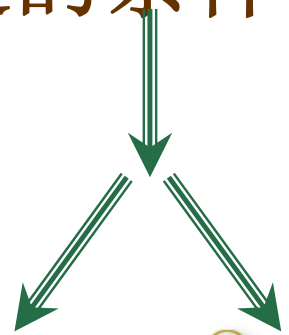
双分支结构

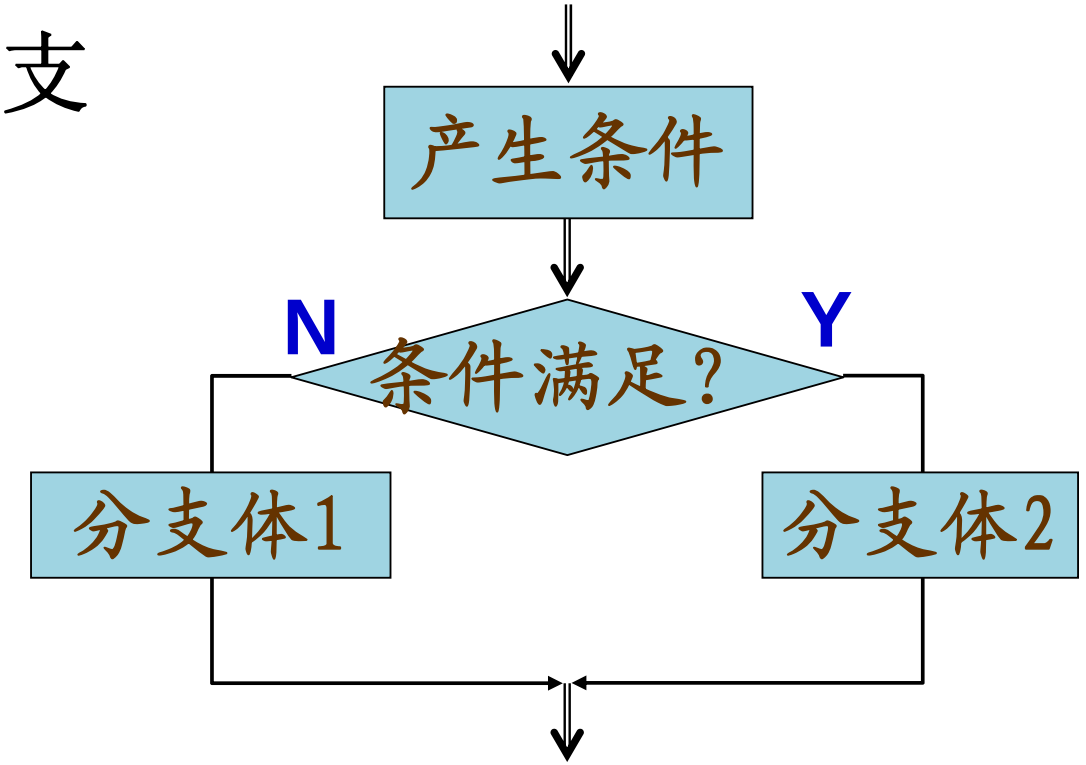
分支程序结构

- 有条件产生和条件判断两个部分组成
  - 首先，利用比较CMP、测试TEST、或者加减运算、逻辑运算等影响状态标志的指令形成条件
  - 然后，利用条件转移指令判断由标志表达的条件并根据标志状态控制程序转移到不同的程序段



