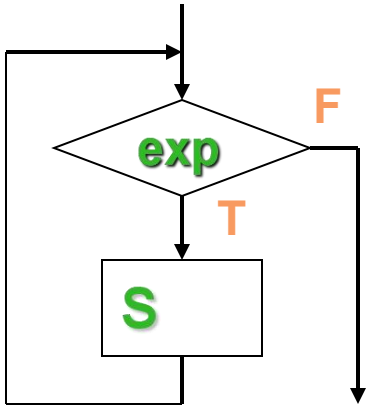
2、选择型



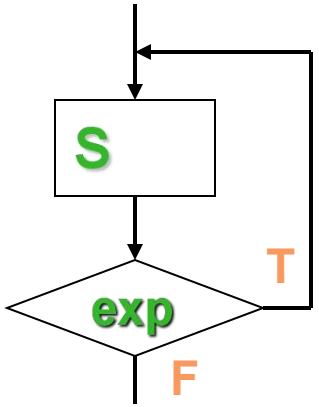
1.2 算法表示方法

- 基本控制结构：

3、当型循环型



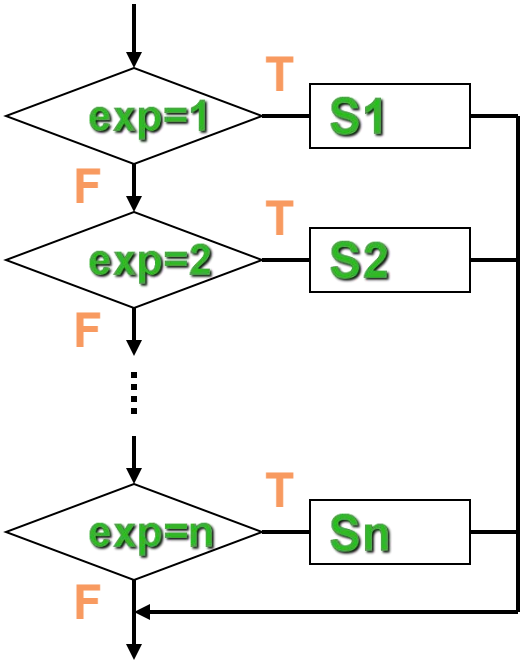
4、直到型循环型



1.2 算法表示方法

- 基本控制结构：

5、多情况选择型



Microsoft Visio(产品激活失败)

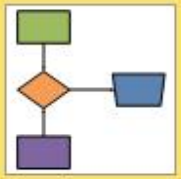
文件 开始 插入 设计 数据 进程 审阅 视图 Acrobat

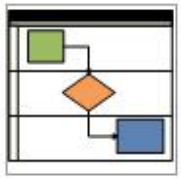
保存
另存为
另存为 Adobe PDF
打开
关闭
信息
最近所用文件
新建
打印
保存并发送
帮助
选项
退出

