●常见对象(正则表达式的概述和简单使用)

- A:正则表达式
  - 是指一个用来描述或者匹配一系列符合某个语法规则的字符串的单个字符串。其实就是一种规则。有自己特殊的应用。

作用：比如注册邮箱，邮箱用户名和密码，一般会对其进行限制长度，这个限制长度的事情就是正则表达式做的。

  - B:案例演示
  - 需求：校验 qq 号码.
    - 1:要求必须是 6-12 位数字
    - 2:0 不能开头
    - 3:必须都是数字
- a:非正则表达式实现

```
package com.sxt.homework;

public class Demo {
    public static void main(String[] args) {
        /*
           1）要求必须是6-12 位数字  :  qq.length
            2）0 不能开头    :qq.startWith("0")判读是否以0 开头
             3）必须都是数字 :  '7' >= 48 && '7' <= 57
         */
        String qq = "78978967823";
        String result =  check(qq)? "是正确的 qq 号" : "qq 不合法";
        System.out.println(result);
    }
    public static boolean check(String qq){
        boolean flag = true;//申明一个布尔值
      if(qq.length() >= 6 && qq.length() <= 12){
            //满足条件:长度在6-12 位之间
          if(!qq.startsWith("0")){
                //满足0 不能开头
            for (int i = 0; i <qq.length() ; i++) {
                    char c = qq.charAt(i);
                    if(c >=48 && c<= 57){
                        //满足都是数字

                    }else{
                            flag = false;
                        }
                    }
            }else{
                //这里0 开头进入这里
             flag = false;
            }
        }else{
            flag = false;
        }
        return flag;
```

```
    }
}
```

- b:正则表达式实现

```
package com.sxt.homework;

public class Demo {
    public static void main(String[] args) {

        String qq = "089780967831";
        String regex = "[1-9]\\d{5,11}";

        boolean result = qq.matches(regex);//matches 判断是否匹配,如果匹配返回 true
        System.out.println(result);
        /*
             [1-9]\\d{5,11}   正则表达式
             [1-9] :表示匹配 1~9 之间的任意一个数
             \\d  :第一个\是转义符; \d :表示 0~9 之间的数字
             {5,11}:表示\\d 最少重复出现 5 次,最多重复出现 11 次

         */
    }

}
```

●常见对象(预定义字符类演示)

- A:预定义字符类
  - . 任何字符(一个点代表一个任意字符)。
  - \d 数字：等同于[0-9]
  - \w 单词字符：等同于[a-zA-Z_0-9]
  - [abc] a、b 或 c（简单类）
  - [^abc] 任何字符，除了 a、b 或 c（否定）
  - [a-zA-Z] a 到 z 或 A 到 Z，两头的字母包括在内（范围）
  - [0-9] 0 到 9 的字符都包括

```
package com.sxt.homework;
public class Demo {
    public static void main(String[] args) {

      /*
    . 任何字符(一个点代表一个任意字符)。
   \d 数字：等同于[0-9]
   \w 单词字符：等同于[a-zA-Z_0-9]
   [abc] a、b 或 c（简单类）
```

```
    ▪ [^abc] 任何字符，除了 a、b 或 c（否定）
    ▪ [a-zA-Z] a 到 z 或 A 到 Z，两头的字母包括在内（范围）
    ▪ [0-9] 0 到9 的字符都包括
        */
        System.out.println("------匹配任何一个字符-------");
        String regex1 = ".";
        System.out.println("长".matches(regex1));
        System.out.println("------匹配\\d 匹配 0-9 之间任意一个字符-------");
        String regex2 = "\\d";
        System.out.println("3".matches(regex2));
        System.out.println("01".matches(regex2));
        System.out.println("------匹配\\w 匹配 a-zA-Z_0-9 之间任意一个字符-------");

        String regex3 = "\\w";
        System.out.println("2".matches(regex3));
        System.out.println("w".matches(regex3));
        System.out.println("E".matches(regex3));
        System.out.println("------匹配[abc]匹配 a 或 b 或 c 中任意一个字符-------");
        String regex4 = "[abc]";
        System.out.println("b".matches(regex4));
        System.out.println("d".matches(regex4));
        System.out.println("a".matches(regex4));
        System.out.println("ab".matches(regex4));
        System.out.println("------匹配[^abc]匹配除了 a 或 b 或 c 之外的任意一个字符-------");

        String regex5 = "[^abc]";
        System.out.println("e".matches(regex5));
        System.out.println("f".matches(regex5));
        System.out.println("你".matches(regex5));
        System.out.println("a".matches(regex5));
        String regex6 = "[changsha]";//匹配 changsha 这些字符中的任意一个字符
       System.out.println("z".matches(regex6));
        System.out.println("a".matches(regex6));
        System.out.println("------匹配[a-zA-Z]中任意一个字符-------");

        String regex7 = "[a-zA-Z]";
        System.out.println("b".matches(regex7));
        System.out.println("T".matches(regex7));
        System.out.println("9".matches(regex7));
        System.out.println("------匹配[0-9]中任意一个字符-------");
        String regex8 = "[0-9]";
        System.out.println("4".matches(regex8));
        System.out.println("2".matches(regex8));
        System.out.println("10".matches(regex8));

    }

}
```

