

## TI-83PLUS 图形计算器程序功能(一)

在数学中，现代意义上的“算法”通常是指可以用计算技术来解决的某一类问题的程序或步骤，这些程序或步骤必须是明确和有效的，而且能够在有限步之内完成。在高二数学新教材第10章《算法初步》中，对于“算法”归纳了四方面典型特点：

- (1) 有限性——操作步骤必须是有限的；
- (2) 确定性——每一步操作都有确定的意义，不能有歧义；
- (3) 可行性——每一步操作都是程序语言所规定的可行的步骤；
- (4) 可呈现性——有输入和输出步骤；

一般地，当我们解决问题需要执行一些方法相同，过程重复的操作或运算时，我们可以编制程序，让计算机（器）来帮助我们完成这些计算。算法程序是为了要在计算机（器）中解决某个问题而设计的一系列指令，由一个或多个命令行组成的按规定的顺序执行的命令集合。执行程序时，按程序指令的顺序执行命令行的指令。

### 问题 1 根据框图编写程序并运行

根据程序框图编写程序是中学生学习算法的主要形态。在这个问题解决过程中，学生必须经历以下两个步骤：

(1) 认识框图：利用程序框图表示程序流程是一种比较直观、清楚地表现出算法执行过程的方法。框图是由一定形状框的结构，再加上联接它们的线条和箭头及标识判断成立与否的“Y”或“N”组成。矩形框为处理（执行）框，菱形框为判断框，平行四边形为输入输出框，一般还有起止框等。箭头表示程序的执行方向。

在新教材中，把常见的算法结构分为：顺序结构、条件结构和循环结构。通过对框图的阅读，我们能了解该算法的基本结构。

(2) 认识数学问题：即理解这个程序（框图）是用来解决什么问题的。

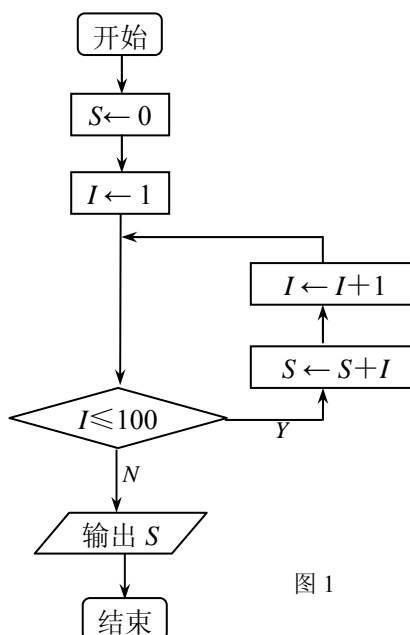


图 1

对于比较简单的问题,其算法思想往往也并不复杂,算法框图一般很容易说明问题。本例的算法就是“求从 1 到 100 的连续正整数之和”。分析其算法,它其实只是在重复加法运算,并没有用到“逆序相加”等计算方法。不过,对于我们人工完成比较“枯燥”的工作,计算机(器)却能“不厌其烦”地加以完成。因此,我们研究算法的一个重要的作用就是使用现代计算技术代替人工解决更多的问题。

由于我们采用 TI-83PLUS 图形计算器作为算法与编程学习的载体,因此我们必须先了解图形计算器的基本程序语言。

## 一、基本程序语言与操作

### (一) 创建程序



图 2

1. 按[PRGM]显示 PRGM NEW 菜单。



图 3

2. 按[ENTER]选择[1], 这时输入新程序名。

注: 此时默认输入方式为英文字母输入方式;

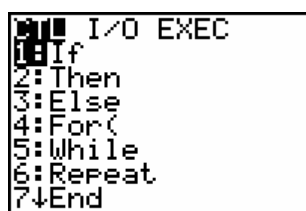
3. 按[ENTER]进入程序编辑。

[说明]:

1. 程序名的长度为 1~8 个字符, 第 1 个字符必须是字母。
2. 按[2nd][QUIT]退出程序编辑, 并返回到主屏幕中。

### (二) 基本程序语言

1. 如何调用基本程序语言?