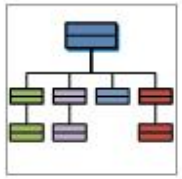
选择模板

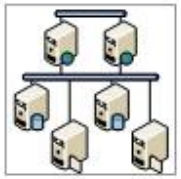
主页

最近使用的模板


基本流程图


跨职能流程图


组织结构图


详细网络图

模板类别


常规


地图和平面布置图


工程


流程图


日程安排


软件和数据库


商务


网络

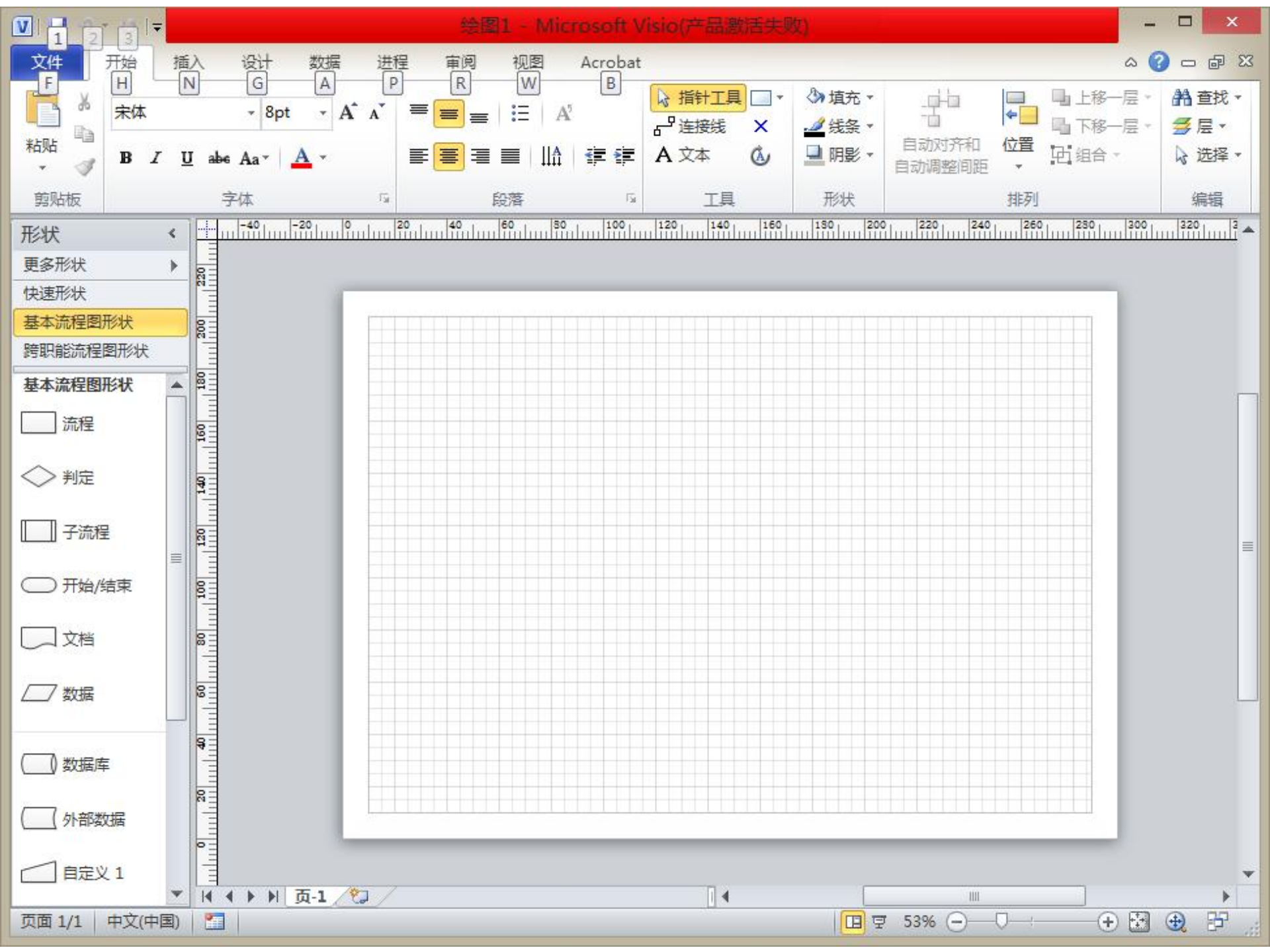
开始使用的其他方式

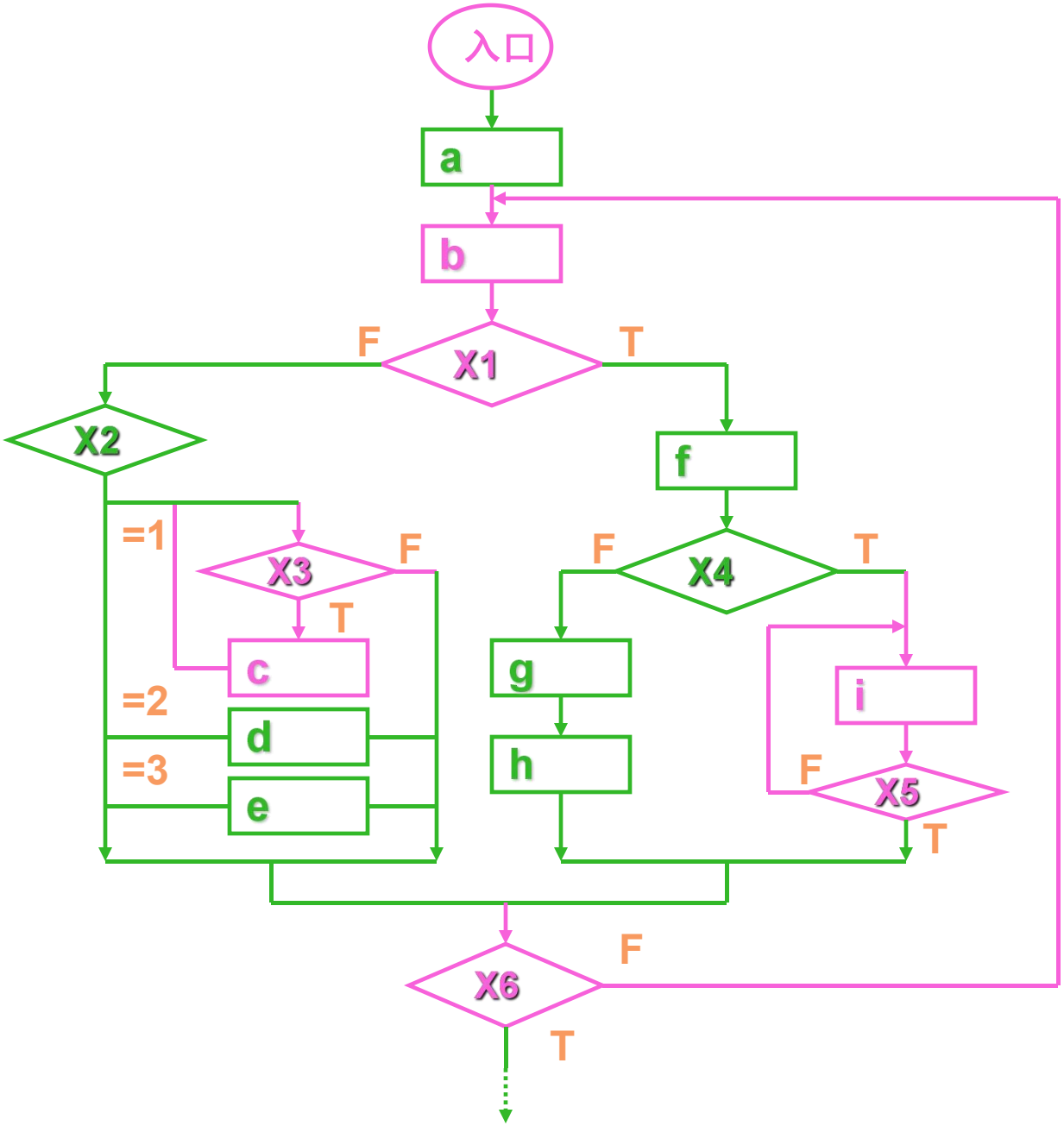

空白绘图


Office.com 模板


示例图表


根据现有内容新建


绘图1 - Microsoft Visio(产品激活失败)
文件
开始
插入
设计
数据
进程
审阅
视图
Acrobat
粘贴
宋体
8pt
指针工具
连接线
文本
填充
线条
阴影
自动对齐和
自动调整间距
位置
上移一层
下移一层
组合
查找
层
选择
剪贴板
字体
段落
工具
形状
排列
编辑
形状
更多形状
快速形状
基本流程图形状
跨职能流程图形状
基本流程图形状
流程
判定
子流程
开始/结束
文档
数据
数据库
外部数据
自定义 1
页-1
页面 1/1
中文(中国)
53%



