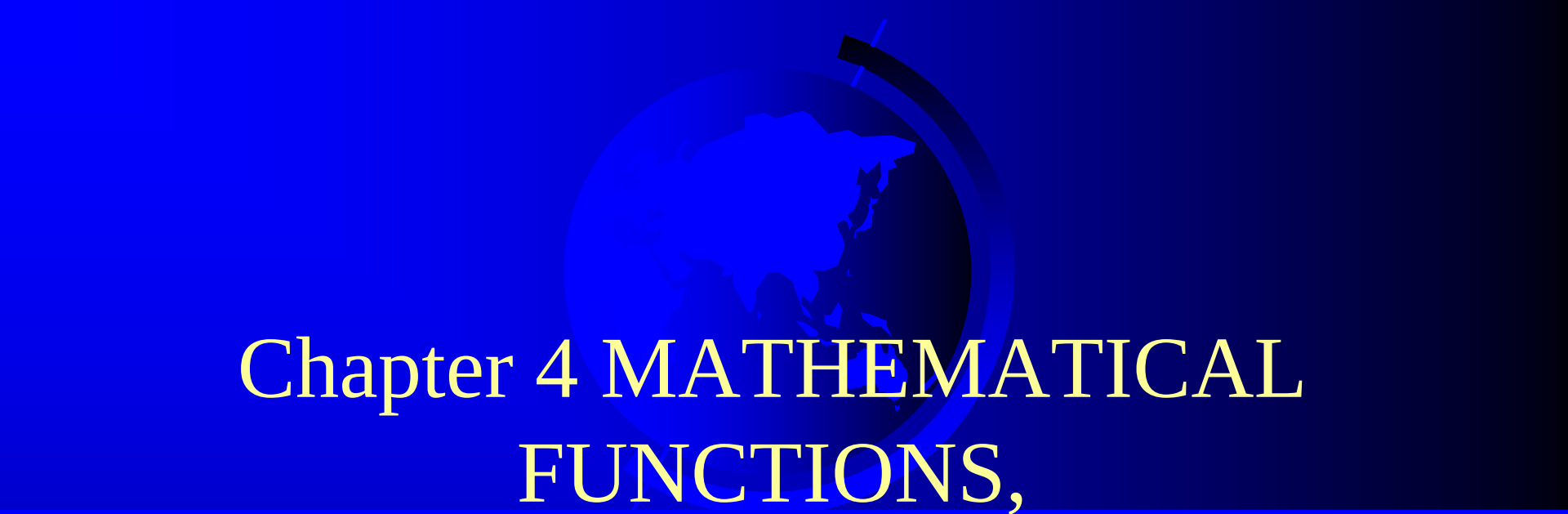


Chapter 4 MATHEMATICAL FUNCTIONS, CHARACTERS, AND STRINGS

Math 类

- 两个预定义常量:
  - Math.PI
  - Math.E
- 有很多数学方法:
  - 三角函数类
  - 指数函数类
  - 舍入函数
  - 最大，最小，绝对值以及随机函数

三角函数类

- sin(double a)
- cos(double a)
- tan(double a)
- acos(double a)
- asin(double a)
- atan(double a)

Radians

toRadians(90)

例子:

```
Math.sin(0) returns 0.0
Math.sin(Math.PI / 6)
  returns 0.5
Math.sin(Math.PI / 2)
  returns 1.0
Math.cos(0) returns 1.0
Math.cos(Math.PI / 6)
  returns 0.866
Math.cos(Math.PI / 2)
  returns 0
```

指数函数类

- `exp(double a)`
  Returns `e` raised to the power of `a`.
- `log(double a)`
  Returns the natural logarithm of `a`.
- `log10(double a)`
  Returns the 10-based logarithm of `a`.
- `pow(double a, double b)`
  Returns `a` raised to the power of `b`.
- `sqrt(double a)`
  Returns the square root of `a`.

