

今日作业的目标

熟悉匿名内部类和lambda表达式的使用，手敲一敲Object类的方法

完成作业后，需要将md文件转换成PDF格式，并命名为当天的课程名+下划线+自己的名字！压缩后提交！

- 可以通过查看共享目录下，课程资料中**dayXx_Xxx**就是课程名
- 下划线不要弄错了，不能是空格或者横杠
- 下划线后跟自己的名字，不要在名字后面添一些乱七八糟的东西，如pdf后缀名
- 必须压缩后提交，压缩格式不限，rar、7z等等都可以
- 以上格式满足后，就可以提交作业了

提交作业的网址（局域网内网网站）：

<http://192.168.2.100:8080/upload/java/..th>

链接最后的“..th”表示班级的期数，比如你是Java28期学生，这里就填入28th

一般来说，打开这个网站对浏览器种类没有特别的要求，仅建议不要直接使用微信自带浏览器
需要注意的是，如果多次重复提交某一天的作业，必须保持名字不同

建议在“课程名+下划线+自己的名字”的后面加上2，3...之类的数字以示区分

操作题

操作题，无需表现在作业答案中，自己琢磨和练习即可

待补充！

非编程题

简答题，以下简答

- 什么是功能接口
 - 功能接口中只能有一个方法吗
不是，只能有且仅有一个必须要实现的抽象方法，可以有多个默认方法，静态方法，也可以有多个静态方法(这几个静态方法不是必须要被实现)
 - 功能接口中只能有一个抽象方法吗
不是，同上
- 描述一下运行时Class对象
字节码文件加载进内存时，堆内存会跟着创建一个类的运行时对象，也就是Class对象，这个对象封装了被加载的类的一些信息；
由于java中任何一个类都默认继承Object,也就默认继承了对象.getClass()方法，在没有重写覆盖的情况下，子类对象.对象.getClass()会返回一个对象的类对象地址值，并赋给Class类型的引用。
Class类本身有getName(),getSimpleName()等方法，可以得到类的一些信息。
- Object类是什么？
是所有类的根类，上帝类，所有的子类都会默认继承Object，继承他的一些属性方法，在子类没有重写覆盖Object方法的情况下，子类会执Object的。

编程题

编程题的答题要求：

编程题，需要先编写代码，执行调试完毕后
将代码以代码块（CTRL+A贴入整个Java文件内容，而不是一个main方法）的格式贴入md文件
并附上执行结果图片

如何在Typora中插入代码块？

1. 可以直接从idea复制代码，然后粘贴进md文档，Typora会自动转换成代码块的格式
2. 可以在md文档空白处中右键，然后插入代码块，再把代码复制进来（熟练了可以使用快捷键）
3. 代码块右下角可以选择语言，建议直接填入Java（这样做会有颜色标记关键字）

如何在Typora中插入图片？

1. 可以使用微信/QQ/windows/Snipaste截图等截图工具截图到计算机粘贴板，然后直接粘贴到md文档中
2. 可以在md文档空白处中右键，然后插入图像，自己选择本地图片的路径（可以用，但不推荐）

敲一遍老师上课的代码

根据老师在每一个Demo类注释的头部写的问题，逐一敲一遍老师的代码

尤其是那些不知道该怎么下手做作业的同学，一定要认真敲一遍老师代码

- 敲一下lambda表达式的使用

lambda表达式的练习

lambda表达式的书写，除了注意格式外，最重要的是关注类型推断

- 提供以下6个功能接口，请用lambda表达式分别创建对象，调用test()方法
- 自由发挥lambda表达式的书写

```
//无返回值无参数的功能接口
@FunctionalInterface
interface INoReturnNoParam {
    void test();
}

//无返回值有一个参数的功能接口
@FunctionalInterface
interface INoReturnOneParam {
    void test(int a);
}

//无返回值两个参数的功能接口
@FunctionalInterface
interface INoReturnTwoParam {
    void test(int a, int b);
}
```

```
//有返回值无参数的功能接口
@FunctionalInterface
interface IHasReturnNoParam {
    int test();
}

//有返回值一个参数的功能接口
@FunctionalInterface
interface IHasReturnOneParam {
    int method(int a);
}

//有返回值两个参数的功能接口
@FunctionalInterface
interface IHasReturnTwoParam {
    int test(int a, int b);
}
```

