**C语言手记**

**[一．简述。](#一)**

[**二．书写规范。**](#二)

**[三．（程序解释—注释）](#三)**

[**四．标识符**](#四)

[**五．变量及赋值**](#五)

[**六．基本数据类型**](#六)

[**七．格式化输出语句**](#七)

[**八．不可改变的常量**](#八)

[**九．转换类型**](#九)

[**十．运算符**](#十)

[**十一、C语言结构语句**](#十一)

**#手记**

**一.简述 。**

**1.C程序由若干个头文件和函数组成。**

**2.必须要有主函数，即main函数。**

**3.主函数只有一个。可以说主函数是C语言的唯一入口。**

**例如：#include<stdio.h> //头文件。**

**int main()**

**{**

**} //主函数。**

**二.（良好习惯之规范。）**

**1.一个说明或一个语句占一行。**

**例如：包含头文件、一个可执行语句结束都需要换行。**

**2.函数体内的语句要有明显的缩进，通常以按一下Tab键为一个缩进。**

**3.括号和引号要成对写，删除也要成对删。**

**例如：<>；””；()；{}; [];**

**4.当一句可执行语句结束的时候末尾需要有分号。**

**5.代码中所有符号均为英文半角符号。**

例如：

#include<stdio.h> //进行换行

Int main () //括号成对写

{

printf(“Hello\n”); //缩进和标点半角输入

}

**三．（程序解释—注释）**

**C语言注释方法有两种：**

1. **多行注释：/\*注释内容\*/**
2. **单行注释：// 注释一行**

**四．标识符**

**标识符是编程时给变量或者函数起的名字。**

1. **标识符可以是字母（A-Z，a-z）、数字（0-9）、下划线\_ 组成的字符串，并且第一个字符必须是字母或下划线。**
2. **标识符的长度最好不要超过8位，因为在某些版本的C中规定标识符前8位有效，当前8位相同时，则被认定为一个函数。**
3. **标识符严格区分大小写。例如：Imooc和imooc是两个不同的标识符。**
4. **标识符最好选择有意义的英文单词组成做到“见名知意”，不要使用中文。**
5. **标识符不能是C语言的关键字。[C语言关键字知识，请查阅网页。](C语言一共有32个关键字.docx)**

**五．变量及赋值**

**变量就是可以变化的量，而每个变量都有一个名字（标识符）。变量占据内存中一定的储存单元。**

1. **变量定义的一般形式为：数据类型 变量名；**
2. **多个类型相同的变量：数据类型 变量名，变量名，变量名~~；**

**例如：int a，b，c；**

1. **赋值方式：（1）先声明再赋值。**

**例如：int num;**

**num=100;**

**（2）声明的同时赋值。**

**例如：int x=100;**

1. **注意：（1）使用变量之前必须先定义变量。**

**（2）要区分变量名和变量值是两个不同的概念。**

**（3）在定义中不允许连续赋值。**

**例如：int a=b=c=9，是不合法的。**

**六．基本数据类型**

**1.C语言中，数据类型可分为：基本数据类型、构造数据类型、指针类型、空类型 四大类。如图：**

1. **整形、实型、字符型。**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据类型 | 说明 | 字节 | 应用 | 示例 |
| char | 字符型 | 1 | 用于储存单个字符 | char sex=’m’ |
| int | 整型 | 2 | 用于储存整数 | int height=18 |
| float | 单精度浮点型 | 4 | 用于储存小数 | float price=11.1 |
| double | 双精度浮点型 | 8 | 用于储存位数更多的小数 | double pi=3,1415926 |

**4．整型数据是指不带小数的数字。**

**整型的类型比较多：**

|  |  |  |  |
| --- | --- | --- | --- |
| 数据类型 | 说明 | 字节 | 取值范围 |
| int | 整型 | 2 | （-32768~32767）-215~215-1 |
| short int | 短整型（int可以省略） | 2 | （-32768~32767）-215~215-1 |
| long int | 长整型（int可以省略） | 4 | （-2147483648~2147483647）-231~231-1 |
| unsigned int | 无符号整型 | 2 | （0~65535）0~216-1 |
| unsigned short int | 无符号短整型（int可以省略） | 2 | （0~65535）0~216-1 |
| unsigned long int | 无符号长整型（int可以省略） | 4 | （0~4294967295）0~232-1 |