注意:匹配的都是单个字符

●14.04_常见对象(数量词)

- A:Greedy 数量词
  - X? ： X 出现一次或零次
  - X* ： X 出现零次或多次
  - X+ ： X 出现一次或多次
  - X{n} ： X 出现恰好 n 次
  - X{n,} ：X 出现至少 n 次
  - X{n,m} ： X 出现至少 n 次，但是不超过 m 次

```
package com.sxt.homework;
public class Demo {
    public static void main(String[] args) {

      /*
          X 代表正则表达式
            ▪ X? ：X 出现一次或零次
            ▪ X* ： X 出现零次或多次
            ▪ X+ ： X 出现一次或多次
            ▪ X{n} ： X 出现恰好 n 次
            ▪ X{n,} ：X 出现至少 n 次
            ▪ X{n,m} ： X 出现至少 n 次，但是不超过 m 次
       */
        System.out.println("--------?--------");
        String regex1 = "[abc]?";//表示a 或b 或c 中任意一个出现一次或0 次
      System.out.println("a".matches(regex1));
        System.out.println("c".matches(regex1));
        System.out.println("d".matches(regex1));
        System.out.println("-------*-----------");
        String regex2 = "[^abc]*";//除了abc 以外的任意字符出现0 次或多次
      System.out.println("defe".matches(regex2));
        String regex3 = "[0-9]*";//0~9 之间的任意一个数出现0 次或多次
      System.out.println("324234".matches(regex3));
        System.out.println("ab".matches(regex3));
        System.out.println("-----+------");
        String regex4 = "[0-9]+";//0~9 之间的任意一个数出现,至少出现一次
      System.out.println("0989ad".matches(regex4));
        System.out.println("0989".matches(regex4));
        System.out.println("------{n}-------");
        String regex5 = "[a-zA-z]{4}";//表示a-zA-z 之间的字母出现正好四次
      System.out.println("abcd".matches(regex5));
        System.out.println("abcde".matches(regex5));
        System.out.println("abc1".matches(regex5));

        System.out.println("-----{n,}-------");
        String regex6 = "[nihao]{2,}";//表示nihao 中的任意一个字母最少出现2 次,最多无限次
      System.out.println("nihaowo hen hao ".matches(regex6));
        System.out.println("ninihaha".matches(regex6));
```

```
        System.out.println("-----{n,m}-------");
        String regex7 = "[ilovey]{2,5}";//表示 ilovey 中的任意一个字母最少出现 2 次,最多 5 次
       System.out.println("iiloveloveyou".matches(regex7));//false
        System.out.println("iiloveloveyoyo".matches(regex7));//false
        System.out.println("iilll".matches(regex7));//true
        System.out.println("iillli".matches(regex7));//false
        System.out.println("ilove".matches(regex7));//true
    }
}
```

●常见对象(正则表达式的分割功能)

- A:正则表达式的分割功能
  - String 类的功能： public String[] split(String regex)
- B:案例演示
  - 正则表达式的分割功能

```
String s = "我 a 爱 2 大 B 长 9 沙";
        //要求输出:我爱大长沙
```

```
package com.sxt.homework;
public class Demo {
    public static void main(String[] args) {

        String s = "我 a 爱 2 大 B 长 9 沙";
        //要求输出:我爱大长沙
       //方法一:
         String[]arr  =s.split("\\w");//     \\w 表示匹配 a-zA-Z0-9 之间的任意字符
        for (int i = 0; i <arr.length ; i++) {
            System.out.print(arr[i]);
        }
        //方法二:
         System.out.println();
        String newS = s.replaceAll("\\w","");
        System.out.println(newS);

    }

}
```

●常见对象(把给定字符串中的数字排序)

- A:案例演示
  - 需求：我有如下一个字符串:”91 27 46 38 50”， 请写代码实现最终输出结果是：”27 38 46 50 91”

```java
package com.sxt.homework;


import java.util.Arrays;

public class Demo {
    public static void main(String[] args) {
        String s = " 91 27 46 38 50";
        String s1 = s.trim();
        String[]arr = s1.split(" ");
        Arrays.sort(arr);
        for (String str:arr) {
            System.out.print(str+" ");
        }

    }

}
```

●常见对象(正则表达式的替换功能)

- A:正则表达式的替换功能
  - String 类的功能： public String replaceAll(String regex,String replacement)
- B:案例演示
  - 正则表达式的替换功能

  需求：用正则表达式去掉字符串"我爱 23 长沙 576ab 星 ec 城"里面的数字和字母

- B:案例演示

  a:切割

  需求：请按照叠词切割： "sdqqfgkkkhjppppkl"; 结果为: sd fg hj kl

```
```

  b:替换

  需求：我我....我...我.要...要要...要学....学学..学.编..编编.编.程.程.程..程

  将字符串还原成:“我要学编程”。

```
```