双分支程序结构

- 双分支程序结构有两个分支
  - 条件为真，转移
    执行分支体2
  - 条件为假，顺序
    执行分支体1



分支体1最后一定有JMP指令跳过分支体2

条件转移指令（Jcc）

- 根据指定的条件确定程序是否发生转移

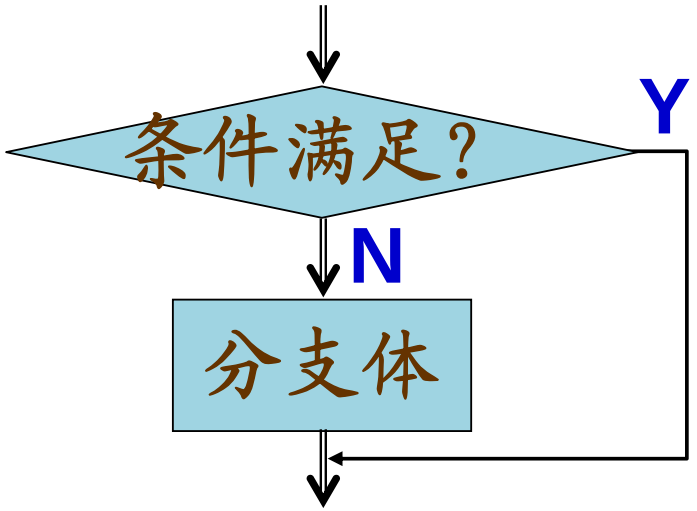
  Jcc label

  ;条件满足，发生转移；

  ;否则，顺序执行下条指令

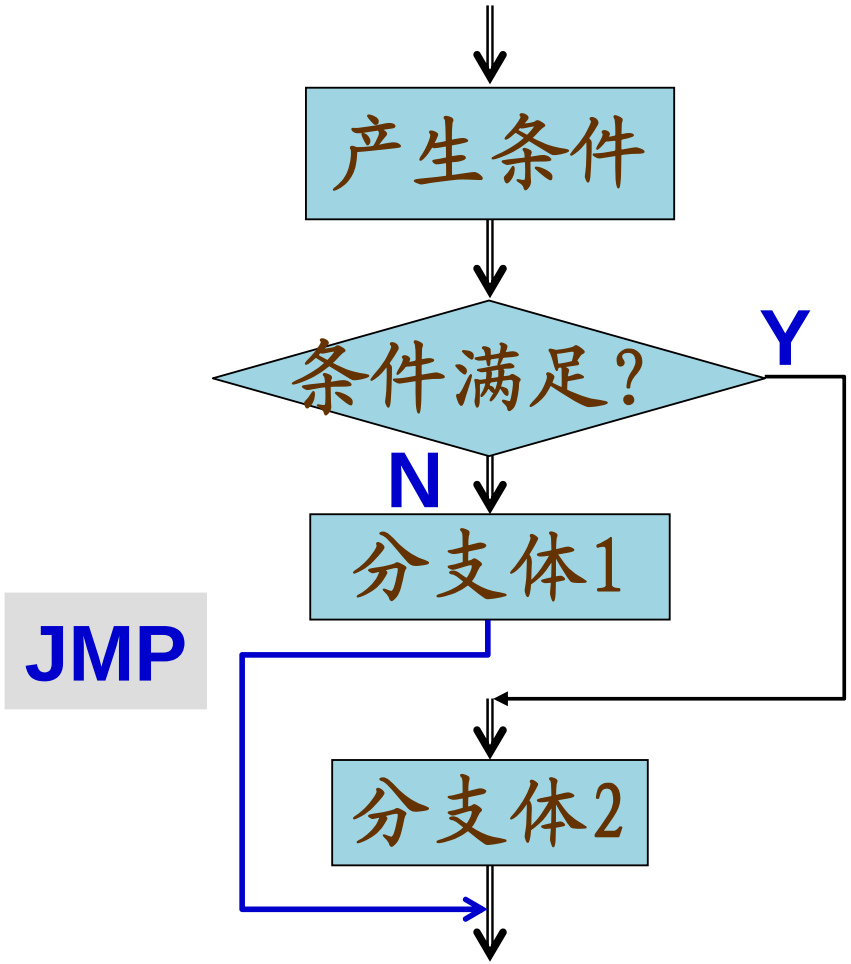
- label表示目标地址

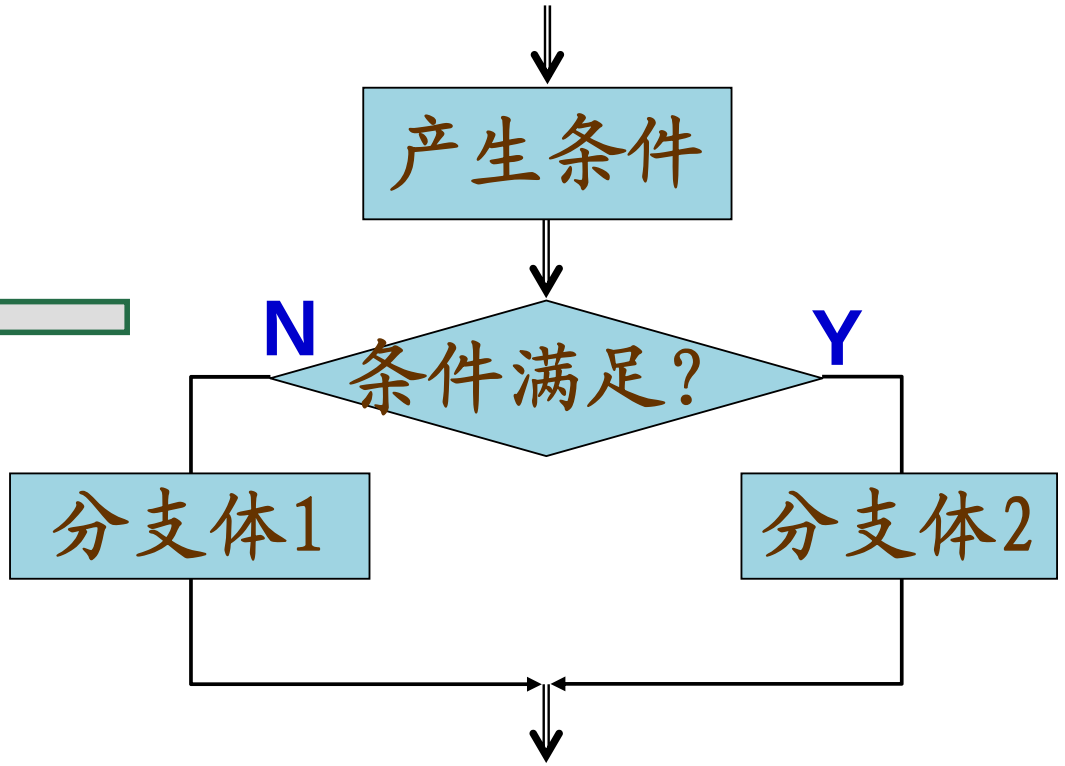
  采用段内相对寻址



Jcc: jump with condition

