计算机导论与程序设计学习报告

计算机工作原理：存储程序+程序控制

以二进制形式输入指令，存储后由控制器读取并执行。多个指令组成完备的程序控制。

十进制

(123)10 = 1 × 102 + 2 × 101 + 3 × 100 ;

二进制

(77)10=(01001101)2=0×27+1×26+0×25+0×24+1×23+1×22+0×21+1×20；

//通常PC的存储器以1字节8位形式安排，可以书写为2位16进数字；

4位二进制刚好对应一个16进数字，如果是7位或9位就在前面补0，凑成4的倍数就行。

字节//1字节=1B，（通过网络传输||在硬盘或内存中存储信息）的单位。

一个英文字母不分大小和英文标点都只写占一个字节的空间，一个中文汉字与中文标点都占两个字节的空间；一个二进制数字序列，在计算机中作为一个数字单元，一般为8位二进制数，如一个ASCII码就是一个字节；

ASCII（American Standard Code For Information Interchange）美国标准信息交换代码；基于拉丁字母的一套电脑编码系统，主要用于显示现代英语和其他西欧语言。如今最通用的单字节编码系统，并等同于国际标准ISO/IEC 646。



%//11%2=1即取余数。

**/**//11/2=5即取商。整数相除结果为整数

原码：正数符号位为0，负数符号位为1.

对于正数，原码=反码=补码；

对于负数，原码=反码符号位不变，其他位按位取反；补码=反码+1；

引入补码后，使减法统一为加法。 (+77)补 + (−77)补 = 0100 1101 + 1011 0011 = 0000 0000。

ASCII 码连续排列 ‘0’∼‘9’, ‘A’∼‘Z’, ‘a’∼‘z’；

大小字符间隔: ‘a’ - ‘A’ = 32；

程序设计的任务

问题分析---设计算法---编写程序---对源程序进行编辑、编译和连接---运行程序分析结果---编写程序文档；

#include<stdio.h> // standard input/output编译预处理指令

int main() // 主函数

{ // 函数开始标志

printf(“...”); // 输出一行信息

return 0; // 函数执行完毕返回函数值0

} // 函数结束标志

开始---编辑---编译---是否有语法错误---执行---结果对不对。选择“文件”菜单, 选择“源文件”, 编辑程序。 保存时，保存为.cpp 或.c 文件。 选择“编译和运行”菜单，生成.exe 文件，运行程序。 int a,b; // 定义变量a, b为整型数值，同类型变量可以在一条语句中定义。 float f; // 定义变量f为单精度浮点数

double d; // 定义变量d为双精度浮点数

char c; // 定义变量c为单个英文字母

格式符 对应的数据类型 备注

%d int

%f float

%c char

%lf double

%.2f float 保留两位小数, 四舍五入。不适用于 scanf()。

%.2lf double 保留两位小数, 四舍五入。不适用于 scanf()。

%x int 十六进制显示

%ld long int

==表判断是否相等= 表判断是否不相等 =表赋值 \n表换行符 i++//i=i+1 a=i++//a=i,i++ a=++i//i++,a=i

&&//且 ||//或 ！//非

if(条件表达式)

{ 表达式为真 (非 0) 时执行语句; }

#include <stdio.h>

int main()

{

int a=10;

if(a>=10)

{

printf(”a>=10\n”);

}

else

{

printf(”a<10\n”);

}

return 0;

}

while(条件表达式)

{ 表达式为真 (非 0) 时执行的语句;} #include

int main()

{

int a=10;

while(a>=0) //若a大于0，重复执行{}中的内容

{ printf(”a=%d\n”,a);

a--;

}

return 0;

变量在使用之前首先要定义它的数据类型。

输出语句 printf(“原样输出, % 格式符”, 对应变量值);

输入语句 scanf(“% 变量格式符”, & 变量名);

字符输出函数 putchar

char a = ‘B’; //定义1个字符变量并初始化

putchar(a); //向显示器输出字符B

putchar (‘\n’); //向显示器输出一个换行符

字符输入函数 getchar

char a

a = getchar(); //从键盘输入一个字符，送给字符变量a

字符缓冲区

getchar()函数等待输入直到按回车才结束（前提是缓冲区没有数据），回车前的所有输入字符都会逐个显示在屏幕上。但只有第一个字符作为函数的返回值

#include<stdio.h>  
#include<conio.h>  
void main()  
{  
    char c;  
     c=getchar();  
           //getchar()在这里它只返回你输入字符串的第一个字符，并把返回值赋值给c  
    putchar(c);  
    printf("/n/n");  
}  
 第一次getchar()时，确实需要人工的输入，但是如果你输了多个字符，以后的getchar()再执行时就会直接从缓冲区中读取。

