

Java程序设计基础

流程控制

流程控制主要内容

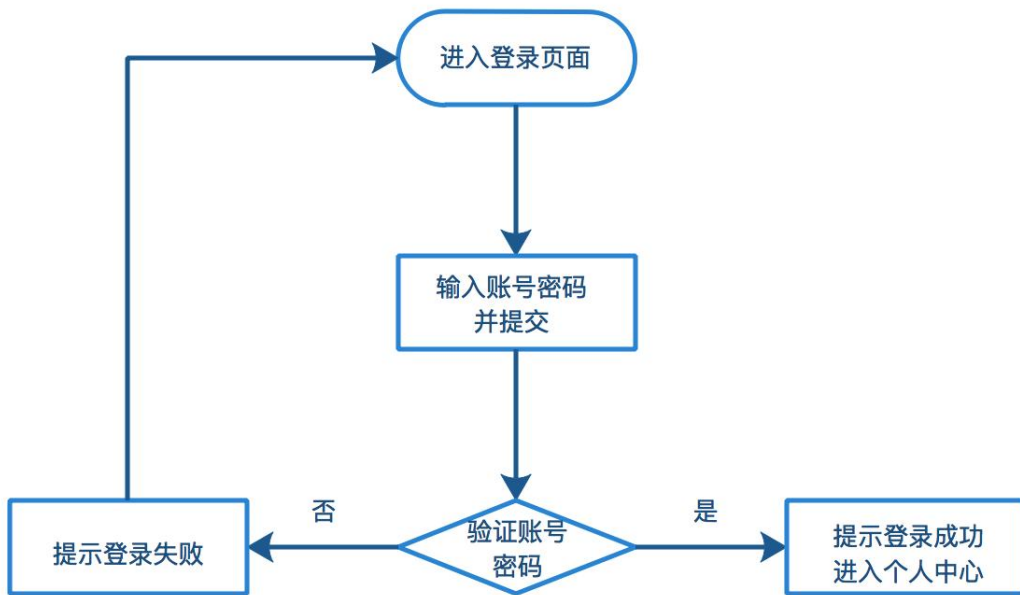
- 3.1 if语句
- 3.2 switch语句
- 3.3 while/do-while循环
- 3.4 for循环
- 3.5 跳转语句
- 3.6 debug

本章教学目标

- 能理解并根据业务需求编写if语句
- 能说出if/if-else/if-else if的区别和使用场景
- 能理解并根据业务需求编写switch语句
- 能说出if语句和switch语句的区别
- 能理解并根据业务需求编写while语句
- 能说出while/do-while的区别和使用场景
- 能熟悉使用for循环
- 能理解双重for循环的执行过程和步骤
- 能使用debug调试程序

为什么要用if语句

- 默认情况下，Java代码是由上向下执行所有语句，但在某些情况下，我们需要根据业务情况执行不同的代码，这个时候就需要用到if语句。
- 例如用户登录的时的流程图：

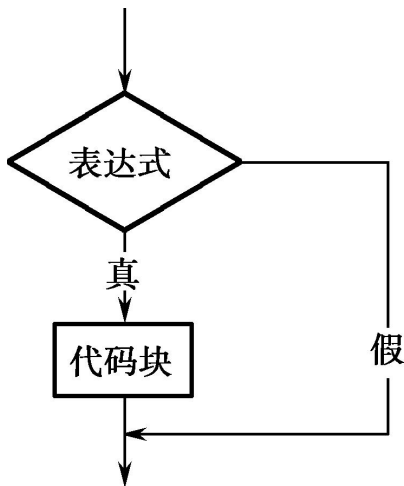


if的基本形式

- if语句共有三种形式。

- 第一种形式:

```
if(表达式){  
    代码块  
}
```



if的基本形式

- 代码块中只有一条语句时可简写为：

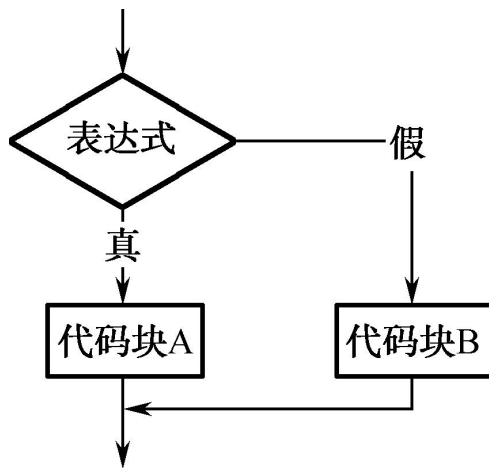
if(表达式)