具有嵌套形式的程序流程图

1.2 算法表示方法

1. 流程图

- 算法流程图的主要缺点：
  - 不是逐步求精的好工具，它诱使算法设计人员过早地考虑算法的控制流程，而不去考虑算法的全局结构。
  - 随意性太强，结构化不明显。
  - 不易表示数据结构。
  - 流程图的层次感不明显。

1.2 算法表示方法

例：求两个正整数的最大公约数。

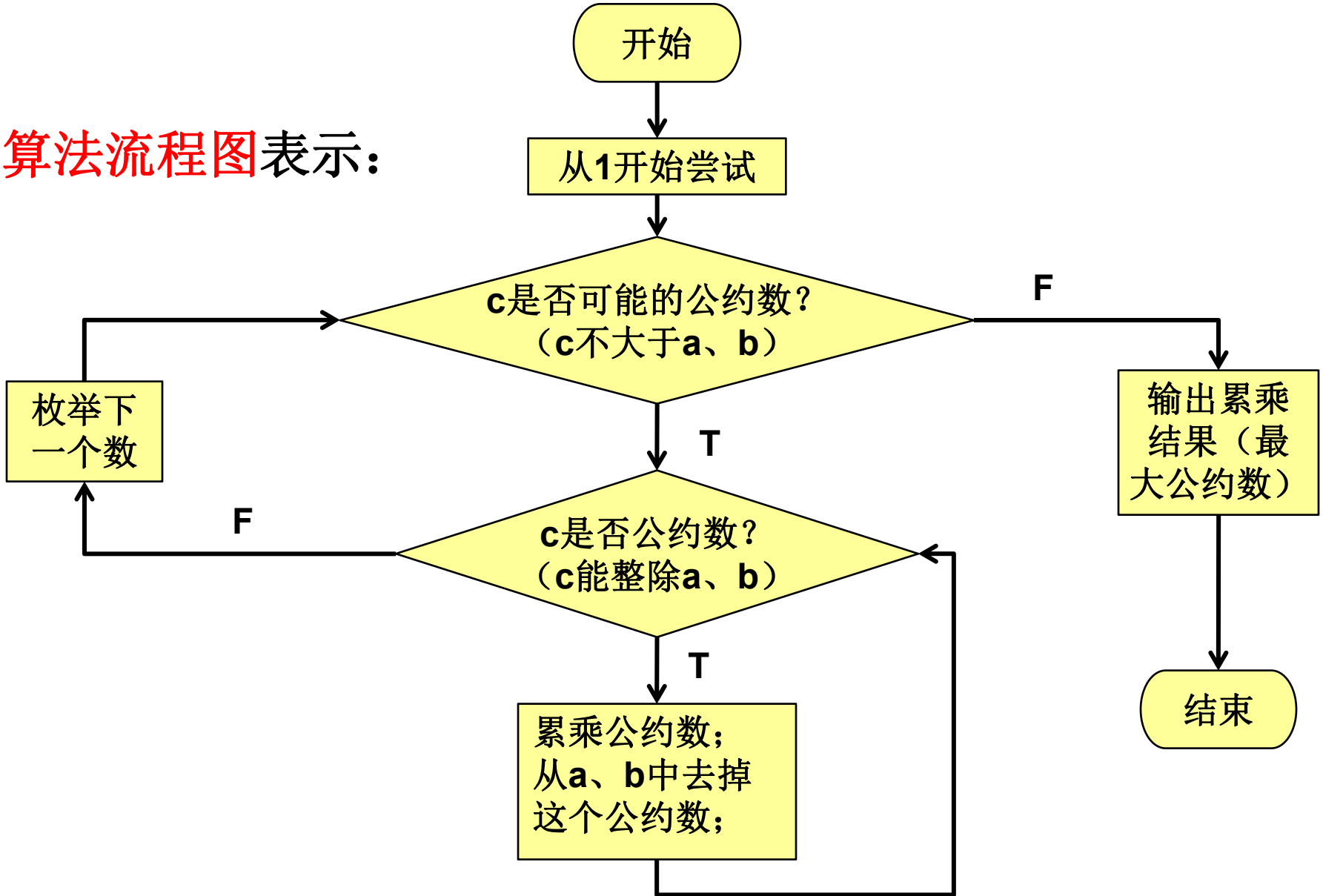
问题分析：此题只需有小学知识，就可以建立数学模型。

数学模型：a，b>0 的整数，求c，c能整除a，b，且a/c与b/c互质。

算法设计：计算机通过“枚举尝试”（逐个尝试）就可以“试出”a,b有哪些是公约数，并将这些公约数“累乘”，就能得到最大公约数。

1.2 算法表示方法

算法流程图表示：



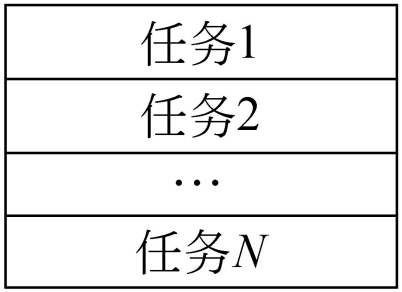
1.2 算法表示方法

算法实现：

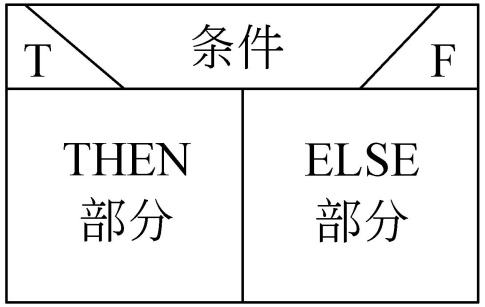
```
main( )
{
   int a,b,t,c;
   scanf("%d%d",&a,&b);
   t=1;              //变量t是为累乘公因数而设置的
   for (c=2; c<=a && c<=b; c++)  //“枚举”可能的公约数
        while(a%c==0 && b%c==0)    //“试”c是否为公约数
        {
                t=t*c;    //累乘公约数
                a=a/c;
                b=b/c;    //去掉两数中的这个公约数
        }
   printf("最大公约数：%d\n",t);
}
```

