C语言程序由函数构成（函数式），有且只能有一个main函数，开始结束都在main函数中

编写C语言程序步骤：

‘编写 ’程序 (.c/.cpp，为数据文件)

‘编译 ’源程序为目标程序 (.obj 用来检查程序是否出错，将其转化为2进制文件)

‘连接 ’编译后的文件 (将多个.obj文件组合起来)

‘执行 ’程序(.exe)

算法的特性：

有穷性：算法的步骤是有穷的

确定性：算法的步骤是确定的

有0个或多个输入：输入必要的信息

有1个或多个输出：算法的解就是输出

有效性：每一个步骤都可以有效执行

描述一个算法：

自然语言 程序流程图 N-S流程图(N-S合图) 伪代码 计算机语言

算法可以没有输入，但是一定要有输出

< > 包裹的头文件为系统库头文件 " " 包裹的头文件为用户自定义头文件

#include<....>等同于#include"...." ...中的内容也可为文件 例： #include<file.c>

<.h>为头文件 等同于 <.c>

内存单元：计算机中最小的存储单位，大小一个字节，每一个内存单元都有一个唯一的编号

转义字符占一个字节(\t \b \n \0)

变量存储:

静态存储：在程序运行期间分配固定的存储空间的方式

动态存储：在程序运行期间根据需要进行动态的分配存储空间的方式。

C程序把文件分为二进制文件和文本文件（也称ASCII码文件）

用小数或指数(e)来表示实型常量，以%f输出，默认显示小数点后6位

（考试时遇到结果以%f输出，一定要带.0）

e(E)的两端都要有数，后面的数要为整数，前后以及数字之间不得插入空格

double赋值给double还是double 复合语句是一条单语句

数组长度可以为空，若不为空只能为>0的整型常量，可以>元素个数但不能<元素个数

提前在宏定义中声明也可以( #define N 5 int a[N];)

内存字节数：数组维数 \* sizeof(元素数据类型) 数组下标必须为整数

int a[5]={1,2}; 数组中未赋值的元素系统默认赋值0

stactic int a[5]={1,2};同上

char a[5]={'1','2'}; char定义的数组未赋值的元素系统默认赋值\0

int a[ ]; 只声明而未定义的数组,系统会为其自动赋值随机数

sizeof() 求数据大小(不是函数)

数组大小 sizeof(a)); 数组元素的大小 sizeof(a[0]));

数组元素个数 sizeof(a)/ sizeof(a[0]));

数组名+1，是+数组元素大小的字节数；即数组总元素+1

&数组名+1(即数组名的地址+1) 加过一个数组的大小

二维数组行数可以为空,列数不可为空,其余属性同于一维

若光一个a[n]则代表下标为n的呢行

指针只能指向同类型数据，指针变量必须先赋值再使用

指针可以做加减和自增减，两个指针变量只能做减法

指针++，指向下一个元素 指针--，指向前一个元素

定义指针并赋值：int i=10; int\* p=&i;

此时p为指针，存着i的地址，可以理解为p就是一个地址

\*p 找到p所指地址的值，即\*p=10

已被赋值的指针可以作为另一个同类型指针的值

int\* p=0; 空指针(零指针)表示p指向地址为0的单元

野指针：指针指向某一地址直接进行使用

空指针和野指针都不是我们申请的空间，因此不要访问

void\* p=(类型)i; 给void型指针赋值时必须使用强制类型转换

p!=NULL 代表后面还有数据

const int\* p=&i; 为常量指针，const在\*p前，不能操作\*p的值

int\* const p=&i; 为指针常量，const在p前，不能操作指针

const int\* const p=&i; 指针和值都不能操作

函数不可以嵌套定义，可以嵌套使用 函数中可以有多个return语句

形参不分配内存空间

函数类型为int则以return 结尾具有返回值，类型为void则没有返回值，默认 int 型

函数声明要放在开头，若在main函数后定义函数，则一定要在开头进行函数声明

int max(int a[n]) 错误 int max(int a[]，int n) 正确

当数组作为形参时，数组名和长度分别进行传递

float max(){int a,b;return(a>b?a:b);} 最后类型为float

即当函数类型与return语句中表达式的值的类型不一样时按函数的类型为准

变量：全局变量 局部变量

定义在函数外部的变量就是全局变量，作用域从定义位置到本源文件结束

函数内部的变量就是局部变量(例如形参)

