**每日作业卷答案**

**就业班JavaSE第6天**

# 关卡1

## 训练案例1

### 训练描述

请说出你对Date类的理解

### 操作步骤答案

1. Date类表示特定的瞬间，精确到毫秒。
2. 使用空参构造创建日期对象,表示的就是当前操作系统的日期
3. 调用Date对象的getTime()方法,可以获得该Date对象对应的毫秒值
4. Date类有一个有参构造,可传入一个毫秒,创建一个这个毫秒值对应的日期对象.

## 训练案例2

### 训练描述

请阐述如何实现日期对象与指定格式的字符串之间的转换

### 操作步骤答案

1. 要完成日期对象与指定格式的字符串之间的转换核心类是SImpleDateFormat类
2. 使用SimpleDateFormat,创建对象时需要传入指定的格式;这个格式的规则是
   1. 当出现y时，会将y替换成年
   2. 当出现M时，会将M替换成月
   3. 当出现d时，会将d替换成日
   4. 当出现H时，会将H替换成时
   5. 当出现m时，会将m替换成分
   6. 当出现s时，会将s替换成秒
3. 把日期转换为指定格式的字符串
   1. 创建SimpleDateFormat对象 df,传入日期格式字符串
   2. 调用df对象的format方法,传入一个日期,返回格式化的字符串
4. 把指定格式的字符串转换为日期
   1. 创建SimpleDateFormat对象 df,传入日期格式字符串;
   2. 调用df对象的 parse(String dateStr)方法,传入格式化字串,返回为日期对象

## 训练案例3

### 训练描述

请阐述Calendar的基本使用

### 操作步骤答案

1. 调用Calendar的静态方法getInstance() 获取Calendar对象
2. Calendar常见的方法:
   1. public int **get**(int field)获取指定时间字段的值;这里的字段,都是Calendar的静态属性
      1. YEAR 年
      2. MONTH 月，从0开始算起，最大11；0代表1月，11代表12月。
      3. DATE 天
      4. HOUR 时
      5. MINUTE分
      6. SECOND秒
   2. public final void **set**(int field,int value) 设置指定字段的值
   3. public abstract void add(int field,int value) 给定的日历字段添加或减去指定的时间值;正数表示加上指定的时间值,负数表示减去指定的时间值
   4. public final Date **getTime**() 获取当前日历对用的日期对象
3. 注意事项:
   1. 西方星期的开始为周日，中国为周一。
   2. 在Calendar类中，月份的表示是以0-11代表1-12月。
   3. 日期是有大小关系的，时间靠后，时间越大。

## 训练案例4

### 训练描述

请阐述System类的理解

### 操作步骤答案

1. System类用于获取系统的属性信息,它的特点是:
   1. 构造方法私有,不能创建对象
   2. 所有方法都是静态的,直接可以通过类名调用
2. System类常见的方法有
   1. currentTimeMillis() 获取当前系统时间与1970年01月01日00:00点之间的毫秒差值
   2. exit(int status) 用来结束正在运行的Java程序。参数传入一个数字即可。通常传入0记为正常状态，其他为异常状态
   3. gc() 用来运行JVM中的垃圾回收器，完成内存中垃圾的清除。
   4. getProperty(String key) 用来获取指定键(字符串名称)中所记录的系统属性信息

## 训练案例5

### 训练描述

请阐述你对Math类的理解

### 操作步骤答案

1. Math类是进行基本的数学运算的工具类; 工具类的特点是方法都是静态的
2. 常见的方法
   1. abs(double a) 方法,结果都为正数(绝对值);
   2. ceil(double a)方法，结果为比参数值大的最小整数的double值
   3. floor(double a)方法，结果为比参数值小的最大整数的double值
   4. max(double a,double b)方法，返回两个参数值中较大的值
   5. min(double a,double b)方法，返回两个参数值中较小的值
   6. pow(double a,double b)方法，返回第一个参数的第二个参数次幂的值
   7. round(double a)方法，返回参数值四舍五入的结果
   8. random()方法，产生一个大于等于0.0且小于1.0的double小数

## 训练案例6

### 训练描述

请阐述你对基本数据类型包装类的理解

### 操作步骤答案

1. 包装类的作用是: 提供更多的操作基本数值的功能
2. 基本数据类型对应的八个包装类分别是
   1. byte --> Byte
   2. short --> Short
   3. int-->Integer
   4. long-->Long
   5. char-->Character
   6. boolean-->Boolean
   7. float-->Float
   8. double-->Double
3. 把字符串转换为对应的基本数据类型
   1. 调用需要转换的基本数据类型对应包装类的parseXXX(String s);其中XXX表示基本类型，参数为可以转成基本类型的字符串
   2. 注意: 必须保证传入字符串可以转换为基本数据类型,不能包含空格.否则将会发生数字转换的问题[NumberFormatException](mk://../../java/lang/NumberFormatException.html)