顺序执行的分支体最后一定有JMP指令





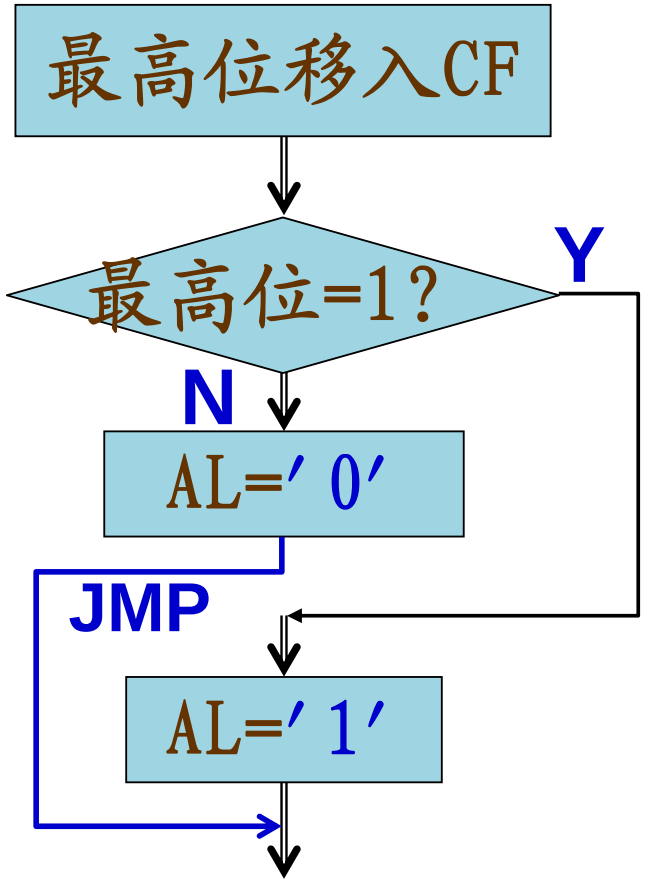
显示数据最高位程序

```
        mov ebx,dvar    ;数据来自变量DVAR
        shl ebx,1       ;EBX最高位移入CF标志
        jc one          ;CF＝1，即最高位为1，转移
        mov al,'0'      ;CF＝0，即最高位为0：AL←’0’
        jmp two         ;一定要跳过另一个分支
one:    mov al,'1'      ;AL←’1’
two:    call dispc      ;显示
```

```
dvar  dword  0bd980613h
```

