/\* \*/ 或// 注释

printf 函 数 输出 可以输出汉字，数字等，数字运算自动执行

； 相当于句号

{ }中内容按顺序执行

转义字符：

\n 换行

\a 响铃

int 定义变量（整数） %d

char 定义变量（字符）%c

float 定义变量（小数）%f

double 定义变量（双精度数）%le

if() 括号中直接填内容：某个变量存在，某个变量大小......

|| 或

M变量=（c=='a'||c=='b'） 当c变量为a或b时M变量为1，否则为0（仍然需要提前定义变量，相当于给M赋值）

== 判断等于

!= 不等于

= 赋值

&& 同时

while() 括号内填成立条件

puts() 与printf()基本相同，但不可进行格式设定和数值输出，但可自动换行。（puts实参只能有一个）

+ - \* / 运算符

if(b)==if(b!=0)

for(a;a<n;a++)

{

} 对a，在a<n时，a加1，同时进行一下循环

a=(double)b/c 在a，b，c均为int型情况下输出double型 == a=1.0\*b/c

&& || 称为逻辑与运算符；

break 脱离某个语句或循环

switch（变量名）

{

case 0 :printf....

case 1 :printf....

....

} 与if相似，但更加简便(实例空格不可省略)（其中case后是该变量的值）

do{

}while() 当......时执行.....循环（至少执行一次）

!a 逻辑非运算符

do{

}while(!( )) 加！，当....不是 .....时执行.....循环；

x && y == !(!x || !y)

x || y == !(!x && !y);(德摩根定律)

a@=b 复合赋值运算符（@=可以是:\*=,/=,+=,%=,-=,<<=,>>=,&=,^=,\=）;

++ 后置递增运算符；

-- 后置递减运算符；

putchar('@') @为任意字符。将char指定的字符写入。

当输入字符时，putchar('') == printf(" ") ;

printf（"%d"，++i） 先增加后显示

printf（"%d",i++） 先显示后增加 ；

#define 函数名称(变量) 算法 例子：#define SUB(X,Y) (X)\*Y；也可以定义数值；

#define a b 将该指令之后的a替换为b；

scanf\_s("%d",&M[i]); 对M[]进行赋值，加上for可以依次赋值;

continue 重复执行，剩余循环语句直接跳过；

数组赋值举例：

1、int a[5]={1,2,3,4,5,}; (最后一个逗号可以省略)

2、int a[]={1,2,3,4,5}; (不指定元素个数，数组会根据初始值的个数自动进行设定)

3、int a[5]={1,2}; (未手动初始化的初始值默认为0)

4、int a[5]={0}; (用{0，0，0，0，0}初始化)；

int a[3][4] 是3行4列的二维数组的声明；

x(arg) 函数调用运算符 向函数x传递实参arg并调用（当实参有多个时，用逗号分隔）。（如果返回值类型不是void）生成函数x返回的值；

a ? b : c 条件运算符 如果a不为0（或a条件成立），则结果是b的值，否则为c的值。简单来说就是a成立得b，不成立得c；

创建函数：

Int/char/double/void/…. +函数名称(输入形式举例)；

调用函数：

函数名称（输入数值）；

Extern int/char/double/…. +变量名 并非创建出变量实体，表明使用的是某处创建的该变量，称为非定义声明；

Const 将某变量定为常量（无法对其赋值等改变操作）

Const int a;即定义一个不变量a（一般用在函数，放置调用函数导致变量值改变）

函数名(const int a) 声明不改变所接收的数的值；

While（1）

{

一直执行{}中语句遇到return，break。。。

}

Static 存储类说明符；

Pow(a,b) == a的b次方（该函数在math.h头文件中，使用需要先include）;



Sizeof a 求a（对象、常量、类型名等）的长度；

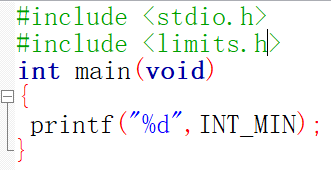
Sizeof（类型名） 

Sizeof 表达式 如果想了解变量或表达式的长度，使用该式（该式虽然不需要括起表达式的小括号，但这样可能不方便理解，因此最好是加上括号）；

Size\_t==unsigned 定义在头文件<stddef.h>中

Typedef A B 为已有类型A创建别名B

#include <limits.h> 可查看各种型的范围及CHAR\_BIT(char的位数)

写法：

#include <stddef.h> 定义了各种变量类型和宏

>> 右移

<< 左移

>>= <<= 参考 a += b

Getchar（） 读取字符并将其返回。输入结束或读取过程中发生错误时，就会返回EOF值；

%s 字符串声明

<ctype.h> 内含toupper和tolower

Toupper 将小写英文字母转换为相应的大写英文字母

Tolower 将大写英文字母转换为相应的小写英文字母

%p 表示对象地址的转换说明