例子：

```
Math.exp(1) returns 2.71
Math.log(2.71) returns 1.0
Math.pow(2, 3) returns 8.0
Math.pow(3, 2) returns 9.0
Math.pow(3.5, 2.5) returns
   22.91765
Math.sqrt(4) returns 2.0
Math.sqrt(10.5) returns 3.24
```

舍入函数类

- `double ceil(double x)`
  向上舍入到整数，注意结果依然为double
- `double floor(double x)`
  向下舍入到整数，注意结果依然为double
- `double rint(double x)`
  舍入到最近整数。如果两边一样近就取偶数
- `int round(float x)`
  四舍五入到整数
- `long round(double x)`
  四舍五入到整数

舍入函数的例子

```
Math.ceil(2.1) returns 3.0
Math.ceil(2.0) returns 2.0
Math.ceil(-2.0) returns -2.0
Math.ceil(-2.1) returns -2.0
Math.floor(2.1) returns 2.0
Math.floor(2.0) returns 2.0
Math.floor(-2.0) returns -2.0
Math.floor(-2.1) returns -3.0
Math.rint(2.1) returns 2.0
Math.rint(2.0) returns 2.0
Math.rint(-2.0) returns -2.0
Math.rint(-2.1) returns -2.0
Math.rint(2.5) returns 2.0
Math.rint(-2.5) returns -2.0
Math.round(2.6f) returns 3
Math.round(2.0) returns 2
Math.round(-2.0f) returns -2
Math.round(-2.6) returns -3
```

min, max, abs

- `max(a, b)` and `min(a, b)`

  Returns the maximum or minimum of two parameters.

- `abs(a)`

  Returns the absolute value of the parameter.

- `random()`

  Returns a random `double` value in the range [0.0, 1.0).

例子:

```
Math.max(2, 3) returns 3
Math.max(2.5, 3) returns
  3.0
Math.min(2.5, 3.6)
  returns 2.5
Math.abs(-2) returns 2
Math.abs(-2.1) returns
  2.1
```

random 方法

产生一个0.0到 1.0之间的随机数(0 <= Math.random() < 1.0)，类型是double。

例如:

```
(int)(Math.random() * 10)        ──→  Returns a random integer
                                      between 0 and 9.

50 + (int)(Math.random() * 50)   ──→  Returns a random integer
                                      between 50 and 99.
```

更一般地,

```
a + Math.random() * b   ──→  Returns a random number between
                             a and a + b, excluding a + b.
```

生成随机字符

- Java字符采用的是Unicode编码，取值从0-65535，所以只要随机生成一个该范围内的整数，转换类型之后就是字符了。
- 由于 0 <= Math.random() < 1.0, 所以可以按照下式生成随机字符：

```
(int)(Math.random() * (65535 + 1))
```

- 注意随机数取不到1，所以65535需要+1

随机生成小写字母

- 字母从 ‘a’, ‘b’, ‘c’, ..., 到 ‘z’ ，它们的Unicode值是递增的，所以随机小写字母可以这么生成：
  - (int)((int)'a' + Math.random() * ((int)'z' - (int)'a' + 1)
- 考虑到char类型在算术运算的时候会被自动转成int，所以上式可以简写成：
  - (char)('a' + Math.random() * ('z' - 'a' + 1))

生成某区间的随机字符

更一般地，要生成 ch1 和 ch2 之间的随机字符（含ch1和ch2，且 ch1 < ch2），可以按照下式生成：

– (char)(ch1 + Math.random() * (ch2 – ch1 + 1))

字符类型

4位16进制数

char letter = 'A'; (ASCII)

char numChar = '4'; (ASCII)

char letter = '\u0041'; (Unicode)

char numChar = '\u0034'; (Unicode)

自增自减运算符可以用在char类型上。例如，下面的代码将会显示一个b字母：

```
char ch = 'a';
System.out.println(++ch);
```

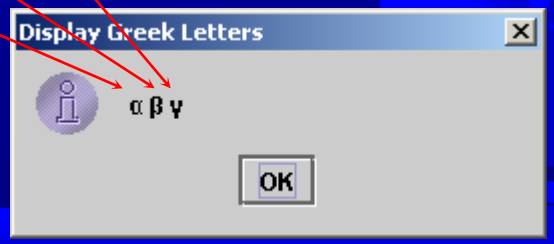
Unicode格式

Java字符采用*Unicode*格式，它是16位的字符编码方案，可以表示世界上的大部分常用文字，包括中文。

Unicode是双字节的，以\u开头，后面跟上4个16进制数字，也就是从‘\u0000’到‘\uFFFF’。所以Unicode可以表示65535 + 1个字符。

注意一定要写满4个数字，所以‘\u0’, ‘\u00’, ‘\u000’的写法都是错误的。

Unicode \u03b1 \u03b2 \u03b3表示3个希腊字母



转义字符

Description	*Escape Sequence*	*Unicode*
退格	`\b`	`\u0008`
Tab	`\t`	`\u0009`
换行	`\n`	`\u000A`
回车	`\r`	`\u000D`
反斜杠	`\\`	`\u005C`
单引号	`\'`	`\u0027`
双引号	`\"`	`\u0022`