## 训练案例7

### 训练描述

请阐述你对自动装箱和自动拆箱的理解

### 操作步骤答案

1. 自动装箱和自动拆箱都是JDK1.5的新特性
2. 自动装箱: 在需要的时候,系统会自动把 基本数据类型转换为其对应的包装类型
3. 自动拆箱: 在需要的时候,系统会自动把 包装类型转换为其对应的基本数据类型

## 训练案例8

### 训练描述

请阐述你对正则表达式的理解

### 操作步骤答案

1. 正则表达式是一个用于定义匹配规则的字符串
2. 正则表达式的作用是验证,检索,替换符合某个规则的文本
3. 字符串中关于正则表达式的常见方法?
   1. public boolean matches(String regex) 判断字符串是否匹配给定的规则
   2. public String[] split(String regex) 根据给定正则表达式的匹配规则，拆分此字符串
4. 请说出正则表达式的匹配规则是什么?
   1. 字符:
      1. x: 代表的是字符x
      2. \\: 代表的是斜线字符'\'
      3. \t: 制表符
      4. \n:换行符
      5. \r:回车符
   2. 字符类
      1. [abc]: 匹配 a、b 或 c中任意一个字符
      2. [^abc]: 匹配除了 a、b 或 c以外的任意一个字符
      3. [a-zA-Z] 匹配字母 中任意一个字符
      4. [0-9]: 匹配任意一个数字
      5. [a-zA-Z\_0-9]: 匹配字母,数字,\_中的任意一个
   3. 预定义字符类
      1. . 用于匹配除换行符以外的任意一个字符;如果要匹配"."号的话,需要使用"\\."
      2. \d: 匹配任意一个数字
      3. \w:匹配字母,数字,\_中的任意一个
      4. \s: 匹配任意一个空白符: 空格,制表符,换行符中的任意一个
   4. 边界匹配器
      1. ^: 代表行的开头
      2. $: 代表行的结尾
      3. \b: 代表单词边界
   5. 数量词:
      1. X?: 代表的是X出现一次或一次也没有
      2. X\*: 代表的是X出现零次或多次
      3. X+: 代表的是X出现一次或多次
      4. X{n}:代表的是X出现恰好 n 次
      5. X{n,}: 代表的是X出现至少 n 次
      6. X{n,m}: 代表的是X出现至少 n 次，但是不超过 m 次
   6. 逻辑符:
      1. XY: 代表的是X后跟Y
      2. X|Y: 代表的是X 或 Y
      3. (X): 代表的是（）括号内的数据作为一组数据出现，(X)的方式称为正则表达式中的组

## 训练案例9

### 训练描述

请用代码实现:

获取当前的日期,并把这个日期转换为指定格式的字符串,如2088-08-08 08:08:08

### 操作步骤答案

/\*

1.创建包com.itheima.level1

2.定义类Test09

3.提供main方法,在main方法中

a)获取当前日期对象 now;

b)创建SimpleDateFormat对象 df,并制定日期格式

c)调用df的format(Date date) 方法,传入now; 接收返回的字符串

d)打印这个字符串

\*/

**public** **class** Test09 {

**public** **static** **void** main(String[] args) {

// a)获取当前日期对象 now;

Date now = **new** Date();

// b)创建SimpleDateFormat对象 df,并制定日期格式

SimpleDateFormat df = **new** SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

// c)调用df的format(Date date) 方法,传入now; 接收返回的字符串

String datestr = df.format(now);

// d)打印这个字符串

System.***out***.println(datestr);

}

## }训练案例10

### 训练描述

请用代码实现如下需求:

把你的生日字符串表示转换为对应的日期对象.

### 操作步骤答案

/\*

1.在com.itheima.level1包下创建Test10类

2.在Test10类中,提供main方法,在main方法中

a)创建SimpleDateFormat对象 df,并制定日期格式

b)调用df的parse(String str)方法,传入你的生日字符串;结束返回的日期对象

c)打印这个日期对象

d)使用try...catch处理解析异常

\*/

**public** **class** Test10 {

**public** **static** **void** main(String[] args) {

**try** {

// a)创建SimpleDateFormat对象 df,并制定日期格式

SimpleDateFormat df = **new** SimpleDateFormat("yyyy-MM-dd");

// b)调用df的parse(String str)方法,传入你的生日字符串;结束返回的日期对象

Date date = df.parse("1999-09-09");

// c)打印这个日期对象

System.***out***.println(date);

} **catch** (ParseException e) {

e.printStackTrace();

}

}

}

## 训练案例11

### 训练描述

使用代码实现

500天后是几几年几月几日.

