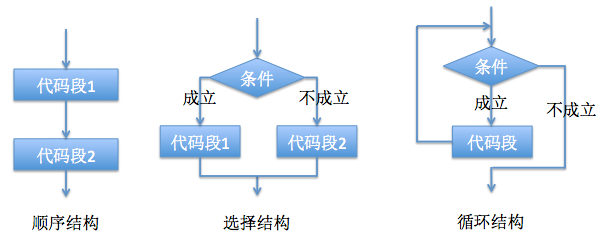
# 流程控制

* 顺序结构：默认的流程结构。按照书写顺序执行每一条语句。
* 选择结构：对给定的条件进行判断，再根据判断结果来决定执行哪一段代码。
* 循环结构：在给定条件成立的情况下，反复执行某一段代码。



# 选择结构-if

## 简单使用

* if (表达式) 语句1;
* if(count>50) 开班; 上课;
* if(表达式) 语句1; else 语句2;
* f(count>50) 开班; else 不开班;
* if(表达式) { }
* if(count>50) {开班; 布置课室;} else {}
* { }的作用域
* if -else if-else
* if（a==0） else if (a>0) else
* 特点：只有一个括号会被执行
* 复合条件
* 上课时间的取值范围(9~12 || 14 ~17)
* 陷阱
* if(a!=0) { a不是0;} else {a是0;}
* if (a); { }
* if (a=0)
* a == 0;
* if (10) int a = 9;

## 习题

1. 输入一个整数day代表星期几，根据day的值输出对应的星期几，比如day==1，就输出“星期一”
2. 输入一个整数month代表月份，根据月份输出对应的季节。

春季：3、4、5

夏季：6、7、8

秋季：9、10、11

冬季：12、1、2

1. 输入一个整数score代表分数，根据分数输出等级（A-E）(用两种方式)

A：90~100

B：80~89

C：70~79

D：60~69

E：0~60

# 选择结构-switch

## 简单使用

* switch(表达式) { case 数值1: break; … default: break;}
* 举例

int a = 10;

switch (a) {

case 0:

printf("这是一个0");

break;

case 5:

printf("这是一个5");

break;

case 10:

printf("这是一个10");

break;

default:

printf("什么也不是");

break;

}

* 简述break的作用
* 在case中定义变量

## if和switch的对比

* 很多情况可以互换
* if用得比较多，比较灵活，switch只能某个单值

## 习题

用switch替代if实现if中的习题

# 循环结构-while

## 简单使用

* 连续printf(“俯卧撑”)20次;
* while(表达式) { }
* continue：输出5次休息一下
* break的使用：某次的时候停止循环

## 特点

只有条件成立才会执行循环体

## 习题

1. 提示用户输入一个正整数n，计算1+2+3+…+n的和
2. 提示用户输入一个正整数n，计算1-2+3-4+5-6+…+n的和

## 陷阱

while(条件);

# 循环结构-do while

* 特点：一定会执行一次循环体
* while和do while循环对比

int i=0;  
while(i<0){  
 i++;  
}

int i=0;  
do{  
 i++;  
} while(i<0);

# 循环结构-for

## 简单实用

* 运行过程
* 初始化语句等可以是多句（把printf放到for的括号中）

## for循环嵌套

好友列表1

好友1

好友2

好友列表2

好友1

好友2

好友列表3

好友1

好友2

## 陷阱

1. 死循环 for(;;);
2. for循环体内部变量的作用域
3. 作用域混淆

for (int i = 0; i<5; i++) int a = 10;

## while循环和for循环的比较

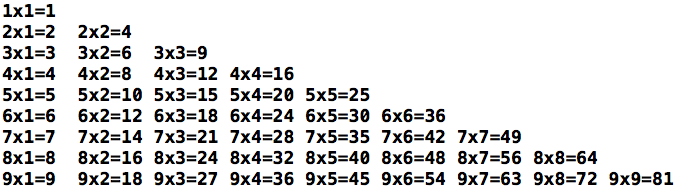
* 可以互换
* for循环的变量可以及时回收

## 习题

1. 提示用户输入一个正整数n，如果n5，就输出下列图形，其他n值以此类推



1. 输出九九乘法表



# break和continue

## break

* 使用场合：switch\循环结构
* 循环嵌套下的break

## continue

* 使用场合：循环结构
* 循环嵌套下的continue