**04 流程控制：单分支和双分支**

（一）编程思路：输入 –〉处理 –〉输出 （即IPO模型）

（二）例题选择思路：

1. 编写一个程序，要求用户输入一个24小时制的时间(不考虑小时部分为0点和12点的情况)，屏幕显示12小时制的格式。（此题包含if分支的2种情形）

* 代码一：

int main( void )

{

int h, m;

int timeFlag = 0;// 0 - AM 1 - PM

printf( "Input a time: " );

scanf( "%d:%d", &h, &m );

if( h > 12 )

{

h -= 12;

timeFlag = 1;

}

if( timeFlag == 0 )

{

printf( "12-hour time: %02d:%d AM\n", h, m );

}

else

{

printf( "12-hour time: %02d:%d PM\n", h, m );

}

return 0;

}

* 代码二：

int main( void )

{

int h, m;

int newH;

printf( "Input a time: " );

scanf( "%d:%d", &h, &m );

if( h > 12 )

{

newH = h – 12;

}

if( h < 12 )

{

printf( "12-hour time: %02d:%d AM\n", h, m );

}

else

{

printf( "12-hour time: %02d:%d PM\n", newH, m );

}

return 0;

}

2．重新编写程序，要求考虑小时部分为0点和12点的情况。（此题包含if语句的嵌套）

( 提示：输入范围：00:00~23:59输出范围：01:00~12:59 )

（三）练习题目：

1. 从控制台输入整数a和整数b，若a2+b2大于100，则输出a2+b2百位以上的数字，否则输出两数之和。

2. 从控制台输入三个正整数，然后按照从大到小的顺序输出。

3. 判断从控制台输入的正整数是否既是5又是7的倍数，若是则输出yes，否则输出no。(使用嵌套if语句实现)

4．P215 第7题（求税率部分用嵌套if语句实现）

**04 流程控制：多分支**

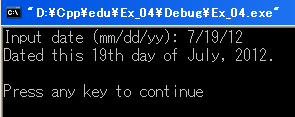
（一）例题选择思路：

1．编写一个程序，把数字显示的成绩转化为字母显示的等级。 使用下面这套评定规则：A＝90～100，B＝80～89，C＝70～79，D＝60～69，F＝0～59。如果成绩在0～100范围之外，则显示出错信息。（使用if级联实现）

2．合同和其他法律文档中经常使用下列的日期格式：

Dated this \_\_\_\_\_\_ day of \_\_\_\_\_\_, 20\_\_\_.

编写程序用来显示这种格式的日期。要求用户按照“月/日/年”格式输入日期，然后屏幕显示出法定的日期。运行结果如下：



（二）练习题目

1．从键盘输入1个数，如果大于0就打印出“正数”；如果等于0就打印出“0”；如果小于0则打印出“负数”。 （使用if级联实现）

2．送花待遇，用户输入花的价格后，程序根据不同的花价输出相应的待遇。如果多于100元，输出：“very happy！”，如果多于50元，输出：“happy!”，如果多于10元，输出：“thanks!”，否则，输出：“coldly!”。

3．P215第7题的求税率部分。（使用if级联实现）

4．P215第10题。（使用if级联实现）

5．编写一个程序，要求用户输入一个两位数，然后显示这个数的英文单词：

Enter a two-digit number: 45

You entered the number forty-five

提示：把数分解成两个数字。用一个switch语句显示第一位数字对应的单词(如“twenty”)，用第二个switch语句显示第二位数字对应的单词。考虑11～19特殊处理要求。

6．使用switch编写一个程序，把数字显示的成绩转化为字母显示的等级。 使用下面这套评定规则：A＝90～100，B＝80～89，C＝70～79，D＝60～69，F＝0～59。如果成绩在0～100范围之外，则显示出错信息。

**05 流程控制：**while**循环**

（一）讲解思路：



1．找出反复要做的事；

2．找出不做的条件；( i > n )

3．确定要做的条件；( i <= n )