### 操作步骤答案

/\*

1.创建包com.itheima.level1中创建Test11

2.在Test11类中,提供main方法,在main方法中

a)获取当前日历对象

b)调用日期对象的add()方法,让当前日历向后移动500天

c)获取日历中的年,月,日

d)输出几几年几月几日.

\*/

**public** **class** Test11 {

**public** **static** **void** main(String[] args) {

// a)获取当前日历对象

Calendar c = Calendar.*getInstance*();

// b)调用日期对象的add()方法,让当前日历向后移动500天

c.add(Calendar.***DATE***, 500);

// c)获取日历中的年,月,日

**int** year = c.get(Calendar.***YEAR***);

**int** month = c.get(Calendar.***MONTH***);

**int** day = c.get(Calendar.***DAY\_OF\_MONTH***);

// d)输出几几年几月几日.

System.***out***.println(year+"年"+month+"月"+day+"日");

}

}

## 训练案例12

### 训练描述

编写一个校验用户名的程序,检测键盘录入的用户名是否合法

要求:必须以英文开头,只能包含英文,数字和`\_`;最少6位,做多12位

### 操作步骤答案

/\*

1.在com.itheima.level1中创建Test12类

2.在Test12类中提供main方法,在main方法中

a)创建键盘录入对象

b)接收用户录入的用户名字符串

c)调用字符串的matches方法,判断用户名是否合法

d)如果合法就打印: 用户名合法

e)如果不合法就打印: 用户名不合法

\*/

**public** **class** Test12 {

**public** **static** **void** main(String[] args) {

// a)创建键盘录入对象

Scanner sc = **new** Scanner(System.***in***);

// b)接收用户录入的用户名字符串

System.***out***.println("请录入登陆用户名");

String userName = sc.nextLine();

// c)调用字符串的matches方法,判断用户名是否合法

// 必须以英文开头,只能包含英文,数字和`\_`;最少6位,做多12位

**if**(userName.matches("[a-zA-Z]\\w{5,11}")){

// d)如果合法就打印: 用户名合法

System.***out***.println("用户名合法");

}**else**{

// e)如果不合法就打印: 用户名不合法

System.***out***.println("用户名不合法");

}

}

}

# 关卡2

## 训练案例1

### 训练描述

请使用代码实现

计算你活了多少天

### 操作步骤答案

/\*

1.创建包com.itheima.level2

2.在包中创建测试类Test13

3.在Test13类中,提供main方法,在main方法中

a)把你生日的字符串,转换为日期对象

b)把日期对象转换为对应的毫秒值

c)获取当前系统的毫秒值

d)当前系统毫秒值-出生时间对于的毫秒值计算你活的毫秒值

e)把你活毫秒值转换为天

f)输出天数

g)使用try...catch处理异常

\*/

**public** **class** Test13 {

**public** **static** **void** main(String[] args) {

**try** {

// a)把你生日的字符串,转换为日期对象

SimpleDateFormat df = **new** SimpleDateFormat("yyyy-MM-dd");

Date birthDay = df.parse("1999-09-09");

// b)把日期对象转换为对应的毫秒值

**long** birthTime = birthDay.getTime();

// c)获取当前系统的毫秒值

**long** now = System.*currentTimeMillis*();

// d)当前系统毫秒值-出生时间对于的毫秒值计算你活的毫秒值

**long** liveTime = now - birthTime;

// e)把你活毫秒值转换为天

**long** day = liveTime / 1000 / 60 / 60 / 24;

// f)输出天数

System.***out***.println("活了"+day+"天");

} **catch** (ParseException e) {

e.printStackTrace();

}

}

}

## 训练案例2

### 训练描述

写代码验证使用SimpleDateFormat把日期转换为字符串的时候;