字符和数值类型的相互转换

```
int i = 'a'; // Same as int i = (int)'a';

char c = 97; // Same as char c = (char)97;
```

字符串类型

char只能用来表示单个字符，如果需要表示一串字符，可以使用String。例如：

```
String message = "Welcome to Java";
```

String其实是Java预先定义的一个类，和 System 或者 JOptionPane 类一样，String 类不是基本类型。它是一种参考类型（ *reference type*）。

关于参考类型的进一步描述会在后面的章节展开，这里你可以简单把String当作一种类型来使用。

字符连接

```
// 三个字符串连接在一起
String message = "Welcome " + "to " + "Java";

// 字符串后面连上数字
String s = “Chapter” + 2; // s变为Chapter2

// 字符串后面连上字母
String s1 = “Supplement” + ‘B’; // s1变为SupplementB
```

字符串函数

- 字符串有许多成员函数可用，例如：

Method	*Description*
`length()`	Returns the number of characters in this string.
`charAt(index)`	Returns the character at the specified index from this string.
`concat(s1)`	Returns a new string that concatenates this string with string s1.
`toUpperCase()`	Returns a new string with all letters in uppercase.
`toLowerCase()`	Returns a new string with all letters in lowercase
`trim()`	Returns a new string with whitespace characters trimmed on both sides.

Method	*Description*
`equals(s1)`	Returns true if this string is equal to string `s1`.
`equalsIgnoreCase(s1)`	Returns true if this string is equal to string `s1`; it is case insensitive.
`compareTo(s1)`	Returns an integer greater than 0, equal to 0, or less than 0 to indicate whether this string is greater than, equal to, or less than `s1`.
`compareToIgnoreCase(s1)`	Same as `compareTo` except that the comparison is case insensitive.
`startsWith(prefix)`	Returns true if this string starts with the specified prefix.
`endsWith(suffix)`	Returns true if this string ends with the specified suffix.
`contains(s1)`	Returns true if s1 is a substring in this string.

