第4章

循环结构

4.1 项目导引——猜数字游戏

猜数字游戏的游戏规则有很多种，抛开复杂的游戏规则，我们先设想一款规则比较简单的：由电脑随机产生一个1～100之间的数，然后我们去猜，我们有可能不是猜大了就是猜小了，如果在猜的过程中电脑又不给我些暗示，我想我们要在1～100之间的数中猜出到底电脑心里想的那个数，真的是太难了。还好电脑是比较通情达理的，如果我们猜大了，它会提示我们“你猜大了!”，这样我下次猜的时候就会往小的方向猜，如果我们猜小了，它会提示我们“你猜小了!”，我们再猜的时候可以往大的方向猜，这样一来就会逐步的缩小我们的猜测范围，最终如果在次数没有任何限制的情况下，我们一定会猜到电脑心里想的那个数。最后还可以根据我们猜的次数，电脑给出相应的信息，如果1次就猜对了，输出“快来看，上帝……”，如果猜的次数在2次到6次之间，则输出“这么快就猜对了，你很聪明啊!”，如果猜的次数超过6次，则输出“猜了半天才猜出来，小同志，尚须努力啊!”如果限制次数的话，到最后可能还没有猜出电脑心里所想，游戏就提示你“Game Over”了等等。限制次数的猜数字游戏过程如图4-1所示。

图4-1 猜数字游戏

那么，大家一定迫不及待的想要知道这款猜数字游戏是如何实现的吧？你能否很快的猜对

电脑的心思呢？其内部程序应该怎么写呢？

4.2　项目分析

如果仅仅用第 3 章的选择结构是否可以顺利的完成呢？我们先来分析一下。

由导引中描述，我们可以知道，猜数字游戏的过程实际上是将用户的猜测和电脑随机产生的那个数之间不断的进行比较的过程。首先，我们需要声明一个变量 randNumber 来保存电脑想要让我们猜的那个数，一个表示用户输入的变量 guessNumber，然后将 guessNumber 和 randNumber 进行比较，如果 guessNumber 大于 randNumber，则提示用户猜的太大了，如果 guessNumber 小于 randNumber，则提示用户猜的太小了，如果 guessNumber 等于 randNumber，则提示用户猜对了。我们可以使用选择结构中多分支条件语句来实现上述描述。核心代码如下所示。

```
System.out.println("请输入你的猜测!");
Scanner input=new Scanner(System.in);
guess=input.nextInt();
if (guess > randNumber){
    System.out.println("你猜得数太大了,继续猜吧!");
}else if (guess < randNumber){
    System.out.println("你猜得数太小了,继续猜吧!");
}else{
    System.out.println("恭喜你,猜对了!");
}
```

可上述过程只能运行一次，也就是说如果猜大了，电脑虽然会提示“你猜得数太大了，继续猜吧!”，但是程序接着马上就终止了，根本不给我们再次猜的机会。怎样让这个过程能够重复执行呢？显然把这段代码不断的进行复制的做法是不可取的。在 Java 语言中，我们通过***循环结构***来解决重复问题，并可以根据不同的情况选择不同的循环结构。通常情况下，如果事先可以确定循环次数，使用***for 循环***，如果可以断定循环至少执行一次，用 ***do while 循环***，如果事先不能确定循环次数并且只有满足一定的条件下才执行循环，则可以使用 ***while 循环***。当然在循环结构中还提供了两个重要的关键字 ***break*** 和 ***continue*** 。

4.3　技术准备

4.3.1　for 循环

循环语句的主要作用是解决重复问题，重复执行一段代码直到满足一定的条件为止。可以将循环分成 4 个部分：

(1)初始化：设置循环开始的初始值。

(2)循环体：重复执行的语句。

(3)迭代部分:使循环条件发生改变的部分。

(4)循环条件:判断是否继续循环的条件。

for 循环将循环结构的 4 个组成部分集合在一起,更加清晰。在事先能够确定循环次数的情况下,首选 for 循环结构。该循环的语法格式为:

```
for(表达式 1;表达式 2;表达式 3){
    循环执行的语句
}
```