创建SimpleDateFormat对象,比较消耗程序的执行性能

### 操作步骤答案

/\*

1.在com.itheima.level2包下创建Test14类

2.在Test14类中提供main方法,在main方法中

a)使用System的currentTimeMillis()方法获取当前操作系统的毫秒值,作用程序执行的开始时间,使用start变量接收

b)需要测试执行性能的代码

c)使用System的currentTimeMillis()方法获取当前操作系统的毫秒值,作用程序执行的结束时间,使用end变量接收

d)计算代码执行花费的时间 end - start,输出代码执行消耗的时间

3.在Test14类中提供一个静态方法testSimpleDataFormatInLoop(),在方法中

a)创建日期(Date)对象 now

b)写一个循环10000次的for循环

c)在循环中

i.创建SimpleDateFormat对象 df,并制定日期格式为yyyy-MM-dd

ii.调用df的format(Date date),传入now

d)在main方法中,需要测试执行的性能的代码处调用testSimpleDataFormatInLoop()方法

e)记录程序执行的效率

4.在Test14类,完整复制testSimpleDataFormatInLoop()方法

a)改名为testSimpleDataFormatOutLoop

b)把创建SimpleDateFormat的代码移动到循环外部

c)在main方法中,注释调用testSimpleDataFormatInLoop();

d)在语句下一行调用testSimpleDataFormatOutLoop方法

e)运行程序,查看输出结果,并记录

5.得出结论: 创建日期格式对象,确实比较消耗程序性能,所以在实际开发一般会写一个工具类专门处理,日期的格式化与解析.

\*/

**public** **class** Test14 {

**public** **static** **void** main(String[] args) {

// a)使用System的currentTimeMillis()方法获取当前操作系统的毫秒值,作用程序执行的开始时间,使用start变量接收

**long** start = System.*currentTimeMillis*();

// b)需要测试执行性能的代码

// testSimpleDataFormatInLoop();//375

*testSimpleDataFormatOutLoop*();//80

// c)使用System的currentTimeMillis()方法获取当前操作系统的毫秒值,作用程序执行的结束时间,使用end变量接收

**long** end = System.*currentTimeMillis*();

// d)计算代码执行花费的时间 end - start,输出代码执行消耗的时间

**long** cost = end - start;

System.***out***.println("花费的时间为:"+cost);

}

// 创建SimpleDateFormat对象的代码在循环里面

**public** **static** **void** testSimpleDataFormatInLoop(){

// a)创建日期(Date)对象 now

Date now = **new** Date();

// b)写一个循环10000次的for循环

**for**(**int** i=0; i<100000; i++){

// c)在循环中

// i.创建SimpleDateFormat对象 df,并制定日期格式为yyyy-MM-dd

SimpleDateFormat df = **new** SimpleDateFormat("yyyy-MM-dd");

// ii.调用df的format(Date date),传入now

df.format(now);

}

}

// 创建SimpleDateFormat对象的代码在循环外边

**public** **static** **void** testSimpleDataFormatOutLoop(){

// a)创建日期(Date)对象 now

Date now = **new** Date();

// b)写一个循环10000次的for循环

// 创建SimpleDateFormat对象 df,并制定日期格式为yyyy-MM-dd

SimpleDateFormat df = **new** SimpleDateFormat("yyyy-MM-dd");

**for**(**int** i=0; i<100000; i++){

// c)在循环中

// ii.调用df的format(Date date),传入now

df.format(now);

}

}

}

## 训练案例3

### 训练描述

使用正则表达式切割字符串

\* 192.168.105.27 按照 . 号切割字符串

\* 18 22 40 65 按照空格切割字符串

提示: 由于 . 号 在正则表达式中用于匹配除换行符以外的任意字符,会把字符串切的连渣都不剩,此处使用 . 号需要转义; 使用\\.作为切割使用正则表达式

### 操作步骤答案

/\*

1.在com.itheima.level2包下创建Test15

2.在Test15类中提供main方法,在main方法中

a)定义字符串变量str1,赋值为192.168.105.27

b)调用str1的split的方法,传入 “\\.”,使用字符春数组strs1接收

c)遍历数组strs1,输出每一个元素

d)定义字符串变量str2,赋值为 18 22 40 65

e)调用str1的split的方法,传入“\\s+”,使用字符春数组strs2接收

f)遍历数组strs2,输出每一个元素

\*/

**public** **class** Test15 {

**public** **static** **void** main(String[] args) {

// a)定义字符串变量str1,赋值为192.168.105.27

String str1 = "192.168.105.27";

// b)调用str1的split的方法,传入 “\\.”,使用字符春数组strs1接收

String[] strs1 = str1.split("\\.");

// c)遍历数组strs1,输出每一个元素

**for** (**int** i = 0; i < strs1.length; i++) {

System.***out***.print(strs1[i]+" ");

}

System.***out***.println();

// d)定义字符串变量str2,赋值为 18 22 40 65

String str2 = "18 22 40 65";

// e)调用str1的split的方法,传入“\\s+”,使用字符春数组strs2接收

String[] strs2 = str2.split("\\s+");

// f)遍历数组strs2,输出每一个元素

**for** (**int** i = 0; i < strs2.length; i++) {

System.***out***.print(strs2[i]+" ");

}

System.***out***.println();

}

}

## 训练案例4

### 训练描述

请使用代码描述

有一段文本 " anglebaby 女 18 515151511 "

分别描述一个人的姓名 性别 年龄(int) 和QQ号码信息,完成如下需求:

1. 创建一个Person类,包含性别,性别,年龄和QQ号码属性,使用解析出来信息创建一个Person对象,并打印这个人信息

2. 并判断QQ号是否合法,匹配规则,全部都是数字,第一位非0,长度为5到12位.