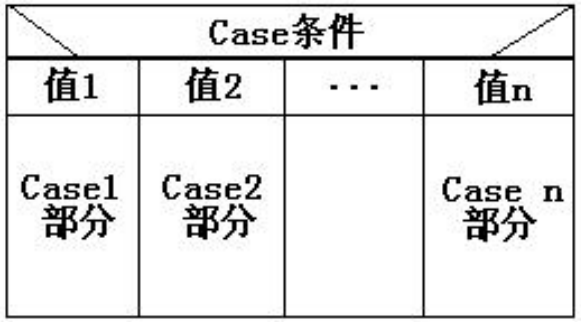
1.2 算法表示方法

2. 盒图（N-S图）

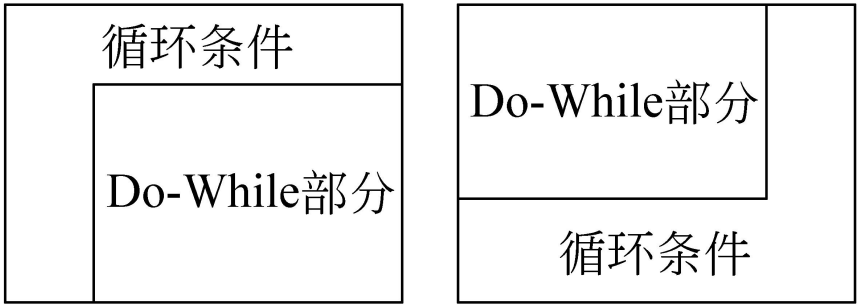


顺序结构





分支结构



循环结构

1.2 算法表示方法

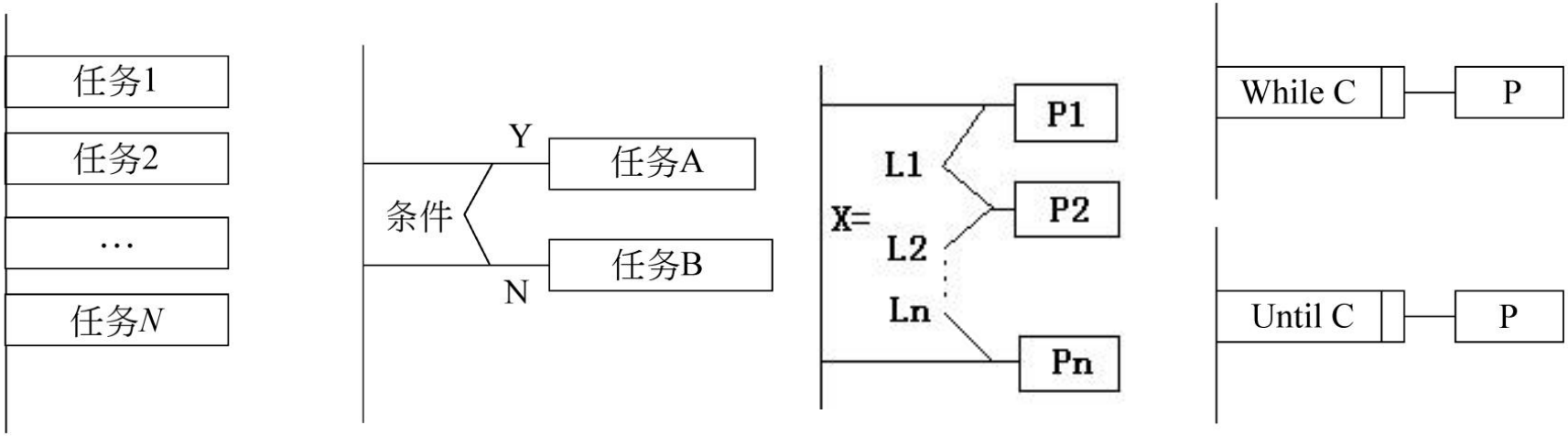
2. 盒图（N-S图）

- 避免随意转移控制；
- 很容易确定局部和全局数据的作用域；
- 很容易表现嵌套关系；
- 不易扩充和修改，不易描述大型复杂算法。