一条语句

if的第二种形式

- 第二种形式:

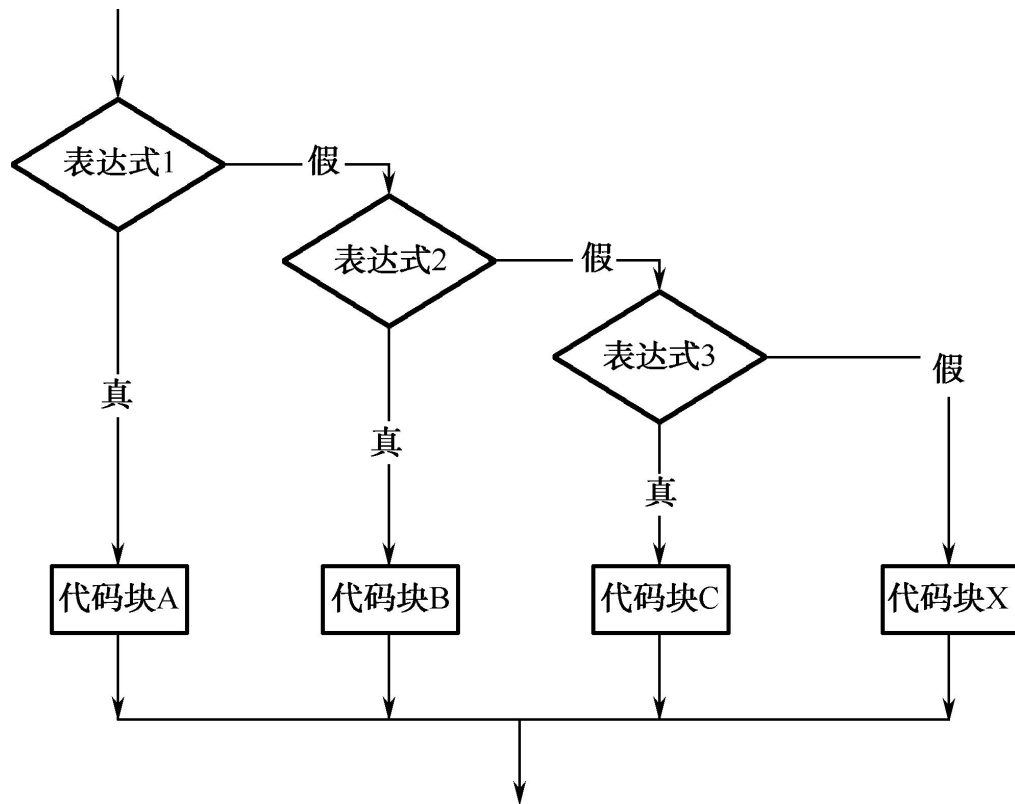
```
if(表达式){  
    代码块A  
}  
else{  
    代码块B  
}
```



if的第三种形式

- 第三种形式:

```
if(表达式 1){  
    代码块 A  
}else if(表达式 2){  
    代码块 B  
}else if(表达式 3){  
    代码块 C  
...  
}else{  
    代码块 X  
}
```