每个表达式的含义如表 4-1 所示。

表 4-1　　for 循环中 3 个表达式的含义

表达式	形式	功能
表达式 1	赋值语句	设置循环开始的初始值
表达式 2	条件语句	循环条件
表达式 3	赋值语句,通常使用++或--运算符	使循环条件发生改变

for 循环中的 3 个表达式以及循环执行的语句使循环结构必须的 4 个部分完美的组合在了一起,非常简洁。

for 循环的执行过程流程图如图 4-2 所示。

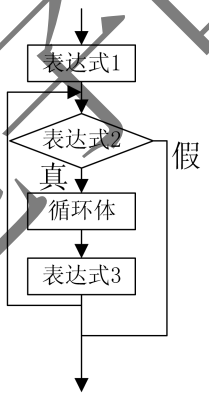

图 4-2　for 循环执行过程的流程图

【例 4-1】输出 10 遍“学习 Java 真开心”。

【程序实现】

```
1    //ForTest.java
2    public class ForTest {
3       public static void main(String[] args) {
4          //表达式 1 int i=1 完成给循环变量赋初值,表达式 1 只执行一次
5          //表达式 2 i<=10 表明循环条件,但次数没有达到 10 次时执行循环体
6          //表达式 3 i++使循环变量自增 1
7          for (int i = 1; i <= 10; i++) {
8             System.out.println("学习 Java 真开心");//重复执行的语句,即循环体
9       }
10    }
11 }
```

【程序说明】

● 第 7 行 for 语句，表达式 1：int i=1 完成给循环变量赋初值，表达式 1 只执行一次；表达式 2：i<=10 表明循环条件，次数没有达到 10 次时执行循环；i++使循环变量自增 1。

● 第 8 行 循环体 输出字符串“学习 Java 真开心”。

● 重复执行 10 次第 8 行的循环体语句，输出 10 行“学习 Java 真开心”

【运行结果】

```
学习 Java 真开心
学习 Java 真开心
学习 Java 真开心
学习 Java 真开心
学习 Java 真开心
学习 Java 真开心
学习 Java 真开心
学习 Java 真开心
学习 Java 真开心
学习 Java 真开心
```

4.3.2　while 循环

while 循环是当满足一定的条件时才执行循环操作，当一开始条件就不满足时，循环体有可能一遍都不会被执行。该循环的语法格式为：

```
while(循环条件){
  循环体
}
```

其语义为：首先判断循环条件是否成立，如果成立则执行循环体，否则退出循环；执行完循环体后，回来再次判断循环条件，决定继续执行循环或退出循环。

while 循环的执行过程流程图如图 4-3 所示。

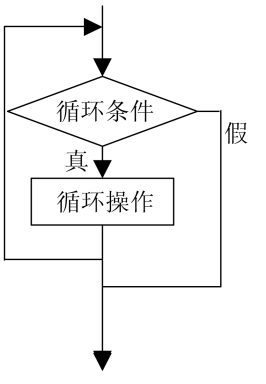

图 4-3　while 循环的执行过程流程图

使用 while 循环实现【例 4-1】。

```
1  //WhileTest.java
2  public classWhileTest {
3    public static void main(String[] args) {
4      int i=1;//循环变量赋初值
5      while(i<=100){//循环条件
6        System.out.println("学习 Java 真开心");//循环语句
7        i++;//改变循环变量的值
8      }
9    }
10 }
```

【例 4-2】编程计算整数 1～100 的和。

【问题分析】

这是典型的“累加和”问题，读者可以想一下如果让你口算 1＋2＋……＋100 的和，当然这里不考虑算法技巧，就是一个个的加完看结果，我们只能怎么做呢？读者可以想象一下我们心里一开始是没有数的，当老师报了数字 1，心里的那个数就变成了 1，当老师报下一个要加的数字是 2，心里的数就变成了 3，老师再报 4，心里的数又变成了 7，……当老师报到 100 的时候，读者就可以直接得出结果了。