按[PRGM], 可显示程序语言。第一列是基本程序控制语句(CTL) (图 4~ 图 6)

图 4

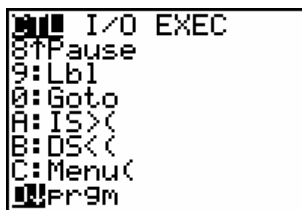


图 5

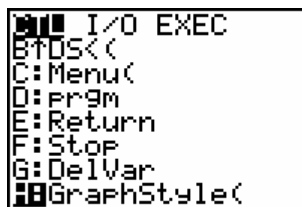


图 6

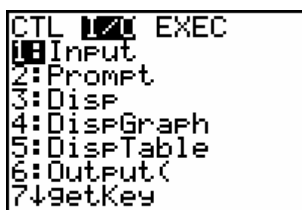


图 7

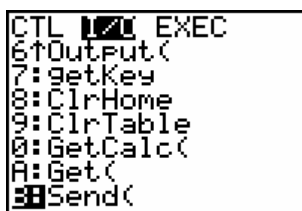
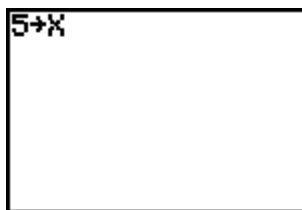


图 8

按 ，显示第二列程序命令（I/O），第二列程序语言主要是输入、输出命令；

下面，介绍若干 TI-83PLUS 图形计算器基本语言的含义及其格式。

- 赋值命令（格式：数值 变量）



例如：将 5 赋值给 X，依次按

（或 ）

图 9

- 显示命令（格式：Disp “变量，变量……”）

例如，显示 A，按 选择 3: Disp ， A

- 判断命令（格式：

:IF 条件

:Then

: 命令——执行条件为真

(: ELSE

: 命令——执行条件为假)

: END

- 输入命令

(格式 1: Input “提示内容”, 变量)

(格式 2: Prompt 变量, 变量, …)

由以上基本语言, 我们已经可以将问题 1 的程序框图用图形计算器的程序语言编写出一个简单的程序。如下图:

```
PROGRAM:EG0
:0→S
:1→I
:Lbl 1
:If I≤100
:Then
:S+I→S
:I+1→I
```

图 10

```
PROGRAM:EG0
:Goto 1
:End
:Disp S
:
```

图 11

### (三) 执行程序

若要执行一个已编写完毕的程序, 我们可以在主屏幕的一个新的命令行中, 按[PRGM], 然后在 EXEC 列中找到程序名, 按[ENTER]即可执行。

```
PRGM EDIT NEW
1:EG0
2:EG1
3:EX1
```

图 12

```
PrgmEG0      5050
              Done
```

图 13

- 循环命令 (格式:

: FOR(变量, 初值, 终值<, 增量>)

: 命令

...

: END

利用循环命令, 问题 1 的程序也可以编写成如下步骤, 运行结果是一致的。对于同一个问题, 我们可以用不同的算法实现, 所以对于一个问题的解决, 其算法一般都不是唯一的。

```

PROGRAM:EG0B
:0→S
:For(I,1,100)
:S+I→S
:End
:Disp S
:

```

图 14

```

EDIT NEW
1:EG0
2:EG0B
3:EG1
4:EX1

```

图 15

练习题:

1. 编程: 显示 2~10 之间各偶数的平方。
2. 编程计算:  $1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{200}$ 。
3. 根据程序框图, 分析该算法执行之后的结果, 并编程验证。

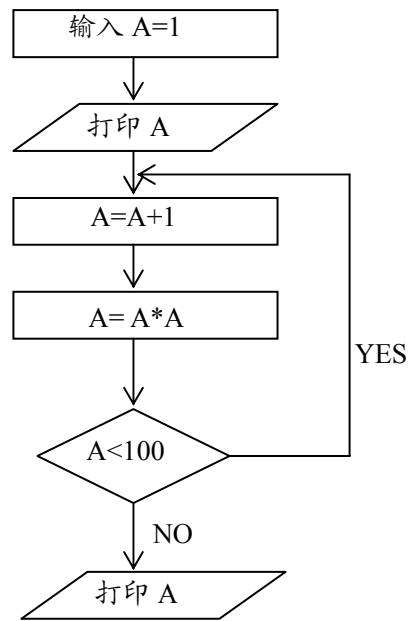


图 16

## 二、优化程序

解决数学运算问题的算法, 并不仅仅是编出一个程序。衡量算法的标准一般有: 运算次数是否较少; 运算过程是否规律; 变量运用是否尽量少; 算法过程是否能控制误差, 保证结果的精确度等等。一般而言, 我们编制的程序应尽量缩短运行时间、减少误差。

