

程序设计基础训练

2020·国庆

目录

语法要点

变量与类型，输入与输出，运算，条件判断，循环，数组

程序调试

常见错误，GDB 的使用（in Dev-C++）

赛题选讲

脸熟赛F/G/I/K

语法要点

语法要点

变量与类型（输入输出），四则运算，条件判断，循环，数组，字符串

程序调试

常见错误，GDB 的使用（in Dev-C++）

赛题选讲

脸熟赛F/G/I/K

变量与类型

变量是什么？你可以把它想象成一个盒子，盒子里面可以存东西（也就是值）。

盒子有不同的类型，比如有专门放水果的盒子

变量也有不同的类型，比如只能放整数（int/long long），或者可以放小数（double）。

C 语言中的类型基本上可以分为两种，整型和浮点型。

整型包括 int/long long（最常用的两种）

还有 char（char 是放字符的，本质上和 int 没有区别，下一页会讲）

浮点型一般是 double（float 不常用，可以不记）

使用整数时要特别注意他们的存储范围，而浮点型一般不用管范围

类型	范围
int (10^9)	-2 147 483 648 ~ 2 147 483 647
long long (10^18)	-9223372036854775808 ~ +9223372036854775807

变量与类型

char 实际上和整数是一样的。

char 的范围是-128~+127，每个 char 中存的整数对应了 ascii 码表中的字符。

比如 `printf("%c", 65);` 输出了 A，因为 A 的 ascii 为 65。

类型	输入	输出
int	<code>scanf("%d", &a);</code>	<code>printf("%d", a)</code>
long long	<code>scanf("%lld", &a);</code>	<code>printf("%lld", a);</code>
char	<code>scanf("%c", &c);</code>	<code>printf("%c", c);</code>
double	<code>scanf("%lf", &x);</code>	<code>printf("%f", x);</code>

注意，在运算中浮点数不能用 `==` 判断，因为计算机的局限性，所以一般存在误差，如 3 变成了 3.0001
常用 `fabs(a-b)<(1e-6)` 代替 `a==b` （需要事先 `#include <math.h>`）

可以把整数赋值给浮点数变量，如 `double x = 1`

也可以把浮点数赋值给整数变量，如 `int a = 1.5` （此时 a 的值为 1，小数部分被截断了）

运算

常见的有 $+$ $-$ $*$ $/$ $\%$ 。还有位运算 $\&$ $|$ \sim $^$ (以后会学) 注意 $^$ 不是乘方的意思

当两个数字都是整数的时候，结果一定为整数！比如 $5/3=1$ 。（整除）

当数字中有浮点数时，结果可以为小数，如 $5.0 / 2 = 2.5$ 。（浮点数除法）

那么假设 a 和 b 都是 `int`，怎么使用浮点数除法？需要先转换类型： $1.0*a/b$ 或者 $(double)a/b$

运算一定要在数据范围之内！否则会溢出（出现奇怪的 bug，有兴趣可以自己查资料）

比如 `int a = 2147483647 + 1;` 结果居然变成了一个负数！

在涉及到负数的除法中，商的符号和符合负负得正的要求，大小为绝对值相除
余数的符号与被除数相同

$$-3 / 2 = -1;$$

$$-3 \% 2 = -1;$$

$$3 / (-2) = -1;$$

$$3 \% (-2) = 1;$$

条件判断 (if)

if (条件) 做一些事;

if (条件) 做一些事;
else 做一些事;

if (条件1) 做一些事;
else if (条件2) 做一些事;
else 做一些事;

注意, 如果 do_something 里面有多于一条的语句, 那么**一定要大括号**

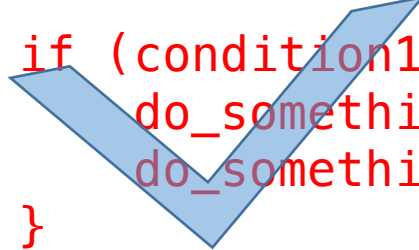
if (条件) 后面**不能直接加分号**



```
if (condition1)
    do_something1;
    do_something2;
```



```
if (condition1); do_something;
```



```
if (condition1) {
    do_something1;
    do_something2;
}
```

HINT:

你可以把大括号括起来的多条语句看成一个整体,
他们被当成了一条语句。

即 { xxx; yyy; } 和 zzz; 一样, 从整体上看都是一条独立的语句。
只不过前者是数条语句合并起来的集合体。

所以用 zzz; 的地方你都可以用 { xxx; yyy; }

个人建议不管 do_something 只有一条语句, 还是有多条语句, 都用大括号, 防止出错

循环 (for, while, do...while)

for(初始值; 条件判断; 变量递增/递减) { 做一些事 }

while(条件判断) { 做一些事 }

do { 做一些事 } while(条件判断);