通过上述分析，我们可以发现这样一个规律，心里的那个数在不断的变化，我们可以用一个变量 sum 来表示心里的那个数，用 i 表示下一个要加的数，我们总是在做重复的两步工作：sum＝sum＋i;i＋＋；显然这又是典型的重复问题，要用循环结构来解决。循环什么时候结束呢？i 为 100 的时候。

【程序实现】

```
1  //WhileTest.java
2  public class WhileTest {
3      public static void main(String[] args) {
4          int i = 1;   // 定义循环变量并赋初值
5          int sum = 0;  // 累加和清 0
6          while (i <= 100) { //i<=100 表示循环条件
7              sum = sum + i;  // 循环变量累加到 sum 中
8              i = i + 1;  // 改变循环变量 i 的值
9          }
10         System.out.println("sum="+sum);
11     }
12 }
```

【程序说明】

- 第 4 行 定义循环变量并赋初值。
- 第 5 行 累加和清 0。
- 第 6 行 为循环条件，i<＝100 时执行循环体。
- 第 7～第 8 行为循环体，将循环变量累加到 sum 中并改变循环变量 i 的值。

【运行结果】

```
sum＝5050
```

4.4　项目实施

学习了循环结构，我们就可以很容易的编写本章开头提出的猜数字游戏。

【问题分析】

通过前面的项目分析，我们知道猜数字游戏的过程是个重复的过程，需要使用循环结构。究竟该选择 for 循环结构、while 循环结构还是 do-while 循环结构呢？让我们再来回顾一下这三种结构的特点：for 循环适合解决一开始就能确定循环次数的情境，while 循环当满足一定条件时就会执行循环操作，do-while 循环和 while 循环的区别是至少会执行一遍循环操作。玩猜数字游戏再聪明的人也至少需要猜一次，循环操作至少会执行一遍，所以我们优先选用 do-while 循环。

循环条件应该是什么呢？大家想想游戏什么时候才算结束呢？那当然是用户猜对了。如何知道用户是否猜对了呢？那当然是用户猜的数和随机产生的那个数相等的时候就不需要再猜了。所以循环条件应该是用户没有猜对的时候，即用户猜的数和随机产生的那个数如果不相等，用户就需要继续猜。

解决这个问题大致需要以下 4 个步骤：

(1)随机产生一个 1～100 之间的随机数并声明一个变量 randNumber 保存这个数；声明一个变量 count 初始值为 0，记录用户猜的次数。

(2)声明一个变量 guess 保存用户输入的猜测并与 randNumber 进行比较，如果 guess 小于 randNumber，提示用户“你猜得数太小了，继续猜吧！”；如果 guess 大于 randNumber，提示用户“你猜得数太大了，继续猜吧！”；使 count 的值增 1；