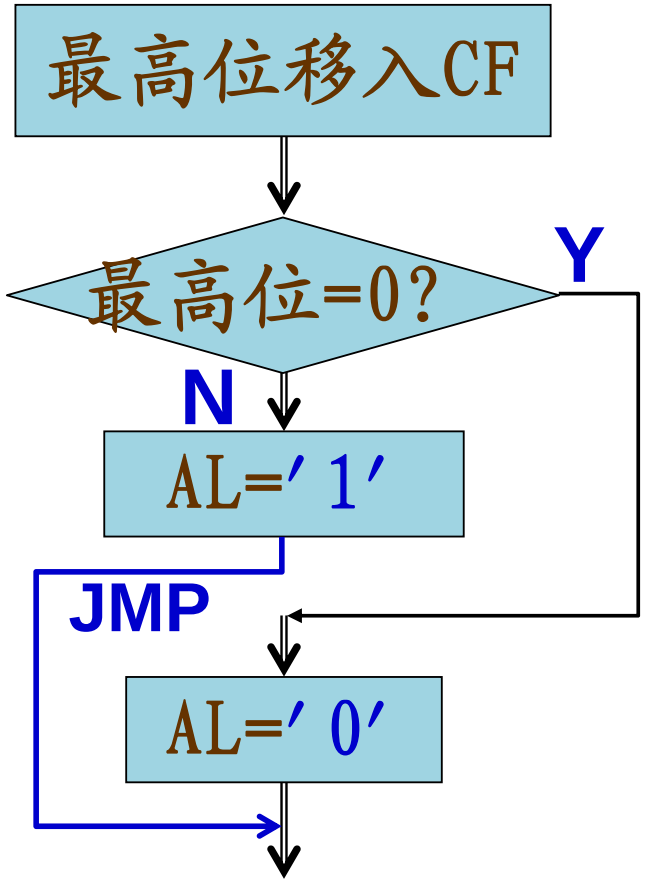
显示数据最高位程序（选用JC）

```
        mov ebx,dvar
        shl ebx,1
        jc one
        mov al,'0'
        jmp two
one:    mov al,'1'
two:    call dispc
```



显示数据最高位程序（选用JNC）

```
        mov ebx,dvar
        shl ebx,1
        jnc one
        mov al,'1'
        jmp two
one:    mov al,'0'
two:    call dispc
```



编写双分支程序结构的注意事项

- 双分支程序结构有两个分支
  - 相当于高级语言的if-then-else语句
- 条件为真执行一个分支
  - 条件为假，执行另一个分支
- 顺序执行的分支体最后有JMP指令跳过另一个分支体
  - JMP指令必不可少
  - 实现结束前一个分支回到共同的出口作用