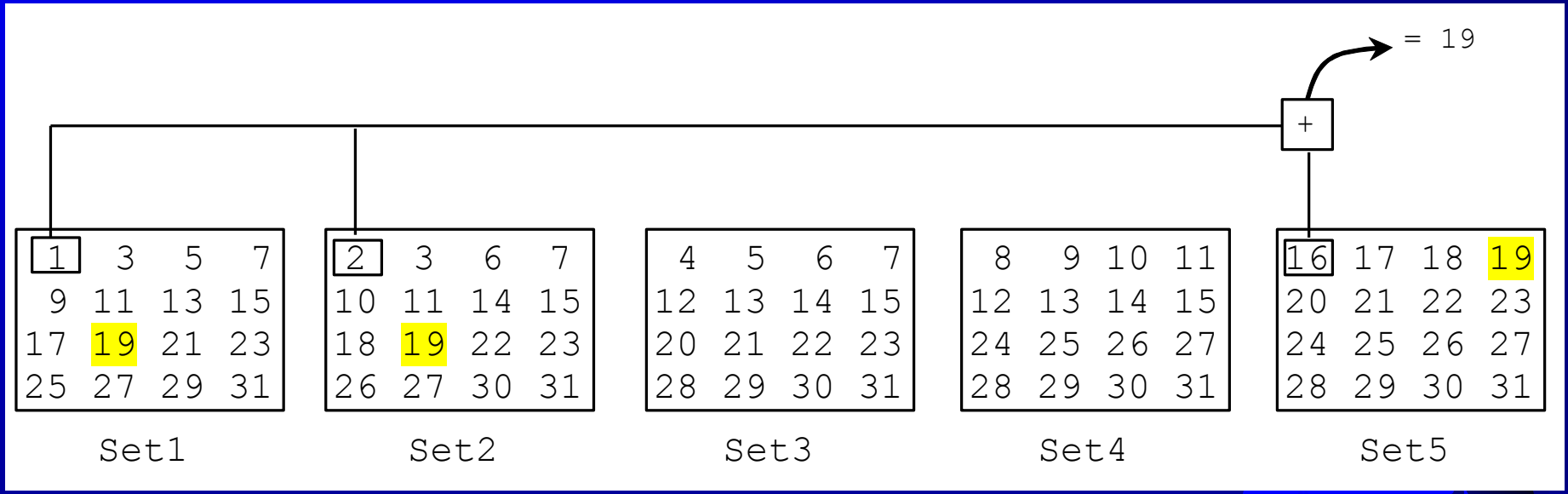
例题：字符串比较大小

LISTING 4.2 OrderTwoCities.java

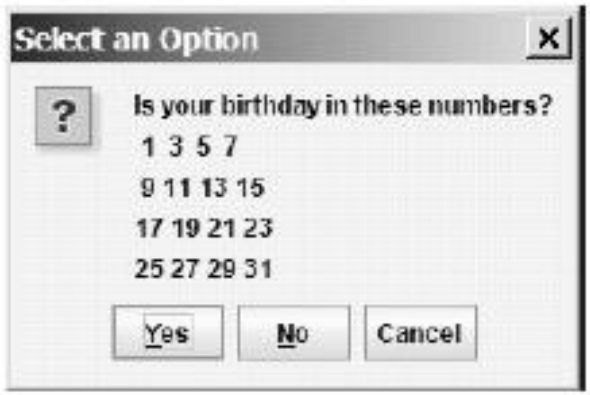
```
1  import java.util.Scanner;
2
3  public class OrderTwoCities {
4    public static void main(String[] args) {
5      Scanner input = new Scanner(System.in);
6
7      // Prompt the user to enter two cities
8      System.out.print("Enter the first city: ");
9      String city1 = input.nextLine();                    input city1
10     System.out.print("Enter the second city: ");
11     String city2 = input.nextLine();                    input city2
12
13     if (city1.compareTo(city2) < 0)                     compare two cities
14       System.out.println("The cities in alphabetical order are " +
15         city1 + " " + city2);
16     else
17       System.out.println("The cities in alphabetical order are " +
18         city2 + " " + city1);
19   }
20 }
```

例题：猜生日

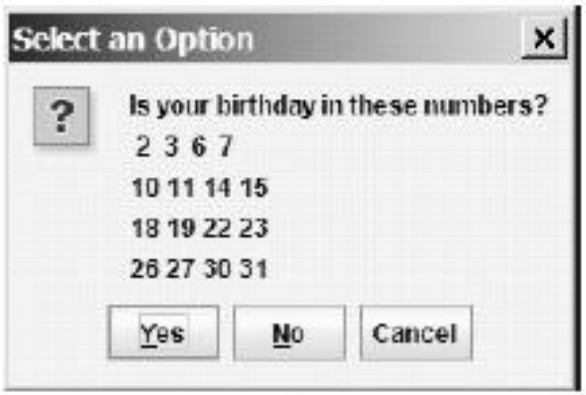
下面有5个表，逐一提问用户生日是否在每一个表中，最后可以猜出用户生日。



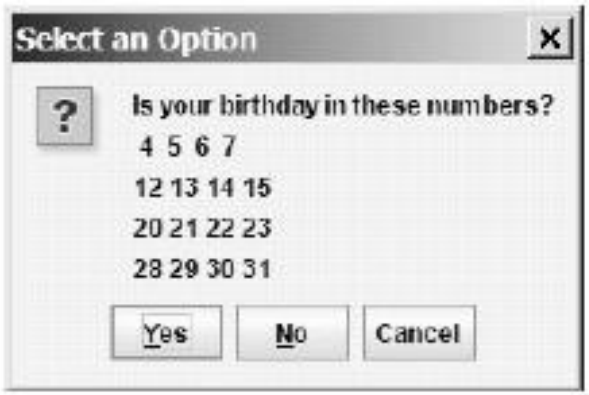
某次执行结果



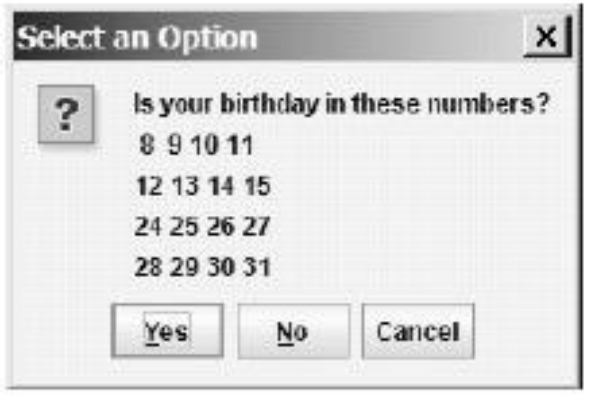
(a)