### 操作步骤答案

/\*

1.在com.itheima.level2包中

2.创建Student类

a)属性: 姓名(name),性别(sex),年龄(int age),QQ号码(qq)

b)提供空参,有参构造方法以及setters与getters方法

c)重写toString()方法,返回属性值

\*/

**public** **class** Student {

// 姓名(name)

**private** String name;

// 性别(sex)

**private** String sex;

// 年龄(int age)

**private** **int** age;

// QQ号码(qq)

**private** String qq;

// b)提供空参,有参构造方法以及setters与getters方法

**public** Student() {

**super**();

}

**public** Student(String name, String sex, **int** age, String qq) {

**super**();

**this**.name = name;

**this**.sex = sex;

**this**.age = age;

**this**.qq = qq;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getSex() {

**return** sex;

}

**public** **void** setSex(String sex) {

**this**.sex = sex;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

**public** String getQq() {

**return** qq;

}

**public** **void** setQq(String qq) {

**this**.qq = qq;

}

// c)重写toString()方法,返回属性值

**public** String toString() {

**return** "Student [name=" + name + ", sex=" + sex + ", age=" + age + ", qq=" + qq + "]";

}

}

/\*

1.在com.itheima.level2包中

2.创建Student类

a)属性: 姓名(name),性别(sex),年龄(int age),QQ号码(qq)

b)提供空参,有参构造方法以及setters与getters方法

c)重写toString()方法,返回属性值

3.创建Test16类,在Test16类中:

a)提供main方法,在main方法中

i.定义字符串变量str,赋值为:" anglebaby 女 18 515151511 "

ii.取出字符str两端的空格,避免切割的时候,产生空字符串

iii.使用”\\s+”正则表达式切割字符串,因为中间的空格可能包含制表符,多个空格

iv.切割后,字符串数组0索引->姓名,1索引->性别,2索引->年龄,3索引->QQ号

v.把年龄字符串转换int类型,赋值给变量age

vi.使用满参构造创建Student对象s,传入对应的数据

vii.打印学生s信息

viii.判断该学生的QQ号码是否合法,如果合法输出QQ号码合法,否则输出QQ号码不合法

\*/

**public** **class** Test16 {

**public** **static** **void** main(String[] args) {

// i.定义字符串变量str,赋值为:" anglebaby 女 18 515151511 "

String str = " anglebaby 女 18 515151511 ";

// ii.取出字符str两端的空格,避免切割的时候,产生空字符串

str = str.trim();

// iii.使用”\\s+”正则表达式切割字符串,因为中间的空格可能包含制表符,多个空格

String[] strs = str.split("\\s+");

// iv.切割后,字符串数组0索引->姓名,1索引->性别,2索引->年龄,3索引->QQ号

// v.把年龄字符串转换int类型,赋值给变量age

**int** age = Integer.*parseInt*(strs[2]);

// vi.使用满参构造创建Student对象s,传入对应的数据

Student s = **new** Student(strs[0], strs[1], age, strs[3]);

// vii.打印学生s信息

System.***out***.println(s);

// 并判断QQ号是否合法,匹配规则,全部都是数字,第一位非0,长度为5到12位.

// viii.判断该学生的QQ号码是否合法,如果合法输出QQ号码合法,否则输出QQ号码不合法

**if**(s.getQq().matches("[1-9][0-9]{4,11}")){

System.***out***.println("QQ号码合法");

}**else**{

System.***out***.println("QQ号码不合法");

}

}

}

## 训练案例5

### 训练描述

通过代码实现:

把字符串 "Hello12345World6789012" 中所有数字替换为 "#"