有符号整数运算溢出程序

处理器硬件支持两种整数编码

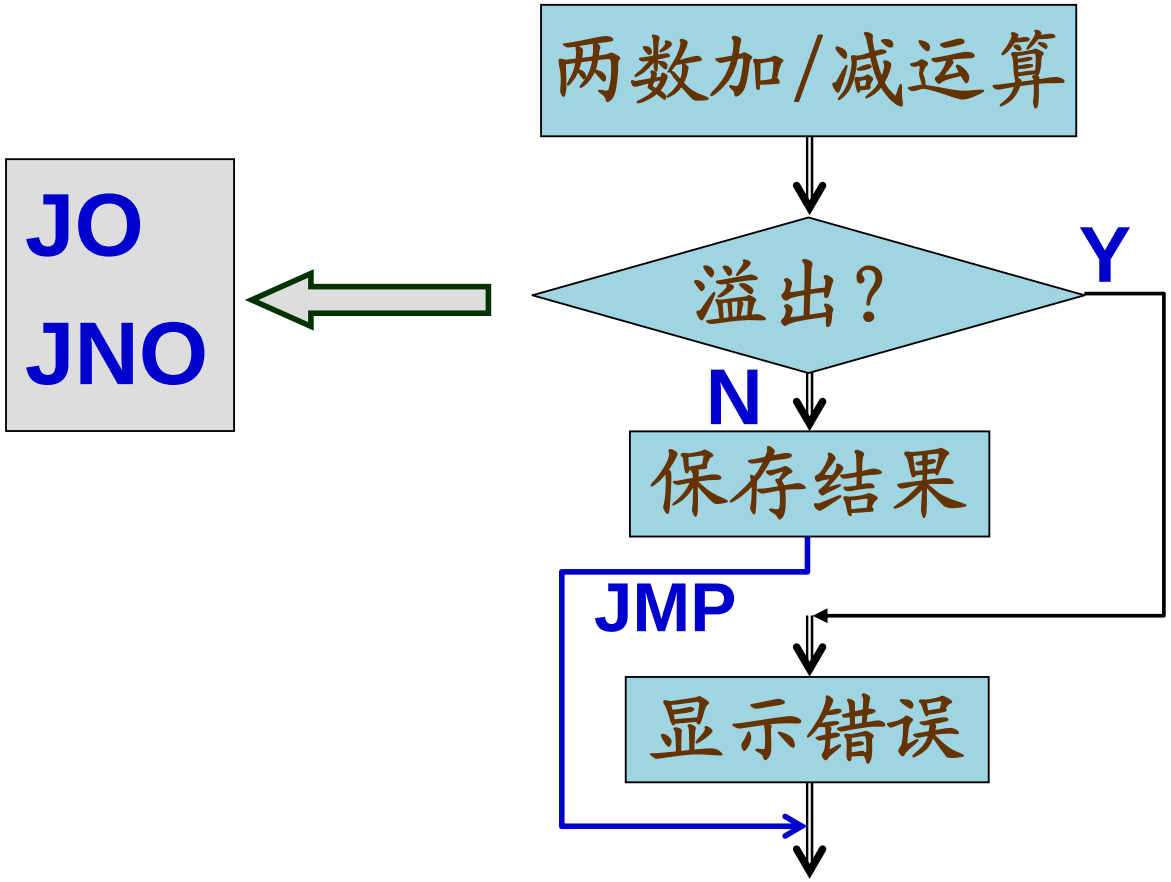
- 无符号整数，表达0和正整数
  - 采用二进制直接编码
- 有符号整数，表达负整数、0和正整数
  - 采用补码编码