**5．浮点数据是指带小数的数字。**

|  |  |  |  |
| --- | --- | --- | --- |
| 数据类型 | 说明 | 字节 | 取值范围 |
| float | 单精度型 | 4 | -3.4\*10-38~3.4\*1038 |
| double | 双精度型 | 8 | -1.7\*10-308~1.7\*10308 |
| long double | 长双精度型 | 16 | -1.2\*10-4932~1.2\*104932 |

**七．格式化输出语句**

**1.格式化输出语句，也可以说是占位输出，是将各种类型的数据按照格式化后的类型及指定的位置从计算机上显示。**

**2.其格式为：printf(“输出格式符”，输出项)；**

**3.格式符的个数要与变量、常量或者表达式的个数一一对应。**

**C语言中的格式符**

|  |  |
| --- | --- |
| 格式符 | 格式符的意义 |
| %d | **以十进制形式输出带符号整数（正数不输出符号）** |
| %c | **输出单个字符** |
| %s | **输出字符串** |
| %f | **以小数形式输出单、双精度实数** |
| %o | **以八进制形式输出无符号整数（不输出前缀0）** |
| %x | **以十六进制形式输出无符号整数（不输出前缀0X）** |
| %u | **以十进制形式输出无符号整数** |
| %e | **以指数形式输出单、双精度实数** |
| %g | **以%f、%e中较短的输出宽度输出单、双精度实数** |

**八．不可改变的常量**

**常量：在程序执行过程中，值不发生改变的量。**

**C语言中的常量可以分为直接常量和符号常量。**

**直接常量也称字面量。如：13、0、13.44 等等；**

**符号常量：在C语言中，可以用一个标识符来表示一个常量。符号常量在使用前必须先定义，其一般形式为：#define 标识符 常量值**

**如：#define PI 3.1415926**

**九．转换类型**

**转换类型分为自动类型转换和强制类型转换。**

**自动类型转换发生在不同数据类型运算时，在编译的时候自动完成。**

**注意：字节小的可以向字节大的自动转换，但字节大的不能向字节小的自动转换。**

**规则：如下图：**

int

char

double

**强制类型转换是通过定义类型转换运算来实现的。**

**一般形式为：（数据类型）（表达式）**

**其作用是把表达式的运算结果强制转换成类型说明符所表示的类型。**

**如：double t=6.44;**

**Int t1=(int)t;**

**其结果为t1=6.**

**注意：**

1. **数据类型和表达式都必须加括号。**
2. **转换后不会改变原数据的类型及变量值，只在本次运算中临时性转换。**
3. **强制转换后的运算结果不遵循四舍五入原则。**

**、十．运算符**

1. **算术运算符**

|  |  |  |
| --- | --- | --- |
| 名称 | 运算符号 | 举例 |
| 加法运算符 | **+** | **2+10=12** |
| 减法运算符 | **-** | **2-1=1** |
| 乘法运算符 | **\*** | **2\*2=4** |
| 除法运算符 | **/** | **8/4=2** |
| 求余运算符 | **%** | **23%7=2** |
| 自增运算符 | **++** | **a++** |
| 自减运算符 | **--** | **a--** |

**除法运算时注意：**

**两个整数相除，结果也为整数；**

**其中一个为小数，结果为小数。**

**取余运算时注意：**

**该运算只适合用两个整数进行取余运算。**

1. **自增与自减运算符**

|  |  |  |
| --- | --- | --- |
| 运算表达式 | 说明 | 运算规则 |
| ++a | **a自增1后，再取值** | **先运算，**  **再取值。** |
| --a | **a自减1后，再取值** |
| a++ | **a取值后，a的值再自增1** | **先取值，**  **再运算。** |
| a-- | **a取值后，a的值再自减1** |

1. **赋值运算符**

**赋值运算符又分为简单赋值运算符和符合赋值运算符。**

**简单赋值运算符：=**

**符合赋值运算符：+=、-=、/=、%=、\*=；**

**如：a+=5，其意义为a=a+5；a\*=5-1，其意义为a=a\*（5-1）；**

**并且 复合运算符中运算符和等号之间时不存在空格的。**

**4．关系运算符**

**C语言中的关系运算符：**

|  |  |  |  |
| --- | --- | --- | --- |
| 符号 | 意义 | 举例 | 结果 |
| > | **大于** | **10>5** | **1** |
| >= | **大于等于** | **10>=10** | **1** |
| < | **小于** | **10<5** | **0** |
| <= | **小于等于** | **10<=10** | **1** |
| == | **等于** | **10==5** | **0** |
| != | **不等于** | **10 !=5** | **1** |