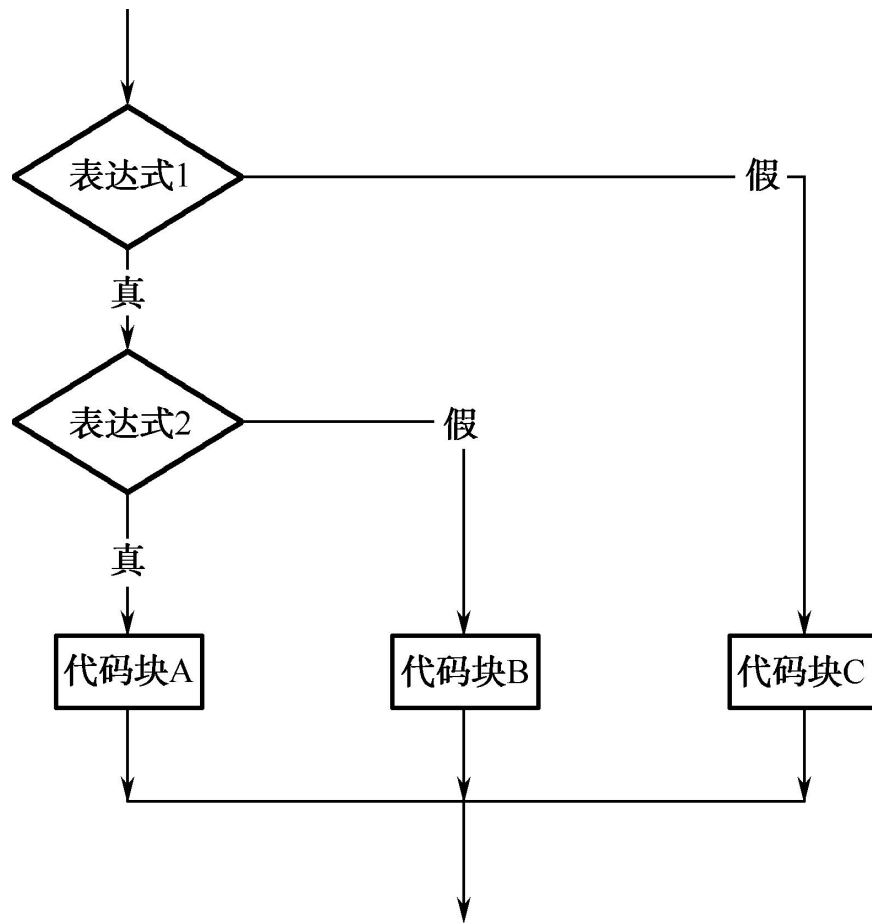
if的第三种形式

- 成绩等级查询:
- >100 : 输入有误
- 85-100: 优秀
- 70-84: 良好
- 60-69: 合格
- <60 : 不合格

嵌套的if语句

- 形式如下：

```
if(表达式1){  
    if(表达式2){  
        代码块A  
    }else{  
        代码块B  
    }  
}else{  
    代码块C  
}
```



switch语句

- switch 语句的语法形式如下:

```
switch(表达式){  
    case 常量1:  
        代码块A;  
        break;  
    case 常量2:  
        代码块B;  
        break;  
    ...  
    default:  
        代码块X;  
        break;  
}
```

- 表达式支持的类型: byte、short、int、char、String、枚举
- default和break语句不是必须的。

- 循环语句可以轻松高效的完成有规律或重复性的功能。
- 例如输出1-100、计算1-100的总和、输出如下图形或内容:

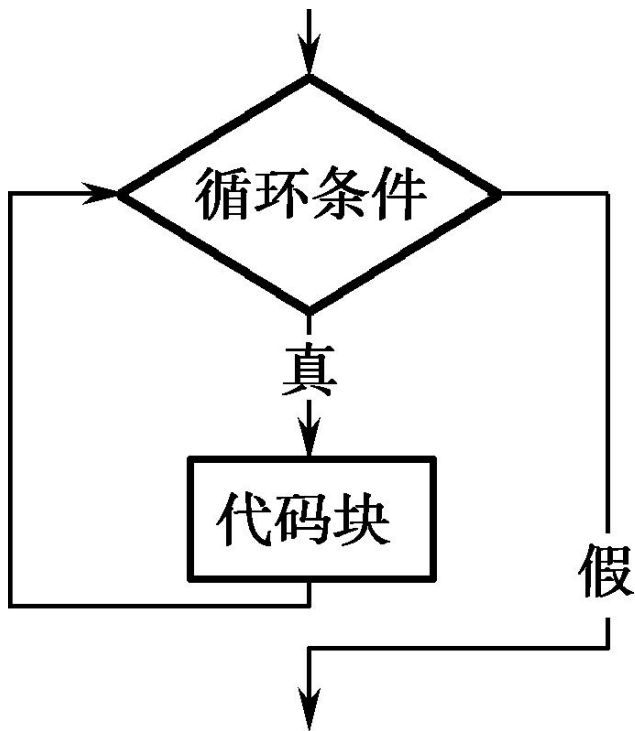
!

