黑马程序员------C 语言流程控制

一、 流程结构分类

i. 顺序结构：按照代码从上到小的顺序执行代码

ii. 选择结构：对给定条件进行判断，决定执行哪段代码

iii. 循环结构：在给定条件成立的情况下，反复执行某段代码

二、 选择结构：

a) if

i. if 第一种结构

```
if(条件)
{
  语句;
}
```

如果条件成立（为真）执行大括号里所有代码

如果条件不成立（为假）不执行 if 语句

ii. if 第二种结构

```
if(条件 1)
{
  语句;
}
else
{
}
```

如果条件 1 成立（为真）执行 if 后面大括号里所有代码

如果条件 1 不成立（为假）执行 else 后面大括号里的所有代码

iii. if 第二种结构

```
if(条件 1){
  语句 1;
}
else if (条件 2){
  语句 2;
} .....
else{
```

}

从上向下顺序验证，哪个条件成立，执行哪个条件后面的代码块，如果所有条件都不成立，执行 else 后面的代码块

1.同时只会有一个代码块会被执行，不能出现两个条件同时成立的情况(会报错)

2.当顺序验证条件时，条件成立后，下面的条件将不会再继续验证。

iv. if 第四种结构

if(条件 1)

语句 1；

语句 2；

条件成立执行 if 后的第一条语句，即语句 1，其它语句即语句 2，与 if 无关。

v. if 使用注意

1. 赋值符号“=”和等于号“==”不要混淆
2. if 语句后面不要加“;”分号，否则会导致死循环
3. 定义变量时，注意作用域的大小区分，若再 if 后面定义新的变量，要用大括号括住，否则会导致变量作用域不明确而报错。
4. 在 if 的条件里，若比较相等，把常量写左边，变量写右边，这样可以防止“=”和“==”混淆后编译器不报错，常量在左边如果混淆则会报错。

b) switch

i. 基本结构

```
switch(数值)
{
case 数值 1：
{
  语句 1；
  break；
}
case 数值 2：
{
```

```
  语句 2；
  break；
}
……
default：
{
  语句 3；
  break；
}
}
```

1. 顺序依次判断 switch 后的数值等于大括号里 case 后的哪个数值，等于哪个数值就执行哪个 case 后的语句。
2. break 退出整个 switch 函数
3. 如果一个条件成立的 case 后面没有 break，就会一直向下运行，执行所有 case(无论条件是否成立)后面的语句，直到遇到 break 为止。
4. 如果要在 case 后定义新的变量，要用{}括住，否则会导致变量作用域不明确。

ii. 使用注意

1. switch()；括号里面的内容可以使字符“+”“-”“*”“/”，英文字母等

c) switch 和 if 对比

1. 判断的时取值范围，不能转化为数值时，或者数值量大时，用 if 语句。
2. if 语句能完成的功能，switch 语句不一定能完成，
3. switch 语句能完成的功能，if 语句一定能完成相应的功能。

三、 循环结构

a) while

i. 基本结构

```
while(条件)
{
  语句；
}
```

如果条件成立，执行语句，再次判断条件是否成立，再执行语句，直到条件不成立，跳出循环。

ii. 怎么用

1. 先确定需要重复执行的操作
2. 再确定约束条件（即重复的次数）
3. 定义一个技术变量（记录重复执行的次数）

iii. continue 跳出本次循环进入下一循环

break 跳出循环结构体

iv. continue

1. 在特定条件下，跳出本次循环进入下次循环，条件自己设定

v. break

1. 在特定条件下，结束循环结构。

vi. 运行原理

1. 如果一开始条件不成立，永远不会执行循环体
2. 如果条件成立，就会执行一次循环体，执行完毕，再次判断条件是否成立，循环执行，直到条件不成立。

vii. while 使用注意

1. 死循环程序，用 ctrl + C 退出
2. while 语句后面不要跟分号，否则会引发死循环
3. 最简单的死循环：while(1);

b) do-while 循环

i. 结构：

```
do
{
   语句;
}while(条件);
```

无论条件是否成立，先执行一次循环语句，再判断条件是否成立

ii. while 和 do-while 的区别

1. 很多情况下：while 和 do-while 可以互换
2. 特点
    a) while 特点：如果一开始条件不成立，不执行循环体
    b) do-while 特点：无论一开始条件是否成立，都会先执行一次循环体。

c) for 循环

i. 结构:for(语句 1；条件；语句 2)

1. 语句 1：叫做初始化语句，也可以写在 for 循环前面，语句 2：叫做增量语句，是执行完循环体后再执行的语句
2. for 一开始就会执行一次语句 1，整个 for 循环只会执行一次
3. 判断条件是否成立，如果条件成立就会执行一次循环体，然后执行增量语句，再次判断是否成立，直到条件不成立，退出 for 循环
4. 用 for 循环可以做 while 循环做得所有功能
5. 逗号表达式：在两个语句之间用逗号链接，可以组成一个顺序运行的大语句

ii. for 循环语句使用注意

1. 在 for 循环后不要加分号，因为 for 循环后面不直接跟大括号时，默认把最近的一个语句作为循环体，分号相当于一条空语句
2. 在 for 循环体内定义新的变量，一定要定义在大括号内，否则会导致变量作用域不明确
3. 循环结构 for（；；）小括号里面的变量作用域，比循环体大括号内的变量作用域大
4. 最简单的 for 西循环 for（；；）；

iii. for 循环嵌套使用

1. for 循环嵌套 for 循环
    a) 如果输出的图形为几行几列，则第一个 for 循环控制行数，第二个 for 循环控制列数

四、 流程控制总结

a) if 选择结构：

i. 在同一时间内，只有一个语句会被执行，如果所有条件都不成立，则执行 else 后的语句

b) switch 选择结构：

i. 默认情况下，只有一个 case 后面的代码会被执行

ii. 如果一个 case 的条件成立且这个 case 后面没有 break，就会按顺序执行后面 case 中的语句，直到遇到 break 未知

iii. 如果要在 case 后面定义一个新的变量，一定要用大括号括住

iv. 在 switch 语句中不能缺少 break 和 default 语句

c) 循环结构

i. while

1. 性能中等
2. 如果一开始条件不成立，永远不会执行循环体

ii. do-while

1. 性能差
2. 无论条件是否成立，都会先执行一次循环体

iii. for

1. 性能优等
2. 能及时回收变量的内存空间

d) continue 和 break 使用注意

i. break 适用场合

1. switch 语句：退出整个 switch 语句
2. 循环结构：while，do-while，for：退出整个循环语句
3. 只对最近的循环体或语句起作用

ii. continue 使用场合

1. 循环结构：结束当前循环进入下一次循环体
2. 只对最近的循环体起作用