属于<stdio.h> 的函数

scanf和printf可以输入输出任何类型 printf至少要有一个参数，上不封顶

scanf接收float时用%f 接收double时用%lf，输出时都用%f

scanf("%d b=%d,c=%d",&a,&b,&c); 在终端输入时，要按 n b=n,c=n 输入

char a,b,c; scanf("%c%c%c",&a,&b,&c); printf("a=%c b=%c c=%c",a,b,c);

输入abc显示a=a b=b c=c 输入a b c 显示 a=a b= c=b 将空格算作字符接收了

结论：在scanf接收字符时，任何字符都会被接收(包括 空格、换行 )

字符变量 = getchar() 接收一个字符 putchar(字符常量/变量) 输出一个字符

gets 效果同scanf,可以接受空格 puts 输出字符串（自动添加\n换行）

属于<string.h> 的函数

strlen获取字符串长度 strcat(a,b)将ab的内容拼接赋值给a，a的内容即函数结果

strcpy(a,b)将b覆盖掉a strcmp比较字符串

属于<math.h> 的函数

sqrt 根号 fabs 绝对值 pow(x,y) 以x为底 y为幂

标识符（类型名称）不能使用关键字，只能由字母数字下划线组成且第一个字符不能为数字



main include define scanf printf 这些为预定义标识符，可以用作标识符

逗号表达式的值为最后一个表达式的值

a=3\*5,a\*4,a+5 则 a=15，该逗号表达式的值 =20

a=1;b=2;c=3; printf("%d,%d,%d",(a,b,c),b,c) 则输出 3,2,3

赋值是将 等号右边的值 赋给 等号左边的'变量'