配合CF标志

配合OF标志

指令	无符号整数	有符号整数
加减法	ADD/SUB	ADD/SUB
乘除法	MUL/DIV	IMUL/IDIV
条件转移	JA/JB…	JG/JL…

有符号数加减运算的溢出判断

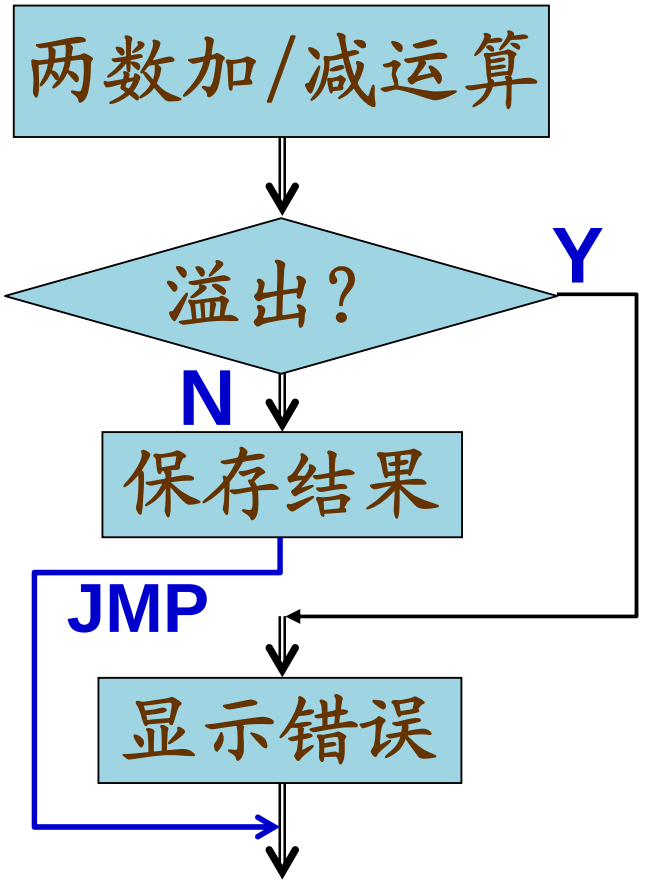


两个有符号整数变量

```
                ;数据段
dvar1    dword  1234567890    ;假设两个数据
dvar2    dword  -999999999
dvar3    dword  ?             ;保存结果的变量
okmsg    byte   'Correct!',0  ;正确时的显示信息
errmsg   byte   'ERROR! Overflow!',0
                ;错误时的显示信息
```

有符号数运算溢出程序

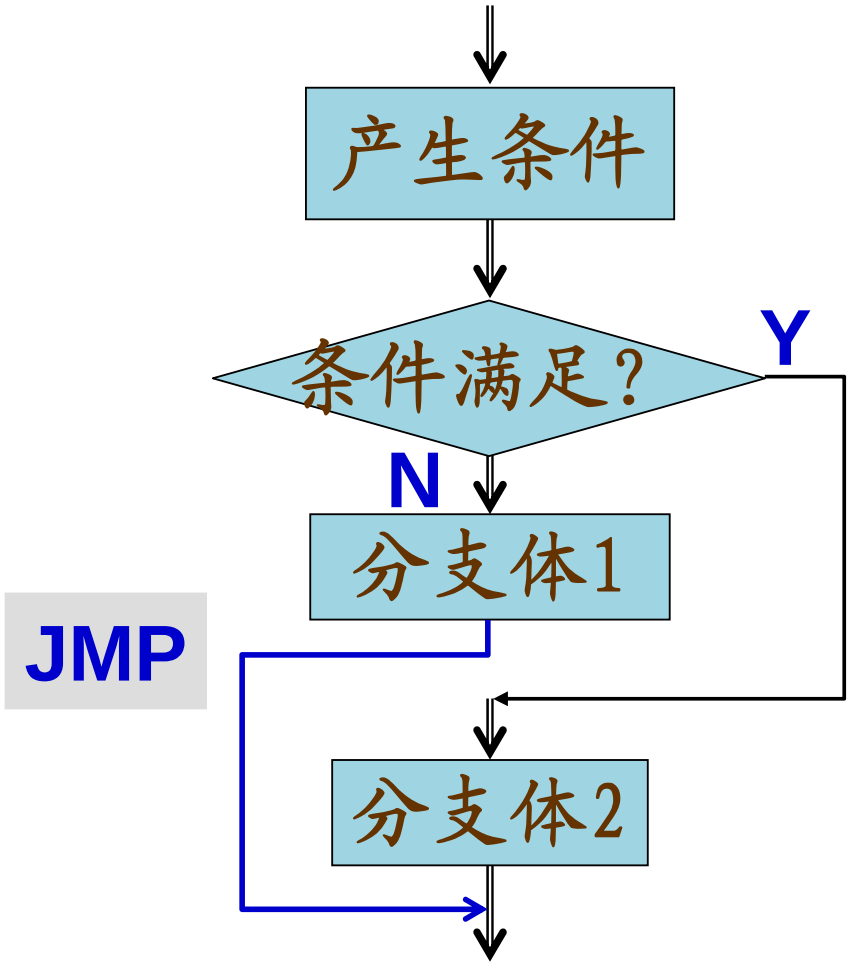
```
        mov eax,dvar1
        sub eax,dvar2
        jo error
        mov dvar3,eax
        mov eax,offset okmsg
        jmp disp
error:  mov eax,offset errmsg
disp:   call dispmsg
```

