## 1 if条件语句

### 一、案例描述

1. 考核知识点

编号：029002015

名称：if条件语句

1. 练习目标

* 熟悉if条件语句的语法格式和执行流程
* 掌握if条件语句的使用

1. 需求分析

if…else if…else语句用于对多个条件进行判断，进行多种不同的处理。为了让初学者掌握if…else if…else语句的用法，本案例将使用if…else if…else语句实现判断某月是哪个季节的功能。

1. 设计思路（实现原理）

1）编写一个类Example01。

2）定义一个变量month用于存储月份。

3）用if条件语句，判断这个月份在哪一个季节并输出结果，如：春季是3、4、5月，夏季是6、7、8月，秋季是9、10、11月，冬季是12、1、2月。

4）由于一年只有12个月，所以要过滤除1-12以外的月份值。

### 二、案例实现

定义Example01类，其代码如下：

class Example01

{

public static void main(String[] args)

{

int month = 3;

if(month == 3 || month == 4 || month == 5)

{

System.out.println(month+"月是春季");

}

else if(month == 6 || month == 7 || month ==8)

{

System.out.println(month+"月是夏季");

}

else if(month == 9 || month == 10 || month ==11)

{

System.out.println(month+"月是秋季");

}

else if(month == 1 || month == 2 || month ==12)

{

System.out.println(month+"月是冬季");

}

else

{

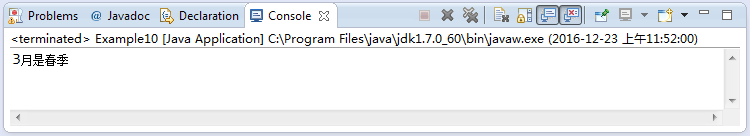
System.out.println("没有这个月份");

}

}

}

运行结果如图2-11所示。



1. 运行结果

### 三、案例总结

1、if…else if…else语句用于对多个条件进行判断，进行多种不同的处理。if…else if…else语句具体语法格式如下：

if (判断条件1) {

执行语句1

} else if (判断条件2) {

执行语句2

}

...

else if (判断条件n) {

执行语句n

} else {

执行语句n+1

}

if…else if…else语句的执行流程如图2-12所示。



1. 流程图

2、因为一年只有12个月，所以当month>12时是不符合逻辑的数据，这时会打印“没有这个月份”。在编程过程中，经常要考虑程序在实际环境下是否符合逻辑需求。

3、“||”表示或操作，当运算符两边的操作数任何一边的值为true时，其结果为true，当两边的值都为false时，其结果才为false。所以只要符合三个条件中的一个就会被判断为这个季节。

**2** **switch条件语句**

**一、案例描述**

1. 考核知识点

编号：029002016

名称：switch条件语句

1. 练习目标

* 了解switch条件语句的特点
* 熟悉switch条件语句的语法格式

1. 需求分析

switch 条件语句也是一种很常用的选择语句，和if条件语句不同，它只能针对某个表达式的值作出判断，从而决定程序执行哪一段代码。为了让初学者熟悉switch 条件语句，本案例将使用switch 条件语句实现判断选择题对错的功能。

1. 设计思路（实现原理）

1）编写一个类Example02。

2）定义一个变量answer用于存储答案选项。

3）用switch条件语句判断如果变量answer为B则正确，其它则错误。

4）如果变量answer不等于ABCD四个选择则打印“不存在此选项”。

**二、案例实现**

定义Example02类，其代码如下：

class Example02

{

public static void main(String[] args)

{

char answer = 'B';

switch(answer)

{

case 'A':

System.out.println("A错误");

break;

case 'B':

System.out.println("B正确");

break;

case 'C':

System.out.println("C错误");

break;

case 'D':

System.out.println("D错误");

break;

default:

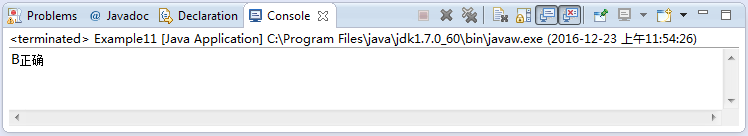
System.out.println("不存在此选项");

}

}

}

运行结果如图2-13所示。



1. 运行结果

**三、案例总结**

1、switch语句的执行流程是：根据表达式的值查找与其匹配的项,如果有匹配项,则执行该匹配项下面的语句。如果没有匹配项，就会执行default后面的语句。

2、switch关键字后括号内的表达式可以是byte,short,int,char四种类型；在JDK5以后表达式可以是枚举；在.JDK7以后表达式可以是字符串。

3、switch语句只适合判断若干值的情况，不适合判断范围。

4、if和switch都可以判断值的情况，这时应该使用switch语句，因为switch的效率比if语句高。

**3** **while循环语句**

**一、案例描述**

1. 考核知识点

编号：029002017

名称：while循环语句

1. 练习目标

* 掌握while循环语句的语法格式和使用
* 了解while循环语句的特点

1. 需求分析

while语句会反复地进行条件判断，只要条件成立，{}内的执行语句就会执行，直到条件不成立，while循环才会结束。为了让初学者熟悉while语句，本案例将使用while语句实现统计1-10以内奇数的功能。

1. 设计思路（实现原理）

1）编写一个类Example03。

2）定义一个变量x为1，用while语句循环条件为x<=10。

3）在while循环中，使用表达式x%2判断是否为奇数，如果为奇数，则变量count的值加1。

**二、案例实现**

定义Example03类，其代码如下：

class Example03

{

public static void main(String[] args)