1.2 算法表示方法

3. PAD图

- 图中最左面的竖线是程序的主干线，即第一层结构。每增加一个层次，图形向右扩展一条竖线。



顺序结构　　分支结构　　两种循环结构

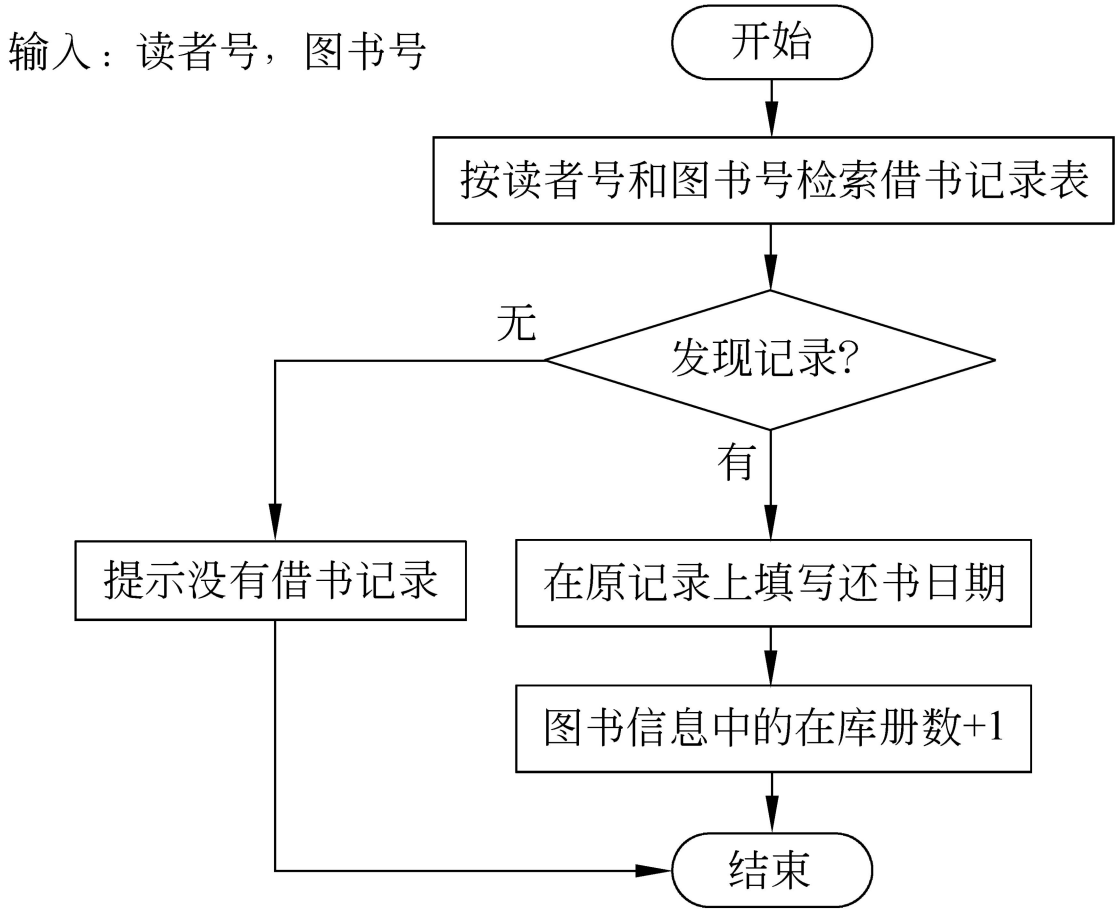
1.2 算法表示方法

3.PAD图

- 克服了算法流程图不能清晰表现程序结构的缺点；
- 不像N-S图那样受到把全部程序约束在一个方框内的限制；
- 既可用于表示算法逻辑，也可用于描绘数据结构；
- 由于是图形符号书写、录入不方便。