4．循环流程图将循环变量步长的调整单独列出来，与反复要做的事并列。

（二）例题选择：

1．求1到100的总和； –〉奇数和偶数分别求和；

( 写出代码后要求单步执行，熟悉循环的执行顺序，同时观察各个变量中值的变化。 )

数字求和讲解技巧：1代表有1个球，2代表有2个球，以此类推，目的是把这些球按顺序依次放到1个盒子里，放完后盒子中的最后数量就是和。还需要一个袋子，作用是临时装球使用。具体操作步骤：先用袋子装1个球放到盒子里，再用袋子装2个球放到盒子里，如此操作下去直到所有球都放完。

2．求阶乘n！；

（三）练习题目

1．编写程序输出2000到2050年之间的所有闰年。判断一个年份是闰年的方法是：该年份能被4整除，但不能被100整除；或者该年份能被400整除。

2．编写一个程序计算用户输入的整数的位数，具体方法是把输入的整数反复除以10，直到结果变为0为止。除法的次数就是所求的位数。运行交互如下：

Enter a nonnegative integer: *368*

The number has 3 digit(s).

* 思路分析：

(1) 反复要做的事：

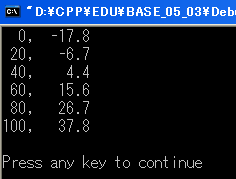
a、取出最低位，计数器加1；( data % 10; count++; )

b、抹去最后1位；( data /= 10; )

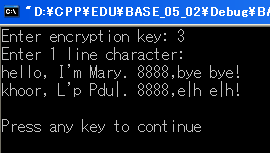
(2) 不做的条件：

data == 0

3．华氏温度和摄氏温度的转换公式为：c = 5 / 9(f – 32)，其中c表示摄氏温度，f表示华氏温度。要求输出从华氏0~360度的华氏-摄氏温度对照表，每隔华氏20度输出一组对照值。 运行界面如下：



4．从键盘输入一行字符，将其中的英文字母进行加密输出(非英文字母不用加密)。具体加密方法：是通过屏幕输入一个整数，然后，将字母的值加上这个整数，再转换为字母进行输出。 运行界面如下：



5．编写一个程序对用户输入的整数数列进行求和计算，输入以0作为结束标志。下面显示的是人机交互的内容：

This program sums a series of integers.

Enter integers (0 to terminate): *8 23 71 5 0*

The sum is: 107

* 思路分析：

(1) 反复要做的事：

a、读入一个整数；( scanf( data ); )

b、将这个整数累加到容器中；( sum += data; )

(2) 不做的条件：

data == 0

6．输出斐波那契(Fibonacci)数列(规律：第1、2个数是1，后面的数是前面2个数的和)的前40个数。斐波那契数列的前几项是1、1、2、3、5、8、13、21、34.....。要求每行输出4个数。

* 思路分析：

(1) 反复要做的事：( count = 0; )

a、产生第3项数据；计数器加1；( n3 = n1 + n2; count++; )

b、输出新数据；按需要输出换行符；( printf( n3 ); count %4 == 0 )

c、调整前2项数据，为下次循环做准备；

(2) 不做的条件：

count <= 40

7．小球从100米高处自由落下，着地后又弹回高度的一半再落下，当第10次着地时，小球共通过了多少路程？

8．P175第6,10,12题

9．P215第2,3题

10．求Fibonacci数列中大于n的第一个数及其在Fibonacci数列中的序号。

* 思路分析：

(1) 反复要做的事：( index = 0; )

a、产生第3项数据；序号加1；( n3 = n1 + n2; index ++; )

b、调整前2项数据，为下次循环做准备；

(2) 不做的条件：

n3 > n( 键盘输入 )

int main( void )

{

int data;

int i, n1, n2, n3;

scanf( "%d", &data );

n1 = 0; n2 = 1;

n3 = n1 + n2;

i = 2;

while( n3 <= data )

{

n1 = n2;

n2 = n3;

n3 = n1 + n2;

i += 1;

}

printf( "%d, %d\n", i, n3 );

return 0;