对于练习题 2 的求和问题(框图见右图 17)。我们设计程序时可以使用图形计算器编程语言中“IF...THEN...ELSE”的判断语句编程, 也可以使用“FOR...END”循环语句

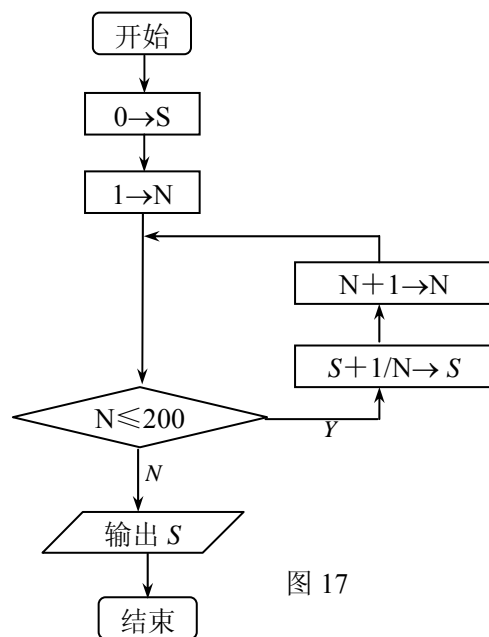


图 17

编辑。尝试运行两个程序，记录结论，记录运行所花的时间。我们会发现两者的运行效率是不同的。所以，“优化程序设计，提高运行效率”是我们设计程序时可以深入考虑的一个问题。

**问题 2：用图形计算器分别计算等式：**

$$\sqrt{10000+0.01}-\sqrt{10000}=\frac{0.01}{\sqrt{10000+0.01}+\sqrt{10000}}$$
 的两边，并观察两者的误差。

使用计算机上的计算器软件再计算一次，比较等式两边的算法哪个更精确。

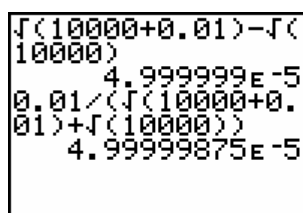


图 18

图形计算器运算结果如左图所示。我们通过执行计算机软件运算后比较可知，等式右边的算法精度更高些。这是由于两个十分近似的数相减时，若其中至少有一个无理数，那么原始数据由于使用近似值进行运算而造成的误差将对计算结果精度造成一定影响。所以，有必要时我们可以设法改变算法以提高精度。

### 三、用程序控制图形计算器的作图功能

图形计算器有很强的作函数图像以及在图形窗口内的运算功能。我们可以通过编程来实现对这些功能的调用与控制，从而在教学中有效发挥图形计算器的优势，减少临场操作所消耗的时间。

**问题 3：（1）用程序控制图形计算器作出函数  $y = x^2 + 1$  的图像；**

**（2）用变量 A 控制程序，使当  $A > 0$  时函数  $y = x^2 + 1$  的图像向右平移 A 个单位；当  $A < 0$  时函数  $y = x^2 + 1$  的图像向左平移  $|A|$  个单位。**

操作步骤：

（1）如下图编制程序 T，相关命令的调用和普通调用方法一致。

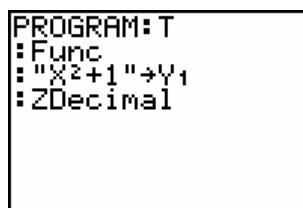


图 19

（1）按 **[MODE]**，找到 Func 命令并确定；

（2）按 **[ALPHA]** " 调用引号。将函数解析式赋给函数名  $Y_1$ ；按 **[VARS]** **[>]**，选择 1: Function...，再选择 1:

$Y_1$  即可调用函数名。

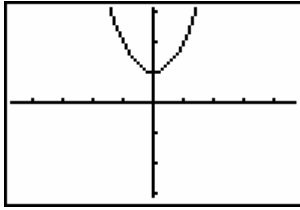


图 20

(3) 运行该程序后，屏幕上作出了函数图像。

说明：有必要时，我们还应在程序中打开坐标系，关闭其他函数图像，以免作图时出现其他不相关的信息等。

(2) 我们可以运用函数图像平移的基本思想设计程序，可参考下图中的程序 T1：

```
PROGRAM:T1
:ClrDraw
:AxesOn
:FnOff :Func
:0→A
:"(X-A)²+1"→Y₁
:Prompt A
:ZStandard
```

图 21

### 练习题：

1. 设计一个程序，使得任意输入一个函数后，程序都可以按操作者要求实现函数的上下平移。
2. 在所有的三位数中，有些数具有这样的特点：个位上的数字的立方、十位上的数字的平方与百位上的数字之和正好等于这个三位数，例如： $1^1 + 3^2 + 5^3 = 135$ 。试编程搜索，除了 135 以外还有这样的三位数吗？

## 四、拓展阅读——用程序实现互动操作

图形计算器还可以通过键盘击键达到编程者意愿，创建互动的效果。检查键盘以获取命令 `getKey`，按 `PRGM` 在 `PRGM` I/O 状态下选择 `7`

`getKey` 返回一个与最后按键相对应的数，若没有按键，则 `getKey` 返回 0。在循环体中使用 `getKey` 可转移控制。常用按键有 `DEL` `▼` `◀` `▲` `▶`，它们的键码分别是：按键 `DEL` 则返回数值为 23；按键 `◀` 则返回数值为 24；按键 `▲` 则返回数值为 25；按键 `▶` 则返回数值为 26；按键 `▼` 则返回数值为 34。在程序执行期间随时

按[ON]来中断程序执行。

从  $y = x^2$  图像，通过按上、下、左、右键来移动抛物线，并显示其顶点。

```
PROGRAM:YIDONG
:ClrHome:ClrDraw
:Func:AxesOn=
:0→M:0→K:1→A
:-8→Xmin:8→Xmax=
1→Xscl
:-6→Ymin:10→Ymax
```

```
PROGRAM:YIDONG
:1→Yscl
:“(X-M)²+K”→Y₂
:“X²”→Y₁
:FnOff
:FnOn 1
:StorePic 1
:Lbl X
```

```
PROGRAM:YIDONG
:getKey→A
:If A=24
:Then
:FnOff
:RecallPic 1
:M-1→M
:FnOn 2
```

```
PROGRAM:YIDONG
:End
:If A=26
:Then
:FnOff
:RecallPic 1
:M+1→M
:FnOn 2
```

```
PROGRAM:YIDONG
:End
:If A=25
:Then
:FnOff
:RecallPic 1
:K+1→K
:FnOn 2
```

```
PROGRAM:YIDONG
:End
:If A=34
:Then
:FnOff
:RecallPic 1
:K-1→K
:FnOn 2
```

1. 按[PRGM]◀显示 PRGM NEW 菜单，建立程序 YIDONG。
2. 给 M、K、A 赋初始值
3. 设置窗口的大小
4. 输入  $Y_1$ 、 $Y_2$  函数解析式。
5. 存储图片 1。
6. 击键值给 A。
7. 若击键值为 24，则重新调用保存的图片 1。
8. M 值减 1，图像向左平移一个单位。
9. 若击键值为 26，则重新调用保存的图片 1。
10. M 值加 1，图像向右平移一个单位。
11. 若击键值为 25，则重新调用保存的图片 1。
12. K 值加 1，图像向上平移一个单位。
13. 若击键值为 34，则重新调用保存的图片 1。
14. K 值减 1，图像向下平移一个单位。

```
PROGRAM:YIDONG
:End
:If A=23
:Then
:ClrHome:ClrDraw
:Stop
:End
:Text(55,5,"Y=X^2")
:Text(55,48,"Y=(X-")
:Goto X
```

15.若击键值为 23，则清除主屏幕及已画的内容，并结束程序。

16.显示函数解析式后返回到击键状态。