循环用来重复做一些操作。和 if 一样，**注意大括号**，for(...)/while(...) **后面不能直接加分号**
注意，do...while 中要做的事写在 do 和 while 之间，所以 do...while 的 while(); 后面必须有分号。

do...while 和 while 的区别：

while 先判断再做事（一开始判断失败的话，有可能一遍也不做）

do...while 先做事再判断（至少会做一遍）

数组

假设题目要用一个变量，怎么做？ `int a;`
要用两个呢？ `int a, b;`

假设题目要用 1000 个变量？不可能用不同的字母一个个命名了。
此时要用数组 `int a[1000];`

变量是一个盒子，数组是一排盒子，每个盒子有独立的编号，从 0 开始递增
比如 `a[0]`, `a[1]`, ..., `a[999]`

读入n个数字：
`for (i=0; i<n; ++i) scanf("%d", &a[i]);`

把数组里面的 100 个数字求和：
`sum = 0;`
`for (i=0; i<n; ++i) sum += a[i];`

二维数组

二维数组可以看成是一个盒子的方阵，比如 100*100 的方阵：`int a[100][100];`

读入一个 n*m 的二维方阵：

```
for (i=0; i<n; ++i) {  
    for (j=0; j<m; ++j) {  
        scanf("%d", &a[i][j]);  
    }  
}
```

其实底层实现上二维数组本质上是一维数组，
`a[2][3]` 其实就是第 $2*100 + 3$ 个盒子

多维数组同理 `b[2][3][4]`

程序调试

语法要点

变量与类型，输入与输出，四则运算，条件判断，循环，数组，函数

程序调试

常见错误，GDB 的使用（in Dev-C++）

赛题选讲

脸熟赛F/G/I/K

常见错误

1. `scanf("%d", a);` //错误示范

注意 printf 不用 &

不加取址符 `&`，会引起各种奇奇怪怪的问题

双引号不要括住逗号后面的东西

```
2. if(a > b)
    b = a;
    a = b;
```

此写法等价于

```
if(a > b){
    b = a;
}
a = b;
```

`if` 语句后如果不加大括号，只会执行第一条末尾带 `;` 的语句

如果需要在if后执行多条语句，需要大括号 `{` 和 `}` 将语句括起来

当然如果不喜欢多行或者括号，可以写成这样：

```
if(a > b) b = a, a = b; //语句之间以逗号连接
```

常见错误

3. `for(int i = 1; i <= n; i++);`//错误示范

```
int i = 1;
while(i <= n); //错误示范
{
    //do something
    i++;
}
```

以上语句中 `for` 和 `while` 后面不要加 `;`，否则会死循环

4. 关于 `=` 和 `==`：

`=` 是赋值运算，而 `==` 是条件判断，不要混用了。正确示范：

```
if(a == b) c = d;
```

5. `for(int i = 1; i <= n; i++){`
 `scanf("%d", &i);`//错误示范
`}`

用来累加累乘（比如 `sum += i`）的变量一定要初始化！（初始化0或其他数字）

`i` 如果作为循环变量，是不能再把数存到其中的，这会导致循环错乱或者死循环

GDB

GDB 可以让你自己有能力找到程序中的大部分问题。

使用方法（以 Dev-C++ 为例）：<https://zhuanlan.zhihu.com/p/100470767>

赛题选讲

语法要点

变量与类型，输入与输出，四则运算，条件判断，循环，数组，函数

程序调试

常见错误，GDB 的使用（in Dev-C++）

赛题选讲

脸熟赛 F/G/I/K

脸熟赛 F

不要用暴力 if 判断所有情况，用 char 和 int 的关系做，用 if 判断范围。注意看题，不要遗漏

脸熟赛 G

线性规划。显然只要考虑 2:1 消耗和 1:2 消耗的两个情况。

当存在两倍关系时，答案受数量少的制约；否则考虑两种工具各用了 x 和 y 种，则可以得到

$$2x + y \leq n, 2y + x \leq m \qquad x + y \leq \lfloor \frac{n + m}{3} \rfloor$$

脸熟赛 I

变形要求可以得到 $A*B = N-C$ ($C=1..N$)，即求 $A*B = M$ ($M=1..N-1$) 的数量

直觉做法 (TLE)：求 i ($i=1..N-1$) 的因数个数，然后求和

改变计数方式：考虑 $N-C$ ($1..N-1$) 中数字的因子有多少种

考虑所有数字的因子有哪些，等价于考虑每个数字可以作为多少数字的因子

$1..N-1$ 的数字里面，有哪些 $N-C$ 含有因子 2? 2, 4, 6, ... 有 $(N-1)/2$ 个

$1..N-1$ 的数字里面，有哪些 $N-C$ 含有因子 3? 3, 6, 9, ... 有 $(N-1)/3$ 个

答案就是 $\sum (N-1)/i$ ($i=1..N-1$)

脸熟赛 K

注意数据范围，用 long long 做