{

int x = 1;

int count;

count = 0;

while(x<=10)

{

if(x%2==1)

{

count++;

System.out.println(x);

}

x++;

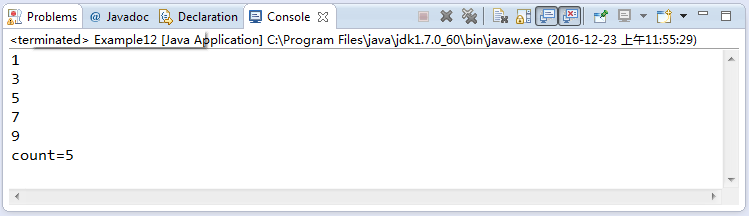
}

System.out.println("count="+count);

}

}

运行结果如图2-14所示。

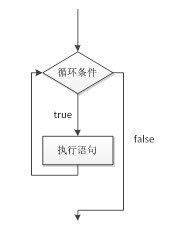


1. 运行结果

**三、案例总结**

1、while循环语句和条件判断语句有些相似，都是根据条件判断来决定是否执行大括号内的执行语句。区别在于，while语句会反复地进行条件判断，只要条件成立，{}内的执行语句就会执行，直到条件不成立，while循环结束。

2、while循环的执行流程如图2-15所示。



1. 流程图

**4** **for循环语句**

**一、案例描述**

1. 考核知识点

编号：029002018

名称：for循环语句

1. 练习目标

* 了解for循环语句的特点
* 掌握for循环语句的语法格式和使用

1. 需求分析

for循环语句是最常用的循环语句，一般用在循环次数已知的情况下。为了让初学者熟悉for循环语句，本案例将使用for循环语句计算1+2+3+4+5...+100表达式的结果

1. 设计思路（实现原理）

1）编写一个类Example04。

2）在该类中定义一个变量sum来存储每次循环整数相加的和。

3）在for循环语句中定义变量x,循环条件为x<=100。

**二、案例实现**

定义Example04类，代码如下所示：

class Example04

{

public static void main(String[] args)

{

int sum = 0;

for(int x = 1 ; x <= 100 ; x++)

{

sum = sum + x;

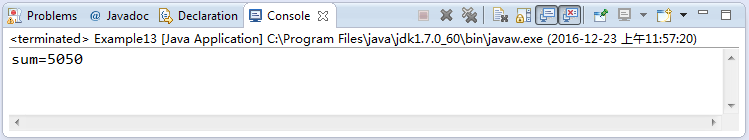
}

System.out.println("sum=" +sum);

}

}

程序编译时发生了错误，结果如图2-16所示。



1. 编译结果

**三、案例总结**

1、在for循环中，for关键字后面()中包括了三部分内容：初始化表达式、循环条件和操作表达式，它们之间用“;”分隔，{}中的执行语句为循环体。

2、通过序号来具体分析for循环的执行流程。具体如下：

for（① ; ② ; ③）{

④

}

第一步，执行①

第二步，执行②，如果判断结果为true，执行第三步，如果判断结果为false，执行第五步

第三步，执行④

第四步，执行③，然后重复执行第二步

第五步，退出循环

**5 一维数组的定义**

**一、案例描述**

1. 考核知识点

编号：029002025

名称：数组的定义

1. 练习目标

* 掌握一维数组的定义和使用

1. 需求分析

数组是指一组数据的集合，数组中的每个数据被称作元素。在数组中可以存放任意类型的元素，但同一个数组里存放的元素类型必须一致。请定义一个数组，将序列{0,1,2,3,4}赋值给数组，并写一个方法用来访问数据的每一个元素。

1. 设计思路（实现原理）
2. 定义一个数组，在初始化数组时有一种方式叫做静态初始化，就是在定义数组的同时就为数组的每个元素赋值。可以在定义数组时将序列{0,1,2,3,4}赋值给数组。
3. 定义一个方法输出数组的元素，可以用for循环语句依次输出数组的元素，各元素用“，”间隔可以更清楚的输出元素。
4. 在最后一个元素输出时，后面就不需要加“，”了，所以循环到length-1时不打印“，”。

**二、案例实现**

定义Example05类，代码如下所示:

class Example05

{

public static void main(String[] args)

{

int[] arr = {0,1,2,3,4};

printArr(arr);

}

public static void printArr(int[] arr){

System.out.print("[");

for(int x = 0 ; x < arr.length ; x++){

if(x != arr.length - 1)

System.out.print(arr[x]+",");

else

System.out.print(arr[x]);

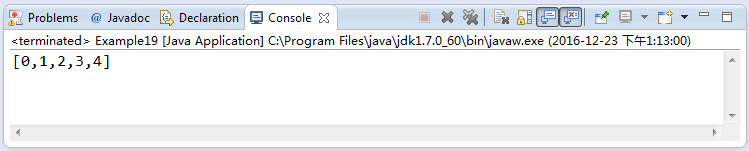
}

System.out.println("]");

}

}

运行结果如图2-22所示。



1. 运行结果

**三、案例总结**

1、每个数组的索引都有一个范围，即0~length-1。在访问数组的元素时，索引不能超出这个范围，否则程序会报数组越界异常ArrayIndexOutOfBoundsException。所谓异常指程序中出现的错误，它会报告出错的异常类型、出错的行号以及出错的原因。

2、在使用变量引用一个数组时，变量必须指向一个有效的数组对象，如果该变量的值为null，则意味着没有指向任何数组，此时通过该变量访问数组的元素会出现空指针异常NullPointerException。