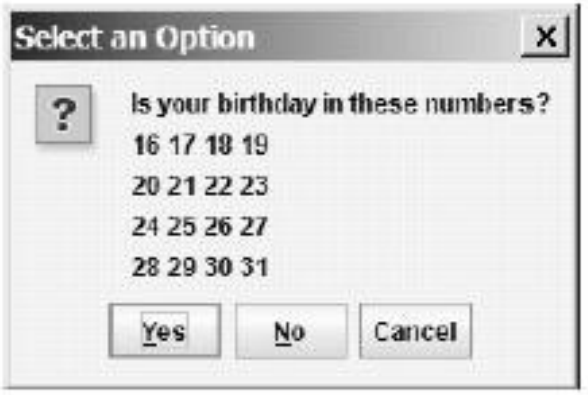
(b)



(c)



(d)

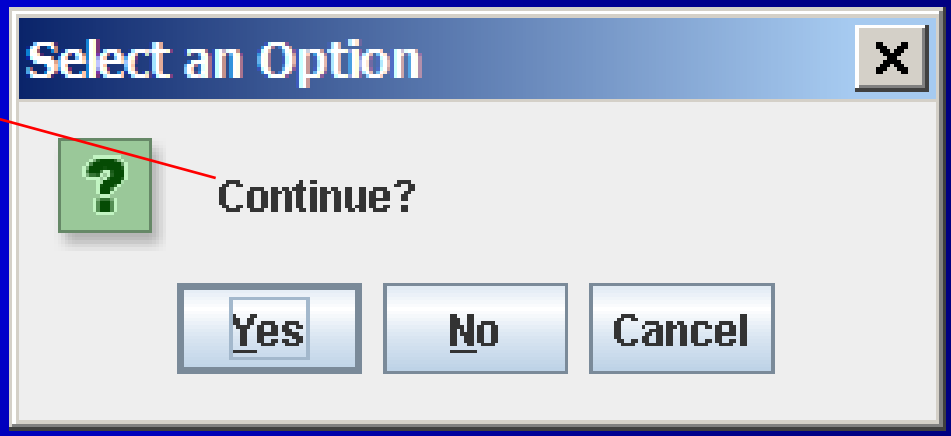


(e)

(f)

确认对话框

```
int option = JOptionPane.showConfirmDialog
    (null, "Continue");
```



返回值有三种：

- Yes按钮：**JOptionPane.YES_OPTION (0)**
- No按钮：**JOptionPane.NO_OPTION (1)**
- Cancel按钮：**JOptionPane.CANCEL_OPTION (2)**

代码1/3

```
import javax.swing.JOptionPane;
 public class GuessBirthday {
 public static void main(String[] args) {
   String set1 = " 1 3 5 7\n" + " 9 11 13 15\n" +
"17 19 21 23\n" + "25 27 29 31";
   String set2 = " 2 3 6 7\n" + "10 11 14 15\n" +
"18 19 22 23\n" + "26 27 30 31";
   String set3 = " 4 5 6 7\n" + "12 13 14 15\n" +
"20 21 22 23\n" + "28 29 30 31";
   String set4 = " 8 9 10 11\n" + "12 13 14 15\n" +
"24 25 26 27\n" + "28 29 30 31";
   String set5 = "16 17 18 19\n" + "20 21 22 23\n"
+ "24 25 26 27\n" + "28 29 30 31";
```

代码2/3

```
int day = 0;
//挨个表问过去，5个表权重分别为1,2,4,8,16，求和结果即为生日
int answer = JOptionPane.showConfirmDialog(null,
"Is your birthday in these numbers?\n" + set1);
if (answer == JOptionPane.YES_OPTION)
  day += 1;
answer = JOptionPane.showConfirmDialog(null,
"Is your birthday in these numbers?\n" + set2);
if (answer == JOptionPane.YES_OPTION)
  day += 2;
answer = JOptionPane.showConfirmDialog(null,
"Is your birthday in these numbers?\n" + set3);
if (answer == JOptionPane.YES_OPTION)
  day += 4;
```

代码3/3

```
  answer = JOptionPane.showConfirmDialog(null,
  "Is your birthday in these numbers?\n" + set4);
  if (answer == JOptionPane.YES_OPTION)
    day += 8;
  answer = JOptionPane.showConfirmDialog(null,
  "Is your birthday in these numbers?\n" + set5);
  if (answer == JOptionPane.YES_OPTION)
    day += 16;

  JOptionPane.showMessageDialog(null, "Your birthday is " +
  day + "!");
  }
}
```

THE END