a+b=50错，a+b是表达式 c=b=a 对，b=a 再 c=b

int a=b=c=100; 错，是将100赋值给了c，但是a,b,c均没有定义，有语法错误

单另的一个 ; 叫空语句

实型数据赋值给整型时，不进行四舍五入，直接舍弃小数部分

整型数据赋值给实型时，数值不变，以指数形式(带小数部分)存储

字符型数据赋值给整型变量，是将字符的ascll码赋给整型变量

定义结构体时关键字stryct不能省 创建结构体变量时关键字stryct可以省

结构体变量利用 . 访问结构体成员

a/b时a<b,则结果为0 a%b时a<b,则结果为a %的两边必须是整型

int n5=-7/5; //-1 先将 - 提出 若a>b 则添加 - 到结果上 若a<b 则还是0

int n6=-7%5; //-2 先将 - 提出 若a>b 则添加 - 到结果上 若a<b 则还是a

int n6=7%-5; //2 - 在 % 前，才会添加，在%号后，无论ab谁大，都不添加 -

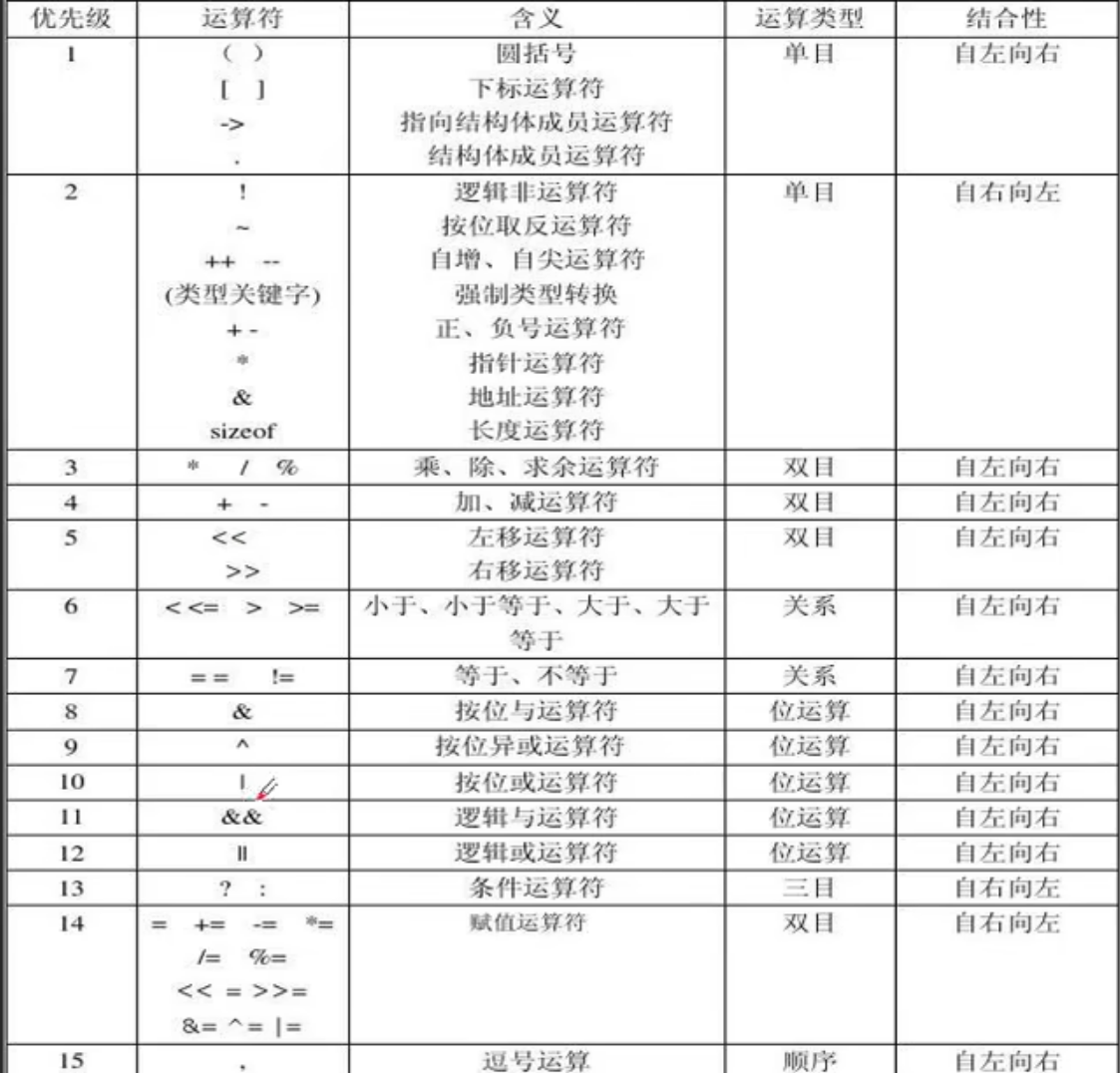
printf("%d\n",(a=2,b=5,(++a)\*b)); 15

printf("%d\n",(a=2,b=5,(a++)\*b)); 10

printf("%d\n",(a=2,b=5,a++,b++,a+b)); 9

printf("%d\n",(a=2,b=5,++a,++b,a+b)); 9

后两种在最后使用ab时，a和b已经进行了运算，所以不区分++在前在后



哪个符号的优先级大，就先算它

若优先级相等，则从左到右计算

已知 i=3, f=2.5, d=7.5 ,则 10+'a'+i\*f-d/3 计算步骤如下

107+i\*f-d/3 => 107+7.5-d/3 => 114.5-d/3 => 114.5-2.5 => 112.0

嵌套的逻辑表达式和关系表达式从左往右看，嵌套的条件表达式从右往左看

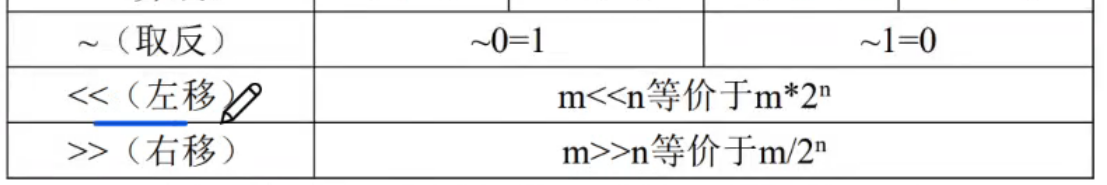
强制类型转换不会改变变量的值

低精度转高为自动类型转换 高精度转低为强制类型转换

强转可能引起精度丢失的原因是：将小数点后面的数舍弃，不进行四舍五入

可以用强转解决%两边不能为实型的问题 例：(int)9.7 % (int)2.8 = 1

运算当中只要有实型参与运算结果就是double



关系运算( < <= > >= == != )：进行关系比较，满足关系返回1，不满足返回0

在逻辑表达式 && || ! 中，非0表示真，0表示假，其值只有0和1

三种基本结构：顺序结构、选择(分支)结构、循环结构

&&(与) 同真为真，有假为假

若&&左边为真，不能判定整体真假，则要执行&&右边的部分

例：int a=10,b; b=(1&&a++); printf("%d %d",a,b); 11 1

若&&左边为假，则整体为假，不执行&&右边的部分

例：int a=10,b; b=(0&&a++); printf("%d %d",a,b); 10 1

||(或) 有真为真，同假为假

若||左边为真，则整体真假，不执行||右边的部分

例：int a=10,b; b=(1||a++); printf("%d %d",a,b); 10 1

若||左边为假，不能判定整体真假，执行||右边的部分