(3)如果 guess 不等于 randNumber，则回到(2)继续；否则执行(4)。

(4)根据用户猜测的次数及 count 的值，输出相应的信息。

程序的流程图如图 4-5 所示。

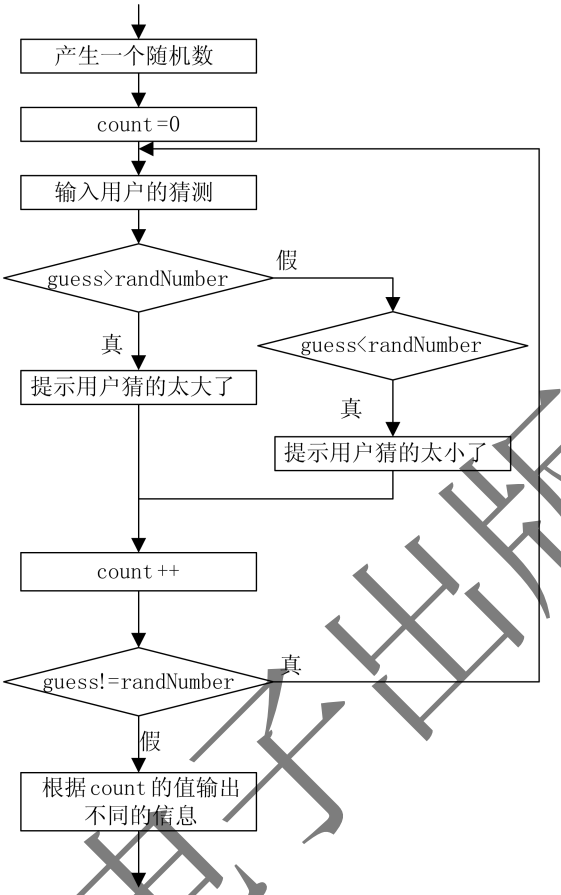

图 4-5　猜数字游戏的执行过程流程图

【程序实现】

```
1  import java.util.Random;
2  import java.util.Scanner;
3  public class GuessNumberGame {
4      public static void main(String[] args) {
5          int randNumber;// //定义存放产生随机数的变量
6          int guess; // 存放用户所猜得数
7          int count = 0;// 统计用户所猜测的次数
8          // 产生随机数
9          Random rand = new Random();
10         randNumber = rand.nextInt(100)+1;
11         // 输入用户所猜的数,直到猜对为止,并统计用户所猜得次数
12         do{
13             System.out.println("请输入你猜得数:");
14             Scanner input = new Scanner(System.in);
15             guess = input.nextInt();
16             if(guess > randNumber)
```

```
17                System.out.println("你猜得数太大了,继续猜吧!");
18            else if (guess < randNumber)
19                System.out.println("你猜得数太小了,继续猜吧!");
20                count++;
21        } while (guess ! = randNumber);
22    // 根据次数打印出不同的信息
23        switch (count) {
24            case 1:
25                System.out.println("快来看上帝……");
26                break;
27            case 2:
28            case 3:
29            case 4:
30            case 5:
31            case 6:
32                System.out.println("这么快就猜对了,你很 smart 啊!");
33                break;
34            default:
35                System.out.println("猜了半天才猜出来,小同志,尚须努力啊!");
36                break;
37            }
38        }
39    }
```

【程序说明】

● 第 1 行使用 import 语句引入了 java.util 包中的 Random 对象,以便用于产生随机数,第 2 行引入了 Scanner 对象,以便用于键盘输入。关于 import 语句、Java 包、对象等知识点我们将在第 6、7 章面向对象篇幅中讲解。此处读者记住需要产生随机数时在程序的第一行写入这行语句即可。

● 第 9 行创建一个 Random 类型的对象 rand,用于产生随机数。第 10 行调用方法 nextInt(100)会产生区间为[0,100)的随机数,如果要产生区间为[1,100]的随机数,则需要再加 1。读者在这里只需记住这两行是为了产生 1～100 之间的随机数即可。如果你想产生[1,10]之间的随机数只需要将 nextInt(100)中的 100 改为 10 即可。

● 第 12～21 行是猜数字游戏的核心。根据前面所进行的问题分析,在本程序中我们使用了 do-while 循环结构。第 13 行是提示信息,如果没用该行语句,程序运行时用户将不知道应该做什么事情,在实际的开发中,和用户的交互性这点是需要重点考虑的。因此一般在输入语句之前都会加入一条输出语句用于给用户提示。第 14 行～15 行语句是接收用户从键盘中的输入保存到声明的变量 guess 中。第 16～19 行是比较 guess 和 randNumber 变量的大小关系,根据比较结果提示用户是猜大了还是猜小了。第 20 行使计数器加 1,因为只要上述判断过程结束,就代表用户已经猜了一次。

