



LINGO 程序段编程

《美国数学建模竞赛》

完整课程请长按下方二维码





目录页

目录

1 段编程的优点

2 程序结构

3 段的使用

4 注意事项





一、段编程的优点

对于数学规划问题，一般选用Lingo程序求解，其优点主要在于：

1. 模型简单整齐，便于理解
2. Lingo程序语言规范，计算速度快
3. Lingo软件比较小，安装方便快捷





二、段编程的程序结构

Lingo程序一般以“Model:”开始，“End”结束。
在程序中若没有Model和End也能执行，但是最好还是
写完整标准的程序。
主要为五段架构。



```
Model :  
Title "Example";  
.....  
.....  
END
```





程序结构

集合段

数据段

初始段

计算段

目标和约束段

五段中目标和约束段一般是不可少的，集合段用的比较多，数据段次之，初始段和计算段不一定有。这些段的顺序可调换。





程序结构

1. 集合段：以“SETS:”开始，“ENDSETS”结束，定义必要的集合变量及其元素（含定义类似于数组的下标）和属性。



SETS :

Car / 1..2 / : a, b;

Box / 1..6 / : c, d;

Link (Car, Box) : x;

ENDSETS





程序结构

2. 数据段：以“DATA:”开始，“ENDDATA”结束，对集合的属性（数组）输入必要的常数数据。常数列表中的数据之间可以逗号“,”分开，也可以用空格分开。



```
DATA :  
  
a = 1, 2, 3, 4, 5, 6;  
b = 7 8 9 10 11;  
c = 12 13 14 15;  
  
ENDDATA
```





程序结构

3. 初始段：以“INIT:”开始，“ENDINIT”结束，对集合的属性（数组）定义初值（因为求解算法是迭代算法，如能给一个比较好的初值，对提高算法的计算效果是非常有益的）。与数据段中的用法类似。



INIT :

$x, y = 1, 2, 3, 4, 5, 6,$
 $7, 8, 9, 10, 11, 12;$


ENDINIT





程序结构

4. 计算段：以“*CALC:*”开始，“*ENDCALC*”结束，对一些原始数据进行计算处理。在实际问题中，原始数据不一定能在模型中直接使用，可以用计算段对原始数据进行一定的“预处理”，得到在模型中可以使用的数据。




```
CALC:  
Total Number=  
@sum(Car(i): a(i)* b(i));  
ENDCALC
```





程序结构

5. 目标与约束段：目标函数、约束条件等没有段的开始和结束标记，因此实际上就是除了其它四个段（都有明确的段标记）外的LINGO模型。它是LINGO程序最重要的部分。



```
Min = @sum(box: c + d);  
@for(Car(i) : a(i) + b(i) < 10);  
@for(Link: @bin(x));
```





三、段的使用

1. 怎样表示: $\sum_{i=1}^{100} x_i \leq 90$?

sets: ! 集合段

s/1..100/:x; !基本集合, 集合名与属性变量;

endsets

!目标与约束段;

@sum(s(i):x(i))<90;! 循环求和函数;





段的使用

2. 怎样赋值（数据段）：

$$b_1=1, b_2=0, b_3=1,$$

$$b_4=2, b_5=3, b_6=5,$$

$$b_7=2, b_8=6, b_9=1 ?$$

sets:

ss/1..9/: b;

endsets

data: ! 数据段;

b=1 0 1 2 3 5 2 6 1 ;

enddata





段的使用

3. 怎样表示: x_{ij} 为0-1变量, $i = 1, 2, \dots, 100,$
 $j = 1, 2, \dots, 200$

sets:

a/1..100/;; b/1..200/;;

c(a,b):x;

endsets

@for(c(i,j):@bin(x(i,j)));





段的使用

4. 怎样表示约束: $\sum_{i=1}^{100} \sum_{j=1}^{200} x_{ij} = 280?$

sets:

a/1..100/;; b/1..200/;; C(a,b):x;! 派生集合;

endsets

! 目标约束段

@sum(c(i,j):x(i,j))=280;





5. 怎么表示多个约束:

$$\sum_{i=1}^{100} x_{ij} \geq 150, j = 1, 2, \dots, 200$$

sets:

a/1..100/;;

b/1..200/;;

c(a,b):x;

endsets

@for(b(j):@sum(a(i):x(i,j))>150);!集合元素的循环函数;





6. 怎样表示过滤语句: $\sum_{\substack{2 \leq k \leq 40 \\ k \neq 10}} x_{ijk} = 100$

$i = 1..20, j = 1..30$

sets:

a/1..20/;; b/1..30/;; c/1..40/;; d(a,b,c):x;

endsets

@for(a(i):@for(b(j):

@sum(c(k)|k#gt#1#and#k#ne#10:x(i,j,k))=100));

!过滤条件;





四、注意事项

问：在写程序时，有哪些容易出错的地方需要注意呀？

答：



10条哦





注意事项

1. 使用的字母没有定义
2. 定义了同名的属性
3. 分号不是英文半角输入
4. 循环语句中元素下标颠倒或者不明
5. 定义了多个长度一样的集合，而在使用中区分不明确





注意事项

6. 约束错误变成不可行或无界
7. 关系运算符（如 “=”）使用逻辑运算符（如 “#EQ#”）
8. 使用了非LINGO语言的输入；（比如%引导说明语句）
9. 函数调用错误
10. 掉了或多了括号，函数的括号写错了地方





注意事项

很多时候是程序写出来了，中间有错误，怎么进行程序的调试呢？可按下面步骤进行：





注意事项

1. 直接点击运行，如果出错会弹出错误提示，根据提示做相应的修改；
2. 可以用“！”把约束变成说明语句，而把这条语句屏蔽掉，缩小寻找出错的范围；





注意事项

3. 可以边写程序边运行，保证每行书写都是正确的程序；



关于 *Lingo* 段编程，
您了解了吗？