例：int a=10,b; b=(0||a++); printf("%d %d",a,b); 11 1

!(非) 若a为真，!a为假；若a为假，!a为真

switch(表达式){ 表达式必须是整型，值为多少则执行对应的case

case 常量:语句

case 常量:语句

case 常量:语句

(可以是多条语句，也可以一条语句也没有,每个case后面的常量必须互不相同 )

......

default:语句 (该语句可以放在任何位置，但只有判断完所有的case语句后才执行)

}

条件表达式等价于 if(表达式1){表达式2} else{表达式3}

循环的嵌套：一个循环里面包括另一个完整的循环 (无限嵌套相应的效率会立马下降)

while(表达式) {循环体}

表达式可以为逻辑表达式、数值、返回真或假的函数，只要其值!=0就执行循环体

if和while语句的区别：

if执行完后退出if语句。

while执行完后继续判断循环条件，直到循环条件为假或执行了break语句才退出循环

while(){}先判断后循环，可能一次也不执行 do{}while();先循环后判断，至少执行一次

for(初值;循环条件;循环增量){循环体}

执行步骤：初值——>循环条件 ——>循环体——>循环增量——>循环条件.....

初值只执行一次，可以是0个一个或多个(中间用逗号隔开)

若条件语句为空循环会一直执行下去

循环增量作为循环的每次调整，也可以为空，程序有可能会无限制的执行下去，可以使用break语句使其退出循环)

退出循环的两种情况：循环条件不成立、执行了break语句

无论()当中怎么变，2个‘；’都不能省略 for(;;)是合法的，程序会无限执行

break跳出整个循环用于循环体和switch语句内

continue跳出本次循环 只能用于循环体内

预编辑处理(#include #define)不是C语言中的一部分，不占运行时间

预处理命令以 # 开头，占单独书写行，语句尾不加分号

宏定义：#define 宏名 宏内容

其中不带任何符号，除非宏内容为一个语句 宏内容可以省略

作用域：从定义命令到文件结束 进行预编译时不进行语法检查

#define MAX MAX+10 错，宏内容中不能有宏名（这属于宏定义的递归）

#define MAX 80 #define MIN MAX-10 对，这种为嵌套 宏定义可嵌套不能递归

#define S(a,b) a\*b

var = S(3+2,2)

将该宏定义展开为：S(a,b) =S(3+2,2) => a=3+2,b=2 => a\*b=3+2\*2=7

例题：

#define ADD (x) x+x

main (){

int m=1,n=2,k=3;

int sum=ADD (m+n)\*k;

printf("sum=%d",sum);

}

结果为：sum=10

解析：m+n对应x，x对应x+x，即m+n+m+n

则sum=1+2+1+2\*3=10

做宏定义时一定要将替换的式子完整展开，再往里代值

宏定义 函数

处理时间 编译时 程序运行时

参数类型 无类型问题 定义实参、形参类型

处理过程 不分配内存 分配内存

简单的字符置换 先求实参再代入形参

程序长度 变长 不变

运行速度 不占运行时间 调用和返回占时间