提示: 字符串类的replaceAll(String regex,string str)方法,使用str替换掉当前字符串中所有与正则表达式匹配的字符

### 操作步骤答案

/\*

1.在com.itheima.level2包中

2.创建Test17类,在Test17类中:

a)提供main方法,在main方法中

b)定义字符串变量str,赋值为:"Hello12345World6789012"

c)调用str的replaceAll(String regex,String str),第一个参数传入”\\d+”,第二个参数传入”#”,定义字符串变量newStr接收替换后的字符串

d)输出: newStr

\*/

**public** **class** Test17 {

**public** **static** **void** main(String[] args) {

// b)定义字符串变量str,赋值为:"Hello12345World6789012"

String str = "Hello12345World6789012";

// c)调用str的replaceAll(String regex,String str),第一个参数传入”\\d+”,第二个参数传入”#”,定义字符串变量newStr接收替换后的字符串

String newStr = str.replaceAll("\\d", "#");

// d)输出: newStr

System.***out***.println(newStr);

}

}

# 关卡3

## 训练案例1

### 训练描述

使用SimpleDateFormat类,把2016-12-18转换为2016年12月18日.

提示: SimpleDateFormat对象的日期格式字符串可以通过**[applyPattern](mk:@MSITStore:C:\\Users\\58334\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/text/../../java/text/SimpleDateFormat.html" \l "applyPattern(java.lang.String))**([String](mk:@MSITStore:C:\\Users\\58334\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/text/../../java/lang/String.html" \o "java.lang 中的类) pattern)进行修改

要求: 异常处理使用throws的方式

### 操作步骤

1. 创建包:com.itheima.level3
2. 创建类Test18,提供main方法,在main方法中
   1. 创建SimpleDateFormat对象 df,指定日期模式为yyyy-MM-dd
   2. 调用df的parse(String str) 方法传入2016-12-18,得到对应日期类型
   3. 修改df的日期模式为yyyy年MM月dd日
   4. 调用df的format(Date date) 传入刚才转换为日期
   5. 打印转换后的字符串

### 操作步骤答案

/\*

1.创建包:com.itheima.level3

2.创建类Test18,提供main方法,在main方法中

a)创建SimpleDateFormat对象 df,指定日期模式为yyyy-MM-dd

b)调用df的parse(String str) 方法传入2016-12-18,得到对应日期类型

c)修改df的日期模式为yyyy年MM月dd日

d)调用df的format(Date date) 传入刚才转换为日期

e)打印转换后的字符串

\*/

**public** **class** Test18 {

**public** **static** **void** main(String[] args) **throws** ParseException {

// a)创建SimpleDateFormat对象 df,指定日期模式为yyyy-MM-dd

SimpleDateFormat df = **new** SimpleDateFormat("yyyy-MM-dd");

// b)调用df的parse(String str) 方法传入2016-12-18,得到对应日期类型

Date date = df.parse("2016-12-18");

// c)修改df的日期模式为yyyy年MM月dd日

df.applyPattern("yyyy年MM月dd日");

// d)调用df的format(Date date) 传入刚才转换为日期

String newStr = df.format(date);

// e)打印转换后的字符串

System.***out***.println(newStr);

}

}

## 训练案例2

### 训练描述

请使用代码实现

分别使用String的+= 和StringBuilder的append方法对字符串做100000次拼接,

计算String拼接100000次花费时间与StringBuilder拼接100000次所花费时间并打印

### 操作步骤

1. 在包com.itheima.level3中,创建类Test19类,在Test19类中
2. 提供main方法,在main方法中
   1. 使用System的currentTimeMillis()方法获取当前操作系统的毫秒值,作用程序执行的开始时间,使用start变量接收
   2. 需要测试执行性能的代码
   3. 使用System的currentTimeMillis()方法获取当前操作系统的毫秒值,作用程序执行的结束时间,使用end变量接收
   4. 计算代码执行花费的时间 end - start,输出代码执行消耗的时间
3. 写一个静态方法: testString(),在该方法中
   1. 定义一个字符串 str,内容随意
   2. 写一个循环100000次for循环,在循环中写上
      1. str +=”随机内容” ; 这里不要写str += (str+”xxx”) 这样会导致堆内存溢出错误.
   3. 在mian方法中,需要测试性能的代码处,调用本方法
   4. 运行程序,记录执行的时间
   5. 注释掉这个调用
4. 写一个静态方法:testStringBuilder(),在方法中
   1. 创建一个StringBuilder对象sb,初始内容与testString()中的字符串相同
   2. 写一个循环100000次for循环,在循环中写上
      1. 调用sb.append()方法,传入的内容与testString()方法中+=后面的内容一样
   3. 循环结束调用sb.toString()方法转换为字符串
   4. 在mian方法中,需要测试性能的代码处,调用本方法
   5. 运行程序,记录执行的时间