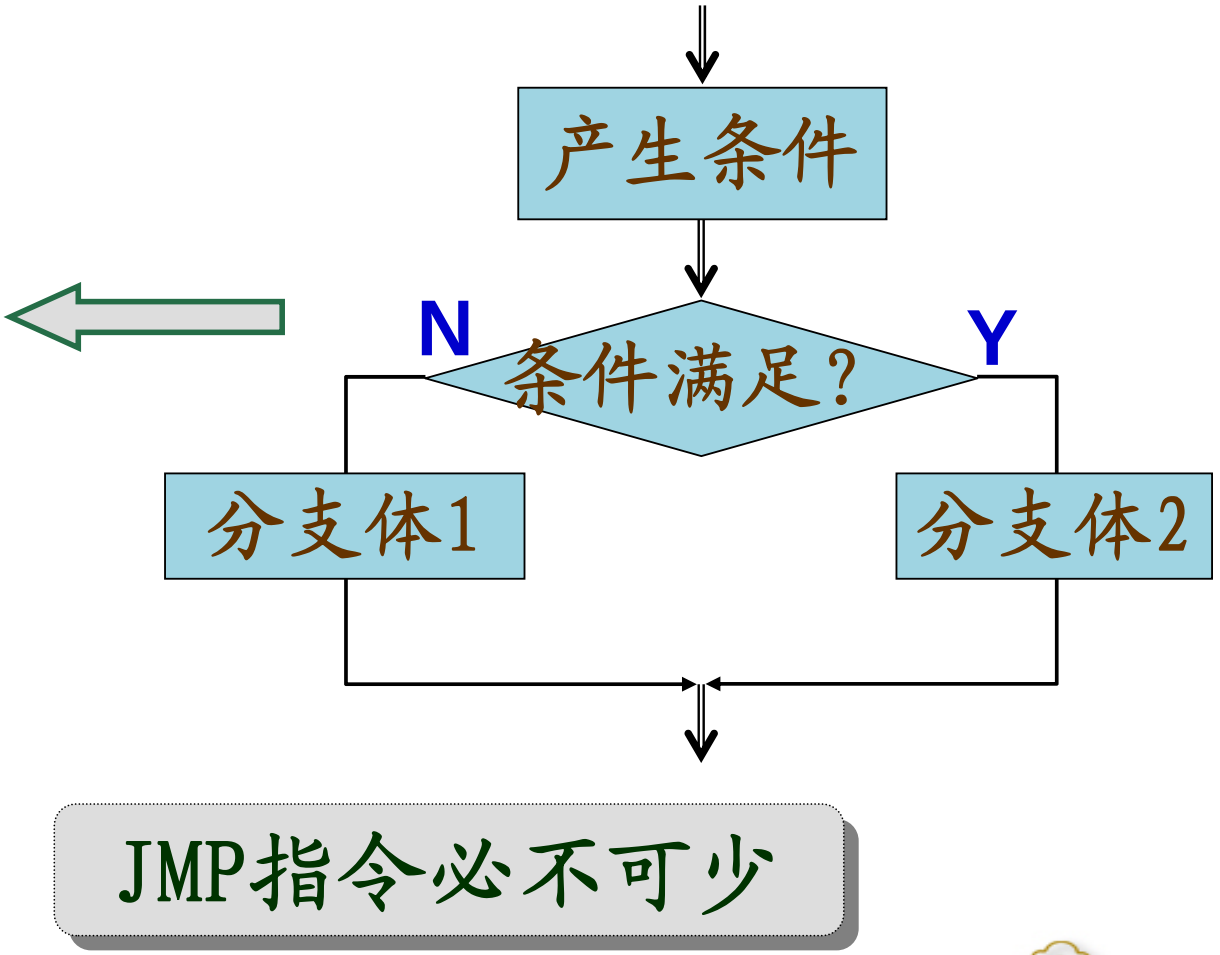


有符号数运算溢出程序

```
        mov eax,dvar1
        sub eax,dvar2                ;求差
        jo error                     ;有溢出，转移
        mov dvar3,eax                ;无溢出，保存差值
        mov eax,offset okmsg         ;显示“Correct!”
        jmp disp
error:  mov eax,offset errmsg        ;显示“ERROR!”
disp:   call dispmsg
```

双分支程序结构





JMP指令必不可少