**注意：这些符号之间没有空格。**

**5．逻辑运算符**

**C语言中的逻辑运算符：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 符号 | 意义 | 举例 | 结果 | 规则 |
| && | **逻辑与** | **0&&1** | **0** | **都是真，结果为真** |
| || | **逻辑或** | **0||1** | **1** | **一真，结果为真** |
| ！ | **逻辑非** | **！0** | **1** | **变量为真，结果为假；变量为假，结果为真** |

**6．三目运算符**

**三目运算符：“？：“，**

**其格式为：**

**表达式1？表达式2：表达式3；**

**其意义：先判断表达式1是否为真，如果是真，执行表达式2，如果是假，执行表**

**达式3。**

|  |  |
| --- | --- |
| 优先级 | 运算符 |
| 1 | **（）** |
| 2 | **！、+（正号）、-（负号）、++、--** |
| 3 | **\*、/、%** |
| 4 | **+（加）、-(减)** |
| 5 | **<、<=、>、>=** |
| 6 | **==、!=** |
| 7 | **&&** |
| 8 | **||** |
| 9 | **?:** |
| 10 | **=、+=、-=、\*=、/=、%=** |

**十一、C语言结构语句**

1. **分支结构之简单if语句**

**结构： if （）**

**{**

**执行代码块；**

**}**

**注意：if（）后没有分号，直接写{}。**

**2．分支结构之简单if-else语句**

**结构：if（表达式）**

**{**

**执行代码块1；**

**}**

**else**

**{**

**执行代码块2；**

**}**

**语义：如果表达式的值为真，则执行代码块1，否则执行代码块2.**

**else是对if的全否定。**

1. **分支结构之多重if-else语句**

**结构：**

**If（表达式1）**

**{**

**执行代码块1；**

**}**

**······ · · · · ·**

**else if（表达式n）**

**{**

**执行代码块n；**

**}**

**else**

**{**

**执行代码块m；**

**}**

**语义：依次判断表达式的值，当出现某个值为真时，则执行对应代码块，否则执行代码块m。**

**下面的else时对上面的if进行全否定。**

1. **分支结构之嵌套if-else语句**

**If（表达式）**

**{**

**If（表达式）**

**{**

**执行代码块；**

**}**

**else**

**{**

**执行代码块；**

**}**

**}**

**else**

**{**

**执行代码块；**

**}**

1. **多分支结构之switch语句**

**switch（表达式）**

**{**

**case表达式常量1：执行代码块1；break；**

**···· · · · · ·**

**case表达式常量n：执行代码块n；break；**

**default：执行代码块n+1；**

**}**

**注意：**

1. **在case后的各常量表达式不能相同。**
2. **case后如果没有break，会一直往后执行一直到遇到break，才会跳出switch语句。**
3. **switch后的表达式语句只能是整型或者字符类型。**
4. **在case后可以有多个语句，可以不用{}括起来。**
5. **各case和default子句的先后顺序可以变动，而不会影响程序执行结果。**
6. **default可以省略不用。**
7. **循环结构之while循环（先判断后执行）**

**循环：反复不停的执行某个动作**

**结构：while（表达式）**

**{**

**执行代码块；**

**}**

**表达式为循环条件，执行代码块为循环体。**

**当表达式为真时，执行代码块；否则，不执行。**

1. **循环结构之do-while循环（先执行后判断）**

**do**

**{**

**执行代码块；**

**}while（表达式）；**

**do-while循环至少执行一次循环结构。**

**而且while后有分号。**

1. **循环结构之for循环**

**for（表达式1；表达式2；表达式3）**

**{**

**执行代码块；**

**}**

**表达式1：初始条件**

**表达式2：判断条件**

**表达式3：趋向结束的条件**

**注意：**

1. **分号不可省。**
2. **缺少表达式1或2变成死循环。**
3. **各个变量必须在for之前定义。**
4. **三种循环的比较**
5. **知道循环次数，用for循环；**
6. **不知道循环次数用while或者do-while循环，如果可能一次都不运行使用while循环，如果至少循环一次用do-while循环。**
7. **循环结构之多重循环**

**多重循环就是在循环结构的循环体中又出现循环结构。**

**11.结束循环之break语句**

**由于某种原因需要中断，可以使用break语句进行该操作。**

1. **结束语句之continue语句**

**由于中断之后需要继续进行操作。可以用continue语句进行该操作。**

1. **臭名远扬之goto语句**

**格式：goto 语句标号；**

**在函数某处加一个标号（如：LOOP等），然后在函数中写goto 语句标号，**

**在此时跳到LOOP处继续运行。和递归语句有些类似。**