Week 3

Monday, August 31, 2020 9:27 PM

Java中BigInteger方法总结
https://www.jianshu.com/p/8b89ab19db84

Class BigInteger
https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/math/BigInteger.html#%3Cinit%3E(int,byte%5B%5D)

hw 2.2 "70" + "07" == "77"，07前面的0必须保留

double 和float 不能用于switch的原因：
1. 精度问题；
2. 这两个类型占用的内存地址太多，不好用JUMP指令跳过；

>>> unsigned right shift - 不会保留最高位（符号位）

Java一般错的的执行顺序：left -> right

只有三元表达式（? :）是right -> left

P86 和 P85的习题，以及scanner的example

Class Scanner
https://www.cs.rit.edu/~hpb/Jdk5/api/java/util/Scanner.html

Class Pattern
https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/regex/Pattern.html

Java 格式化输出 printf 例子
https://www.runoob.com/w3cnote/java-printf-formate-demo.html

Java 正则表达式
https://www.runoob.com/java/java-regular-expressions.html

浅谈Java中try catch 的用法
https://blog.csdn.net/qq_34427165/article/details/83929470

Java中直接输出一个类的对象
https://www.cnblogs.com/tech-bird/p/3525266.html

Java中的static关键字解析
https://www.cnblogs.com/dolphin0520/p/3799052.html

java中 static,final,transient,volatile关键字的作用
https://blog.csdn.net/weixin_40205234/article/details/86630303

浅析Java中的final关键字
https://www.cnblogs.com/dolphin0520/p/3736238.html

volatile关键字解析
https://www.cnblogs.com/dolphin0520/p/3920373.htm

Java中transient关键字的详细总结
https://blog.csdn.net/u012723673/article/details/80699029

regular-expressions.info
https://www.regular-expressions.info/unicode.html#prop

JAVA静态导入(import static)详解
https://www.iteye.com/blog/xiangtui-1010218

```
class X_2
{
  public static void main(String args[])
  {
    System.out.println("I like to play " + 6  + 2 );
    System.out.println("I like to play " + 6  * 2 ); // 乘法优先于加法，output："I like to play 12"
    System.out.println("I like to play " + ( 6  + 2 ) );
  }
}
```

Modifier	Class	Package	Subclass	World
public	Y	Y	Y	Y
protected	Y	Y	Y	N
no modifier	Y	Y	N	N
private	Y	N	N	N

java继承
https://www.cnblogs.com/maopaoer/p/10664884.html

Java 继承（菜鸟教程）
https://www.runoob.com/java/java-inheritance.html

8.8. Constructor Declarations
https://docs.oracle.com/javase/specs/jls/se14/html/jls-8.html#jls-8.8

Class Object
https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/Object.html

Java学习笔记13---如何理解“子类重写父类方法时，返回值若为类类型，则必须与父类返回值类型相同或为其子类”
https://www.cnblogs.com/chanchan/p/7796472.html

java 子类重写父类方法时默认调用子类方法（父类子类加载顺序）
https://blog.csdn.net/weixin_39852602/article/details/105578638

java 父类子类有同名方法时如何调用
https://blog.csdn.net/qq_21808961/article/details/78309180

Java字符串池（String Pool）深度解析
https://www.cnblogs.com/fangfuhai/p/5500065.html

Java 包(package)
https://www.runoob.com/java/java-package.html

Java 中带包（创建及引用）的类的编译与调试
https://www.runoob.com/w3cnote/java-compile-with-package.html

Java 多态
https://www.runoob.com/java/java-polymorphism.html

父类引用指向子类对象，当使用多态方式调用方法时，首先检查父类中是否有该方法，如果没有，则编译错误；如果有，再去调用子类的同名方法。

调用属性，引用谁，调用谁的属性

子类不能引用new 父类：
1.从对象的内存角度来理bai解. 假设现在有一du个父类Father,它里面的变zhi量需要占用1M内存.有一个它dao的子类Son,里面的变量需要占用0.5M内存. 现在通过代码来看看内存的分配情况:
2.f = new Father();//系统将分配1M内存.
https://zhidao.baidu.com/question/941773815405965692.html

Java内部类详解
https://www.cnblogs.com/dolphin0520/p/3811445.html

```
public class  AnOtherExample       {
  int instanceV = 1;
  static AnOtherExample staticAnOtherExample;
  AnOtherExample instanceAnOtherExample;

  public AnOtherExample()  {
        this.instanceAnOtherExample = this;
  }
  public void create() {
        AnOtherExample aAnOtherExample;
        for ( int index = 0; index < 10; index ++ ){
              aAnOtherExample = new AnOtherExample();
        }
  }
  public String toString() {
        return "this/instanceAnOtherExample " + this + "/" + instanceAnOtherExample;
        // return "" + instanceV;
  }

  public static void main(String args[] )     {
        staticAnOtherExample = new AnOtherExample();
        System.out.println(staticAnOtherExample);
        staticAnOtherExample.create();
  }
}
```

How many different instanceV exist?
Only 2, one for staticAnOtherExample, one for aAnOtherExample

```
public class  NotCorrect      {

  static int counter;
  NotCorrect aNotCorrect;  // new NotCorrect(); // hint

  public NotCorrect() {
         // System.out.println("so Many Calls: " + counter++ );
        aNotCorrect = new NotCorrect();
  }
  public static void main(String args[] )     {
        NotCorrect aNotCorrect = new NotCorrect();
  }
}
```

Not compile: StackOverflowError