}

11．用下列公式计算PI的近似值，直到最后一项的绝对值小于10-7。

PI/4 = 1 – 1/3 + 1/5 – 1/7 + 1/9 – 1/11 + ……

* 思路分析：

(1) 反复要做的事：

a、产生1项数据；( num = 1.0 / d; )

b、将这项数据与符号位相乘后累加到容器中；( sum += sign \* num; )

c、d += 2; sign = -sign;为下次循环做准备；

(2) 不做的条件：

num < 1E-7

**05 流程控制：**for**循环**

（一）讲解思路：

1. 跟while循环对比讲解；

2. 可以认为for的流程图是while流程图的细化；

（二）例题选择：

1．使用for循环实现1到1000的之间所有奇数的总和。

2．两种方法输出下列图形。（使用单循环和双循环来实现）



* 思路分析：

(1) 反复要做的事：( 初始化row = 0; )

a、输出字符串”ABCDEFGHIJ”；( 有待细化; )

b、输出一个换行符；( printf( “\n” ); )

(2) 不做的条件：

row >= 6

(3) 细化上面循环体中的a步骤：( 功能：输出i个字符$ )

(A) 反复要做的事：( 初始化ch = ‘A’; )

a、输出变量ch中字符；( printf( ch ); )

b、ch++；

(B) 不做的条件：

ch >= ‘A’ + CHARS (CHARS = ‘J’ – ‘A’ + 1 )

3．打印如下图形。



（三）练习题目：

1．打印如下图形。

* 思路分析：

(1) 反复要做的事：( 初始化i = 0; )

a、输出i个字符$；( 有待细化; )

b、输出一个换行符；( printf( “\n” ); )

(2) 不做的条件：

i >= 6

(3) 细化上面循环体中的a步骤：( 功能：输出i个字符$ )

(A) 反复要做的事：( 初始化j = 0; )

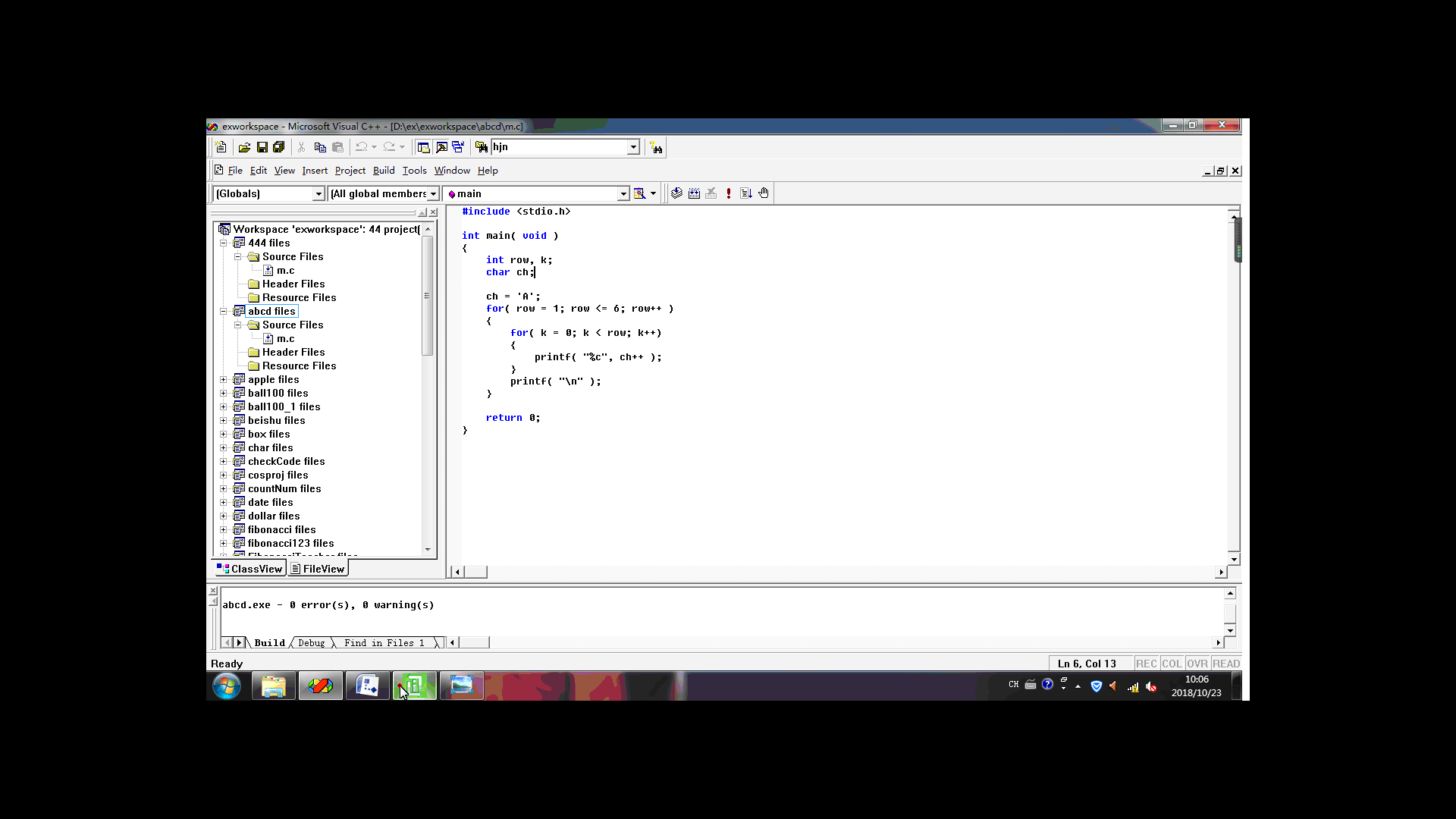
a、输出1个字符$；( printf( “$” ); )