● 第 23～27 行采用 switch 结构对 count 的值进行判断,如果 count 为 1,说明用户猜了一次就猜对了,显示“快来看上帝……”,如果 count 的值在区间[2,6]中,则显示“这么快就猜对

了,你很 smart 啊!",如果超过 6 次,则显示"猜了半天才猜出来,小同志,尚须努力啊!"。

注意

注意计数器 count 的初始化放在循环的外面,否则每次进入循环都会重新将 count 的值重新清 0,还要注意 21 行 do-while 循环末尾的分号千万不能省略。

【运行结果】

```
请输入你猜得数:
50
你猜得数太大了,继续猜吧!
请输入你猜得数:
40
你猜得数太大了,继续猜吧!
请输入你猜得数:
30
你猜得数太大了,继续猜吧!
请输入你猜得数:
20
你猜得数太大了,继续猜吧!
请输入你猜得数:
10
你猜得数太小了,继续猜吧!
请输入你猜得数:
15
你猜得数太大了,继续猜吧!
请输入你猜得数:
14
你猜得数太大了,继续猜吧!
请输入你猜得数:
13
猜了半天才猜出来,小同志,尚须努力啊!
```

这样,读者就可以玩猜数字游戏了,看看你能很快猜出电脑的心思吗? 由于是让电脑随机产生一个数让大家去猜,所以每次玩电脑产生的都不一定是同一个数啊。当我们学到 Java 中的图形用户界面设计时,就能够实现图形化的猜数字游戏了。

4.5 技术拓展

4.5.1 循环语句的嵌套

我们前面已经学习了各种不同的循环结构,若一个循环结构的循环体内包含了另一个循环语句,则构成了循环语句的嵌套,又称为多重循环。

while、do-while、for 这三种循环语句均可互相嵌套,具体语法格式如下:

(1)for 循环的嵌套。

```
for(表达式 1 ; 表达式 2;表达式 3 ){
```

```
        ……
        for(表达式 1;表达式 2;表达式 3 ){
                ……
        }
        ……
}
```

(2)while 循环的嵌套。

```
while(循环条件){
        ……
        while(循环条件 ){
                ……
        }
        ……
}
```

(3)do-while 循环的嵌套。

```
do{
        ……
        do{
        ……
        }while(  ) ;
        ……
}while(  ) ;
```

(4)三种循环结构的互相嵌套。

```
while(循环条件 ){
        ……
        for(表达式 1 ;表达式 2 ;表达式 3 ){
                ……
                do{
                        ……
                }while(循环条件) ;
                ……
        }
        ……
}
```

循环嵌套的特点是外层循环每执行 1 次,内循环就会完整的执行一遍。当所设计的程序具有这个特点时,一定会使用循环嵌套。

4.5.2　for-each 循环

for-each 循环是 JDK1.5 加入的。for-each 为开发人员提供了极大的方便。for-each 语句是 for 语句的特殊简化版本,但是 foreach 语句并不能完全取代 for 语句,然而,任何的 for-each 语句都可以改写为 for 语句版本。for-each 并不是一个关键字,从英文字面意思理解 for-each 也

就是“for 每一个”的意思，实际上也就是这个意思。

foreach 的语句的一般形式如下所示：

```
foreach(元素类型 t 元素变量 x ：遍历对象 obj){   //遍历的对象通常是数组或者集合
  循环体
}
```

【例 4-3】将数组中的每个元素输出出来。

```
1    public class ForEachTest {
2        public static void main(String[] args) {
3            /*定义了一个数组，数组的讲解见第五章，在这里大家可以想象成在内存中开辟了
             一片连续的存储空间，依次的放入 4,3,5,8,7*/。
4            int[] num={4,3,5,8,7};
5            for(int n:num){//依次取出 num 中的每个元素赋值给 n，并且将其输出出来
6                System.out.println(n);
7            }
8        }
9    }
```