#include<stdio.h>

#include<conio.h>

void main()

{

    char c;

    while ((c=getchar())!='/n')

     printf("%c",c);

    printf("/n/n");

}输入的字符串放在内存的缓冲区中，调用一次getchar()就把缓冲区中离出口最近的一个字符输出，即最前的一个字符输出，输出后，后面还有字符串，可以用循环把最前面的一个字符一个个在内存中释放掉，直到不满足循环条件退出。

例子中循环条件里的'/n'实际上是你输入字符串后的回车符，意思就是说，遇到回车符才结束循环，而getchar()函数是等待输入（或缓冲区中的数据）直到按回车才结束，所以实现了整个字符串的输出。也可以把循环条件改一下，比如while ((c=getchar())!='a')，意思是遇到字符'a'就停止循环，如果你输入“12345a213123/n”那么只会输出到a，结果是12345a。

注意：用getchar()它是从“流”中间去读取，所以第一个getchar()接受的是刚刚中断的流队列中即将出列的第一个字符（不限于回车符，上面举过例子），如果流队列不为空，执行getchar()就继续放，直到把回车符也放空为止，空了之后再在执行getchar()就停下等待你的输入；

    补充：按键盘上的回车产生了2个字符:回车符('/r')和换行符('/n')。回车符'/r'(CR:carriage return:倒车）使光标回到这行的首部，换行符('/n')(new line)然后再换行。

所以当输入字符'w',并按下回车键以后。首先得到回车符。那个getchar函数结束了。但是还存在一个换行符。所以如果用getchar()来做判断的时候。最好再写一次getchar()清除缓冲区的'/n'.

键盘输入的字符都存到缓冲区内,一旦键入回车,getchar就进入缓冲区读取字符,一次只返回第一个字符作为getchar函数的值,如果有循环或足够多的getchar语句,就会依次读出缓冲区内的所有字符直到'/n'.之所以你输入的一系列字符被依次读出来,是因为循环的作用使得反复利用getchar在缓冲区里读取字符,而不是getchar可以读取多个字符,事实上getchar每次只能读取一个字符.如果需要取消'/n'的影响,可以用getchar();来清除,这里getchar();只是取得了'/n'但是并没有赋给任何字符变量,所以不会有影响,相当于清除了这个字符。

常量

#define PI 3.14 // 符号常量, 注意没有分号

标识符

标识符就是一个对象的名字。用于标识变量、符号常量、函数、数组、类型等。 以字母或下划线开始; 区分大小写; 不能使用关键字; 最好有含义。

break,continue 改变循环执行的状态

while(表达式)

{

printf(”语句1”);

if(条件表达式) break; //提前终止循环

printf(”语句1”);

}

while(表达式)

{

printf(”语句1”);

if(条件表达式) continue; //结束本次 循环, 进入下轮循环 printf(”语句1”);

}

break 语句只能用于循环语句和 switch 语句之中

区别：continue 语句只结束本次循环，而非终止整个循环。break 语句结束整个循环，不再判断执行循环的条件是否成立。

数组

数组是一组有序数据的集合。

数组中各数据的排列是有一定规律的，下标代表数据在数组中的序号，用数组名和下标即可唯一地确定数组中的元素。

数组中的每一个元素都属于同一个数据类型。

定义数组: int a[10];数组元素的下标从0开始，int a[10]; 10个整型元素，则最大下标值为9，不存在数组元素a[10]。

冒泡排序

#define N 10 // 数组长度 (N − 1) 趟外层循环, (j = 1, 2, ..., N−1), 表示第j趟比较。(N−j)次内层循环,(i = 0, 1, ... , N−1−j)相邻元素两两比较,必要时交换。注意检查数组边界条件, 不要越界。分别进行N=1,2,3,4个数排序演练。

#define N 10 // 数组长度

int a[N]={9,8,7,6,5,4,3,2,1,0},i,j,t;

for(j=1;j<=N-1;j++) //进行N-1次循环，实现N-1趟比较

{

for(i=0;i<=N-1-j;i++) //在每一趟中进行N-j次比较相邻元素两两比较

{

f(a[i]>a[i+1]) //相邻两个数比较, 注意检查数组不要越界

{

t=a[i]; a[i]=a[i+1]; a[i+1]=t;

} // 交换

}

} printf(”\n the sorted numbers :\n”);

for(i=0;i<N;i++) printf(”%d ”,a[i]);

用break可以优化。