(B) 不做的条件：

j >= i + 1

2．P174~175第3,4,5题





int main( void )

{

int row;

char ch, firstCh;

ch = 'A';

for( row = 0; row < 6; row++ )

{

firstCh = ch;

for( ; ch < firstCh + row + 1; ch++ )

{

putchar( ch );

}

printf( "\n" );

}

return 0;

}

3．P215第9题

4．编程实现累加100~1000之内能被3整除的偶数。

5．求分数序列前50项之和。

2/1, 3/2, 5/3, 8/5, 13/8, 21/13, ……

6．求数列：a, aa, aaa, aaaa, ……, aa…a前N项和，第N项有N个a，a和N均为正整数，0<=a<=9，0<=N<=9，a和N从键盘输入。

7．输入一个整数，判断这个整数是否为完数。如果是，输出Yes，否则输出No。（一个数除去本身的所有约数之和等于该数的数，称为完数）

8．输入一个整数，判断这个整数是否为素数。如果是，输出Yes，否则输出No。

9．输入一个整数，判断这个整数是否为回归数。如果是，输出Yes，否则输出No。（一个n位整数，其各位数字的n次幂之和等于这个整数，这个整数称为n位n次幂回归数。）

（例：153，371，407，1634，54748，548834）

**05 流程控制：**do while**循环**

（一）讲解思路：（跟while循环一样）

1．找出反复要做的事；

2．找出不做的条件；( i > n )

3．确定要做的条件；( i <= n )

4．循环流程图将循环变量步长的调整单独列出来，与反复要做的事并列。

（二）例题选择：

1．求1到100的总和； –〉奇数和偶数分别求和；

（三）练习题目：

1. 已知有一个整数num1为2000，从键盘输入一个整数num2，然后用num1减去num2，如果结果大于0，继续输入整数， 循环以上的操作，直到计算结果的值为负数为止。