[illegible]

while语句

- while 循环的语法形式如下：

```
while(循环条件){  
    循环代码块  
}
```



do-while语句

- do-while 循环的语法形式如下：

```
do{
```

```
    循环代码块
```

```
}while(循环条件);
```

- do-while中循环代码块一定会执行至少一次

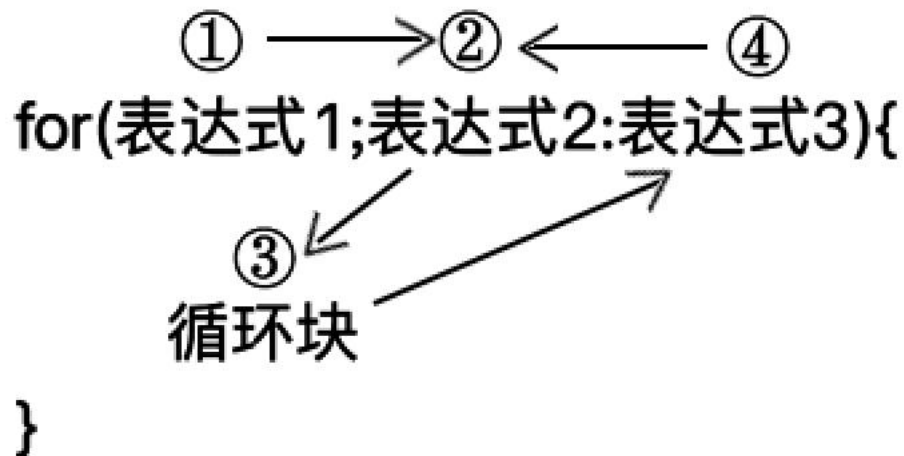
- for循环的语法形式如下：

```
for(表达式1;表达式2;表达式3){  
    循环代码块  
}
```

- for 循环的重点在于其 3 个表达式，其中：
- 表达式 1 通常是赋值语句，一般是循环语句的初始部分，为循环参数赋初值，表达式 1 可以省略。
- 表达式 2 通常是条件语句，即循环条件，当该条件满足时，进入循环，不满足则跳出循环。表达式 2 也可以省略，但省略后就没有了循环判断条件，也就形成了死循环。
- 表达式 3 通常也是赋值语句，属于循环结构的迭代部分，当一次循环代码块执行完毕以后，程序执行表达式 3，然后再去判断表达式 2 的循环条件是否满足。表达式 3 通常用来更改循环参数的值。表达式 3 也可以省略，如果省略，通常需要在循环代码块中添加修改循环参数的语句。

for循环

- for循环执行流程如下图：



嵌套的循环语句

- for循环的语法形式如下：

```
for(表达式1;表达式2;表达式3){  
    for(表达式4;表达式5;表达式6){  
        循环代码块  
    }  
}
```

- 嵌套的循环语句在外层for循环每一次循环都会将内层for全部循环一次。

- 在介绍 switch 语句的时候，首次接触了 break 语句，其作用是跳出 switch 代码块，执行 switch 语句后面的代码。在介绍双重 for 循环时，用到了 continue 语句，continue 语句的主要作用为跳出当次循环，继续执行下一次循环。其中 break、continue 以及后面要学的 return 语句，都是让程序从一部分跳转到另一部分，习惯上都称为跳转语句。
- 在循环体内，break 语句和 continue 语句的区别在于：使用 break 语句是跳出循环进而执行循环之后的语句（即结束当前的循环代码），而 continue 语句是跳过本次循环继续执行下一次循环（即跳过本次循环）。在企业面试的时候，这个问题经常被问到，务必掌握。

debug

- debug不是自动找出程序的bug，而是用于查看程序执行的过程。