### 操作步骤答案

/\*

1.在包com.itheima.level3中,创建类Test19类,在Test19类中

2.提供main方法,在main方法中

\*/

**public** **class** Test19 {

**public** **static** **void** main(String[] args) {

// a) 使用System的currentTimeMillis()方法获取当前操作系统的毫秒值,作用程序执行的开始时间,使用start变量接收

**long** start = System.*currentTimeMillis*();

// b)需要测试执行性能的代码

//testString(); //消耗时间: 29295毫秒

*testStringBuilder*();//消耗时间:6毫秒

// c)使用System的currentTimeMillis()方法获取当前操作系统的毫秒值,作用程序执行的结束时间,使用end变量接收

**long** end = System.*currentTimeMillis*();

// d)计算代码执行花费的时间 end - start,输出代码执行消耗的时间

System.***out***.println("所花费的时间为:"+(end-start));

}

// 3.写一个静态方法: testString(),在该方法中

**public** **static** **void** testString(){

// a)定义一个字符串 str,内容随意

String str = "hello";

// b)写一个循环100000次for循环,在循环中写上

**for**(**int** i = 0; i<100000;i++){

// i.str +=”随机内容” ; 这里不要写str += (str+”xxx”) 这样会导致堆内存溢出错误.

str += "world";

}

}

// 4. 写一个静态方法:testStringBuilder(),在方法中

**public** **static** **void** testStringBuilder(){

// a)创建一个StringBuilder对象sb,初始内容与testString()中的字符串相同

StringBuilder sb = **new** StringBuilder("hello");

// b)写一个循环100000次for循环,在循环中写上

**for**(**int** i = 0; i<100000;i++){

// i.调用sb.append()方法,传入的内容与testString()方法中+=后面的内容一样

sb.append("world");

}

// c)循环结束调用sb.toString()方法转换为字符串

String newStr = sb.toString();

}

}

## 训练案例3

### 训练描述

模拟验证手机号的过程，按照以下要求实现相关代码（友情提示：不一定要用正则）

a. 提示用户在控制台输入手机号，用一个字符串对其进行保存

b. 判断该手机号位数是否是11位

c. 判断该手机号，是否都是数字

d. 判断改手机号最后3为数字如果是相同数字则认为是靓号

### 操作步骤描述

1. 在包:com.itheima.level3中,创建类Test20,
2. 在Test19类中提供main方法,在mian方法中
   1. 创建键盘录入对象
   2. 获取用户录入的手机号码
   3. 判断该手机号位数是否是11位,如果不是打印手机号码位数有误,方法结束
   4. 判断该手机号，是否都是数字; ,如果不是打印存在非法字符,方法结束
   5. 判断改手机号最后3为数字如果是相同数字则认为是靓号,输出手机号是靓号,否则输出手机号不是靓号

### 操作步骤答案

/\*

1.在包:com.itheima.level3中,创建类Test19,

2.在Test19类中提供main方法,在mian方法中

\*/

**public** **class** Test20 {

**public** **static** **void** main(String[] args) {

// a)创建键盘录入对象

Scanner sc = **new** Scanner(System.***in***);

// b)获取用户录入的手机号码

System.***out***.println("请录入一个手机号码");

String phone = sc.nextLine();

// c)判断该手机号位数是否是11位,如果不是打印手机号码位数有误,方法结束

**if**(phone.length() != 11){

System.***out***.println("手机号码位数有误");

**return**;

}

// d)判断该手机号，是否都是数字; ,如果不是打印存在非法字符,方法结束

**if**(!phone.matches("\\d+")){

System.***out***.println("存在非法字符");

**return**;

}

// e) 判断改手机号最后3为数字如果是相同数字则认为是靓号,输出靓号,否则输出不是靓号

// 获取后三位的字符

**char** last1 = phone.charAt(phone.length()-1);

**char** last2 = phone.charAt(phone.length()-2);

**char** last3 = phone.charAt(phone.length()-3);

// 如果后三位的字符相同,则认为是靓号

**if**(last1 == last2 && last2 == last3){

System.***out***.println("手机号"+phone+"是靓号");

}**else**{

System.***out***.println("手机号"+phone+"不是是靓号");

}

}

}

## 训练案例4

### 训练描述

请使用实现:

已知项目的根目录下有一个student.txt,内容如下:

1 郝婷婷 女 30

2 赵鹏 男 35

3 周强 男 38

4 李艳欣 女 37

5 朱可可 女 41

6 张少荣 女 30

7 王小军 男 28

该文件中,每一行表示一个学生信息,从前往后依次是学号,姓名,姓名,年龄

要求:

1. 读取文件内容,把每一个学生信息封装为一个学生对象,存储到ArrayList集合中
2. 打印集合中的学生信息
3. 计算平均年分
4. 异常处理使用throws的方式

### 操作步骤描述

1. 在项目的根目录下创建student.txt文件,把内容拷贝到这个文件中
2. 在包:com.itheima.level3中
3. 创建Student类
   1. 属性: 学号(id),姓名(name),性别(sex),年龄(int age)
   2. 提供空参,有参构造方法,setters和getters方法
   3. 重写toString()方法
4. 创建Test21类,提供main方法,在main方法中
   1. 创建ArrayList集合对象 list,用于存储读取到的学生对象
   2. 创建BufferedReader对象br,绑定数据源
   3. 定义字符变量line;
   4. 使用while循环,每次读一行,在循环中
      1. 调用trim方法,取出读取到字符串两端的空格
      2. 使用split(“\\s+”) 拆分内容,返回字符串数组
      3. 数组的元素0索引是学号,1索引是姓名,2索引是性别,3索引是年龄
      4. 把字符串的年龄转换为整数的年龄
      5. 创建有参构造Student对象s
      6. 把学生对象添加集合中
   5. 遍历list集合,打印每一个学生信息
   6. 统计学生的平均年龄

### 操作步骤答案

/\*

1.在项目的根目录下创建student.txt文件,把内容拷贝到这个文件中

2.在包:com.itheima.level3中

3.创建Student类

a)属性: 学号(id),姓名(name),性别(sex),年龄(int age)

b)提供空参,有参构造方法,setters和getters方法

c)重写toString()方法

\*/

**public** **class** Student {

// 学号(id)

**private** String id;

// 姓名(name)

**private** String name;

// 性别(sex)

**private** String sex;

// 年龄(int age)

**private** **int** age;

// b)提供空参,有参构造方法,setters和getters方法

**public** Student() {

**super**();

}

**public** Student(String id, String name, String sex, **int** age) {

**super**();

**this**.id = id;

**this**.name = name;

**this**.sex = sex;

**this**.age = age;

}

**public** String getId() {

**return** id;

}

**public** **void** setId(String id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getSex() {

**return** sex;

}

**public** **void** setSex(String sex) {

**this**.sex = sex;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

// c)重写toString()方法

**public** String toString() {

**return** "Student [id=" + id + ", name=" + name + ", sex=" + sex + ", age=" + age + "]";

}

}

/\*

4.创建Test21类,提供main方法,在main方法中

a)创建ArrayList集合对象 list,用于存储读取到的学生对象

b)创建BufferedReader对象br,绑定数据源

c)定义字符变量str;

d)使用while循环,每次读一行,在循环中

i.调用trim方法,取出读取到字符串两端的空格

ii.使用split(“\\s+”) 拆分内容,返回字符串数组

iii.数组的元素0索引是学号,1索引是姓名,2索引是性别,3索引是年龄

iv.把字符串的年龄转换为整数的年龄

v.创建有参构造Student对象s

vi.把学生对象添加集合中

e)遍历list集合,打印每一个学生信息

f)统计学生的平均年龄

\*/

**public** **class** Test21 {

**public** **static** **void** main(String[] args) **throws** IOException {

// a)创建ArrayList集合对象 list,用于存储读取到的学生对象

ArrayList<Student> list = **new** ArrayList<>();

// b)创建BufferedReader对象br,绑定数据源

BufferedReader br = **new** BufferedReader(**new** FileReader("student.txt"));

// c)定义字符变量str;

String line;

// d)使用while循环,每次读一行,在循环中

**while**((line=br.readLine())!=**null**){

// i.调用trim方法,取出读取到字符串两端的空格

line = line.trim();

// ii.使用split(“\\s+”) 拆分内容,返回字符串数组

String[] strs = line.split("\\s+");

// iii.数组的元素0索引是学号,1索引是姓名,2索引是性别,3索引是年龄

// iv.把字符串的年龄转换为整数的年龄

**int** age = Integer.*parseInt*(strs[3]);

// v.创建有参构造Student对象s

Student s = **new** Student(strs[0],strs[1],strs[2], age);

// vi.把学生对象添加集合中

list.add(s);

}

// e)遍历list集合,打印每一个学生信息

**for** (Student student : list) {

System.***out***.println(student);

}

// f)统计学生的平均年龄

**int** sum = 0;

**for** (**int** i = 0; i < list.size(); i++) {

sum += list.get(i).getAge();

}

// 计算平均分

**double** avg = (**double**)sum / list.size();

System.***out***.println("平均年龄为:"+avg);

}

}