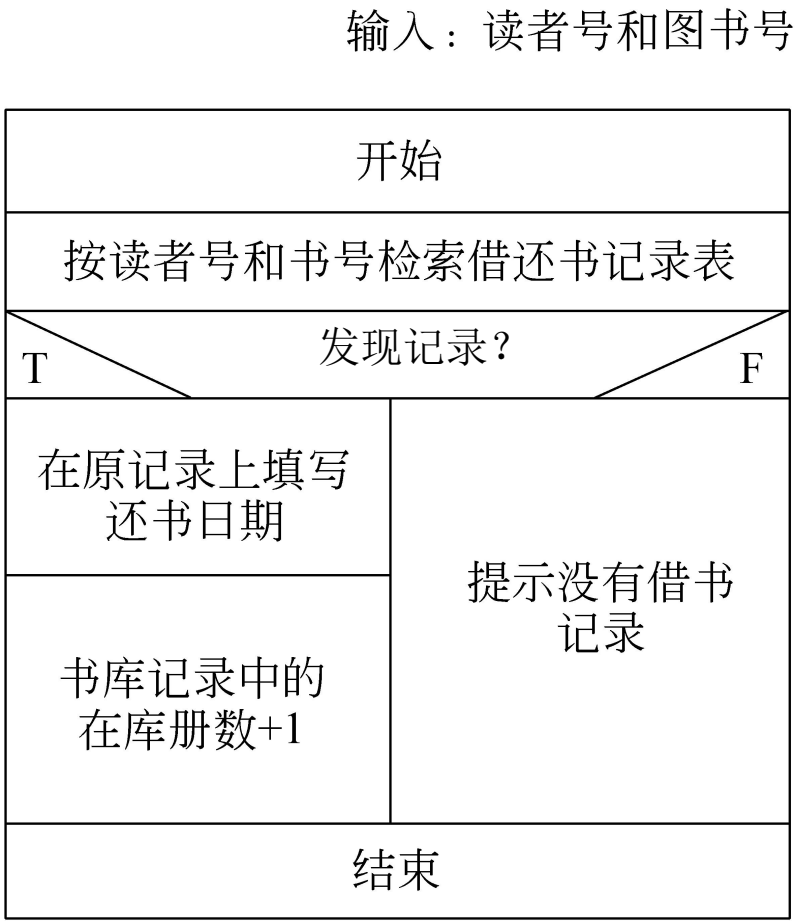
1.2 算法表示方法

- 例：图书馆还书算法流程图



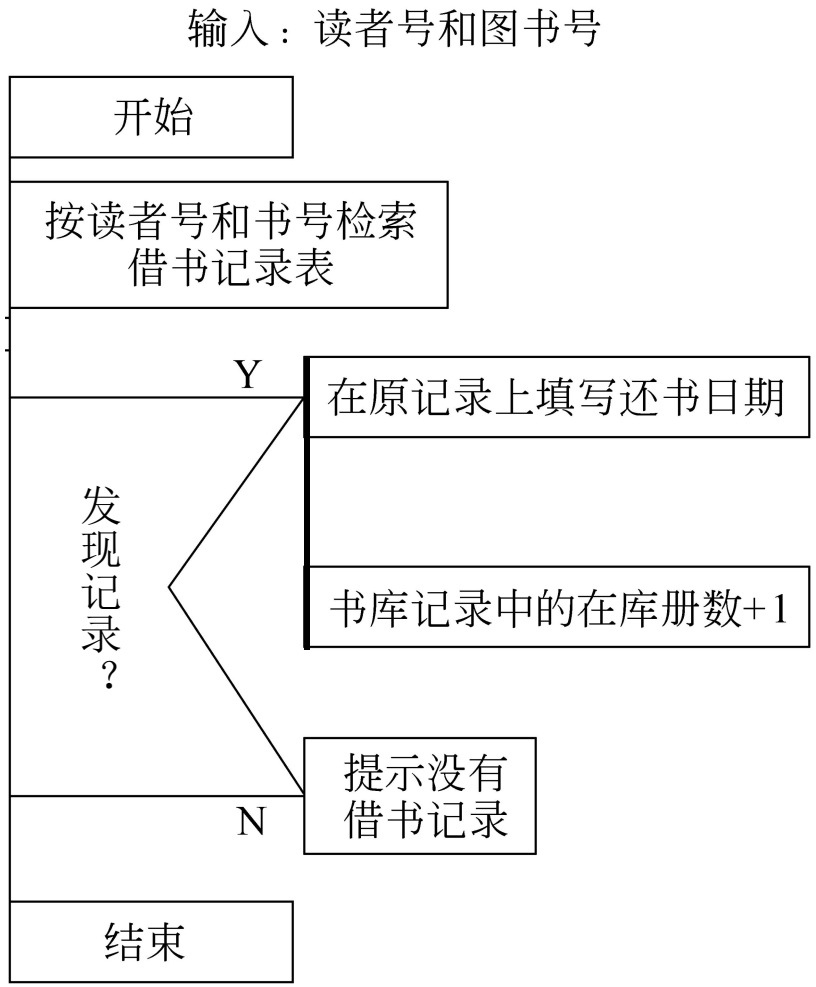
1.2 算法表示方法

- 例：图书馆还书算法盒图



1.2 算法表示方法

- 例：图书馆还书算法PAD图



1.2 算法表示方法

- 练习：用辗转相除法（欧几里德算法）求解最大公约数。（画出算法流程图）
- 定理：两个整数的最大公约数等于其中较小的那个数和两数相除余数的最大公约数。

  例如，求gcd（377，319）：

  ∵ 377÷319=1（余58）

  ∴gcd（377，319）=gcd（319，58）；

  ∵ 319÷58=5（余29），

  ∴ gcd（319，58）=gcd（58，29）；

  ∵ 58÷29=2（余0），

  ∴ gcd（58，29）= 29；

  ∴ gcd（377，319）=29.

1.2 算法表示方法

```
int main()
{
   int a,b,t;
   scanf("%d%d",&a,&b);
   while(b > 0)
   {
        t = a % b;
        a = b;
        b = t;
   }
   printf("%d\n",a);
   return 0;
}
```