2. 阶乘的计算；

3. 编程实现反复从键盘读取字符，统计大写字母、小写字母、数字和其他字符的个数，读取到字符$时结束程序。

4. 实现一个菜单的功能。

1.boot

2.reboot

3.safe mode

4.debug mode

5.exit

输入1～4后，表示各自标题的内容；输入5退出程序。（流程图如下所示）

提示：预先准备一个循环退出标志doExit，并doExit = 0；将选项5作为多分支的一种进行处理，当用户选择5时，将doExit = 1；将doExit作为循环的退出条件。



5．输入一个整数，并要保证是非负数。如果输入的是负数，提示出错信息，并要求继续输入，直到输入为非负数为止。

6．P215第8题。（流程图如下所示）



7．求Fibonacci数列中大于n的第一个数及其在Fibonacci数列中的序号。

int main( void )

{

int data;

int i, n1, n2, n3;

scanf( "%d", &data );

n1 = n2 = 1;

i = 2;

do

{

n3 = n1 + n2;

i += 1;

if( n3 > data )

{

printf( "%d, %d\n", i, n3 );

}

n1 = n2;

n2 = n3;

} while( n3 <= data );

return 0;

}

**06\_数组**

练习思路：

1. 从键盘输入10个数，找出最小值和最大值，将这10个数和最大数和最小数输出来。

2. 有一个数列：28, 14, 5, 10, 23, 344, 12

(1) 求数列中所有数值的和。

(2) 猜数游戏：通过控制台输入任意一个数据，判断数列中是否包含此数，输入-1停止游戏。

3. 编写程序用来检查一个整数中是否有出现多于一次的数字。用户输入数后，程序显示信息Repeated digit或No Repeated digit。使用户可以连续进行重复数字的判断。当用户录入的数-1时，程序终止。

例如：数字939577有一个重复的数字(即7)，而35869这样的数则没有。运行界面如下：

Enter a number：939577

Result：Repeated digit

Enter a number：35869

Result：No repeated digit

Enter a number：-1

(1) 使其可以显示出哪些数字有重复(如果有的话)：

Enter a number：939577

Repeated digit(s)：7 9

(2) 使其打印出一份列表，显示出每个数字在数中出现的次数：

Enter a number：41271092

Digit：0 1 2 3 4 5 6 7 8 9

Occurrences：1 2 2 0 1 0 0 1 0 1.

4．用数组输出Fibonacci数列中前100个元素的值，每行5个元素的形式输出。

5．P175第14题

6．从键盘输入10个整数存放到数组中，将数组的内容前后倒置，然后输出这些元素。

7．编写程序，实现将2个一维数组的对应元素相加，把结果存放在第3个数组中，并输出。

6. 如果要将数组 a 的元素按列依次输出，即先输出第 0 列，再输出第 1列……，该如何编写？

7. 实现矩阵的转置。把一个m行n列的数组，每一个元素行列互换，形成一个新的数组输出。

8. 在m行n列的二维数组中，找出数组每行的最大值，每列的最小值。

9. 求n\*n矩阵的主对角线和次对角线之和；

**07\_函数**

0．将课本p231中的代码录入( checking.c )，然后调试分析程序的功能。

1．输出6 ~ 10000之间的完数。（一个数除去本身的所有约数之和等于该数的数，称为完数）

定义一个函数，用于判断x是否是完数，如果是，返回1，否则，返回0。在main函数调用这个函数来实现。

2．设计一个函数edge，实现：根据直角三角形的两条直角边长度，求斜边的长度。在main函数中输入两条直角边的长度，调用edge函数求斜边长度。

3．编写一个函数square( )，输入一个整数，返回它的平方；再编写一个函数cube( )；输入一个整数，返回它的立方；用函数square( )和cube( )编写函数quartic( )和quintic( )，它们返回一个整数的四次方和五次方。用这些函数编写一个显示从1到25的整数冪的表的程序。程序的输出应该具有如下的形式：