4.6 本章小结

本章介绍了解决重复问题的三种结构，和循环相关的两个关键字 break 和 continue，并详细的分析了循环嵌套的执行特点。学习本章后，读者应该掌握以下内容：

- 解决重复问题的三种结构。
- for 循环的语法与使用。
- while 循环的语法与使用。
- do-while 循环的语法与使用。
- 三种循环各自应用的场合。
- break 和 continue 关键字的使用。
- 循环嵌套的特点和使用。

4.7 强化练习

一、判断题

1. break 语句可以用在循环和 switch 语句中。（　　）
2. continue 语句用在循环结构中表示继续执行下一次循环。（　　）
3. while 循环至少执行一遍。（　　）
4. 嵌套循环的特点是外循环 1 次，内循环 1 遍。（　　）
5. 嵌套循环的次数为外循环的次数加上内循环的执行次数。（　　）

二、选择题

1. 运行下面的程序将输出(　)次“我爱中国”

```
public classChina{
public static void main(String[] args){
    int i=1;
    do{
        System.out.println("我爱中国");
    }while(i<5);
        }
}
```

A. 4　　B. 5　　C. 0　　D. 死循环

2. 阅读下面的程序片断,输出结果是(　)

```
int  a=0;
while(a<5){
    switch(a){
        case 0:
        case 3:  a=a+2;
        case 1:
        case 2:  a=a+3;
        default:  a=a+5;
    }
}
System.out.println(a);
```

A. 0　　B. 5　　C. 10　　D. 其他

3. 阅读下列代码,如果输入的数字是 6,正确的运行结果是(　)

```
import  java.util.*;
public class Test{
     public static void main(String[] args){
        Scanner input=new Scanner(System.in);
        System.out.print("请输入 1 个 1-10 之间的数");
        int number =input.nextInt();
        for(int i=1;i<=10;i++){
            if((i+number)>10){
                break;
            }
        System.out.print(i+" ");
        }
    }

}
```

A. 1 2 3 4 5 6　　B. 7 8 9 10　　C. 1 2 3 4　　D. 5 6 7 8

4. 阅读下面程序中,while 循环的循环次数是(　)

```
public static void main(String[ ] args){
                int i = 0;
                while(i<10){
```

```
                    if(i<1){
                        continue;
                    }
                    if(i==5) {
                        break;
                    }
                    i++;
                }
}
```

A. 1　　B. 10　　C. 6　　D. 死循环

5. 下面程序的输出结果是多少(　　)

```
m=37;n=13;
while(m! =n)
{
    while(m>n)
    m=m-n;
  while(n>m)
    n-=m;
}
System.out.println(m);
```

A. m=13　　B. m=11　　C. m=1　　D. m=2

三、简答题

1. 三种循环结构各自应用的场合?

2. break 关键字的使用?

3. continue 关键字的使用?

4. 循环嵌套是什么意思,用于解决什么问题?

5. for-Each 语句的使用?

四、编程题

1. 改进猜数字游戏:用户玩了一次之后,如果还想继续玩,又会开始新一轮的猜数字游戏。

2. 编写一个程序,输出 2 到 500 之间的所有完数。所谓完数,是指一个整数等于该数所有因子之和。

提示:判断完数的方法:对于一个数 m,除该数本身外的所有因子都应在 1～m/2 之间。要取得 m 的因子之和,只要在 1～m/2 之间找到所有整除 m 的数,将其加起来即可。如果累加和与 m 本身相等,则表示 m 是一个完数。

3. 用 1 元 5 角钱人民币兑换 5 分、2 分和 1 分的硬币(每一种都要有)共 100 枚,问共有几种兑换方案? 每种方案兑换多少枚?

4. 一百元钱去买一百只鸡,公鸡每只 5 元,母鸡每只 3 元,小鸡 3 只一元,问公鸡、母鸡、小鸡各应买多少只?

5. 水仙花数是指个位、十位和百位三个数的立方和等于这个三位数本身的数,编写程序求出所有的水仙花数。