A TABLE OF POWERS

---------------------------------

Integer Square Cuble Quartic Quintic

---------- ---------- -------- ----------- ----------

1 1 1 1 1

2 4 8 16 32

3 9 27 81 243

4. 课本p275~276：复习题中第2，3，7，9；编程练习中的第2，3，4

5. 编写一个程序，提示用户输入3组数，每组数包含5个double类型的数。该程序应完成下列任务。

a.把用户输入的数据储存在3×5的数组中

b.计算每组（5个）数据的平均值

c.计算所有数据的平均值

d.找出这15个数据中的最大值

e.打印结果

每个任务都要用单独的函数来完成（使用传统C处理数组的方式）。完成任务b，要编写一个计算并返回一维数组平均值的函数，利用循环调用该函数3次。对于处理其他任务的函数，应该把整个数组作为参数，完成任务c和d的函数应把结果返回主调函数。（课本p318：13；二维数组和函数方面的内容）

6. 编写一个函数，计算下列各组数值的平方和。要求主程序只能提供1个数组保存第1组数据。

（1） 25， 88， 46， 32， 15， 27， 33， 48

（2） 32， 15， 27， 33， 48

（3） 88， 46， 32， 15， 27

**08\_位运算**

总结：

1、&运算符

1 & bit 表示取出或保留bit位

0 & bit bit位清零

2、|运算符

1 | bit bit位置1

0 | bit 表示取出或保留bit位

3、^运算符

1 ^ bit bit位取反

0 ^ bit 表示取出或保留bit位

将data的第4~10位取出。

data

31 24 16 8 4 0

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

1111 1111 1111 1111 1111 1111 1111 1111 ~0 1

0000 0000 0000 0000 0000 0111 1111 0000 0x000007f0 mask

data & 0x000007f0>>4

练习：

设有无符号整数data，编写函数实现如下功能。

1，将data的第i位置1；

2，将data的第i位清0；

3，将data的第i位取反；

4，取出data的第i位的值；

5，将data的第i位到第j位置1；

6，将data的第i位到第j位清0；

7，将data的第i位到第j位取反；

8，取出data的第i位到第j位的值；

**09\_单链表**

1、单链表的创建

（1）尾插法

1. 准备头指针h
2. 申请一个新的结点
3. 填充数据
4. 将h指向此结点或将前一个结点的next指向此结点。
5. 重复2,3,4；

**void create( struct node \*\*h)**

{

struct node \*head = NULL;

struct node \*tail = NULL;

struct node \*p = NULL;

int data;

scanf( "%d", &data );

while( data != -1 )

{

p = (struct node \*)malloc( sizeof( struct node ) );

if( p == NULL )

{

exit( 0 );

}

p->data = data;

p->next = NULL;

if( head == NULL )

{

head = p;

}

else

{

tail->next = p;

}

tail = p;

scanf( "%d", &data );

}

\*h = head;

return;

}

**struct node \*create( void )**

{

struct node \*head = NULL;

struct node \*tail = NULL;

struct node \*p = NULL;

int data;

scanf( "%d", &data );

while( data != -1 )

{

p = (struct node \*)malloc( sizeof( struct node ) );

if( p == NULL )

{

exit( 0 );

}

p->data = data;

p->next = NULL;

if( head == NULL )

{

head = p;

}

else

{

tail->next = p;

}

tail = p;

scanf( "%d", &data );

}

return head;

}

（2）头查法

1. 准备头指针h
2. 申请一个新的结点
3. 填充数据
4. 将此结点插入到头结点的前面
5. 重复2,3,4；

2、单链表结点的插入

将数据插入到第n个结点的后面

**void insertNode( struct node \*h, int n );**

3、单链表结点的删除

删除指定数据的结点。

算法：

1. 找到要删除结点的前一个结点，用p指向；
2. 用q指向要删除的结点；
3. 删除结点；
4. 释放q。

**void deleteNode( struct node \*\*h, int data )**

{

struct node \*p = NULL;

struct node \*q = NULL;

if( (\*h)->data == data )

{

q = \*h;

\*h = q->next;

free( q );

}

else

{

p = \*h;

while( p->next != NULL && p->next->data != data )

{

p = p->next;

}

if( p->next != NULL )

{

q = p->next;

p->next = q->next;

free( q );

}

}

return;

}