今天先学会使用匿名内部类和lambda表达式，具体详细的有哪些用途，后面会碰到

- 如果实在感兴趣想知道，你也可以自己搜一搜，Java**对象数组的排序**

```
package com.cskaoyan.work.work15;

public class Test {
    public static void main(String[] args) {
        Test test = new Test();
        INoReturnNoParam in1 = test::noParam;
        INoReturnOneParam in2 = test::oneParam;
        INoReturnTwoParam in3 = test::twoParam;
        IHasReturnNoParam in4 = test::hasReturnNoParam;
        IHasReturnOneParam in5 = test::hasReturnOneParam;
        IHasReturnTwoParam in6 = test::hasReturnTwoParam;

        in1.test();
        in2.test(2);
        in3.test(1,2);
        in4.test();
        in5.method(5);
        in6.test(5,6);
    }

    public void noParam() {
        System.out.println("这个是无返回值无参的lambda测试");
    }

    public void oneParam(int a) {
        System.out.println("这个是无返回值参数是"+a+"的lambda测试");
    }

    public void twoParam(int a,int b) {
        System.out.println("这个是无返回值参数是"+a+"和"+b+"的lambda测试");
    }

    public int hasReturnNoParam() {
        System.out.println("这个是有返回值无参的lambda测试");
        return 666;
    }
}
```

```

    }

    public int hasReturnnoneParam(int a) {
        System.out.println("这个是有返回值一个参数的lambda测试");
        return a;
    }

    public int hasReturntwoParam(int a,int b) {
        System.out.println("这个是有返回值2个参数的lambda测试");
        return a+b;
    }
}

//无返回值无参数的功能接口
@FunctionalInterface
interface INoReturnNoParam {
    void test();
}

//无返回值有一个参数的功能接口
@FunctionalInterface
interface INoReturnOneParam {
    void test(int a);
}

//无返回值两个参数的功能接口
@FunctionalInterface
interface INoReturnTwoParam {
    void test(int a, int b);
}

//有返回值无参数的功能接口
@FunctionalInterface
interface IHasReturnNoParam {
    int test();
}

//有返回值一个参数的功能接口
@FunctionalInterface
interface IHasReturnOneParam {
    int method(int a);
}

//有返回值两个参数的功能接口
@FunctionalInterface
interface IHasReturnTwoParam {
    int test(int a, int b);
}

```

```
Test (1) x
F:\itstudy\develop\software\jdk1.8\bin\ja
这个是无返回值无参的lambda测试
这个是无返回值参数是2的lambda测试
这个是无返回值参数是1和2的lambda测试
这个是有返回值无参的lambda测试
这个是有返回值一个参数的lambda测试
这个是有返回值2个参数的lambda测试

Process finished with exit code 0
```

lambda表达式的练习

定义一个计算（compute）接口，接口中有加减乘除四个抽象方法。
然后使用匿名内部类去实现加减乘除并测试

上述功能，昨天已经实现了，今天考虑用lambda改进它，如下：

- 做完以上需求后，可以考虑用以下功能接口，实现一个计算器的工具类
- 在工具类中，只需要一个工具方法，就能够实现所有的计算功能

```
@FunctionalInterface
interface Compute {
    double compute(double a, double b);
}
```

提示：

```
//需要提供一个使用功能接口的方法完成需求
class ComputeTool {
    private ComputeTool() {
    }

    public static void calc(Compute com, double a, double b) {
        //...
    }
}
```

```
package com.cskaoyan.work.work15;

public class ComputeTool {

    private ComputeTool() {
    }

    public static void calc(Compute com, double a, double b) {
        //要有四个com对象，分别实现了加减乘除
    }
}
```

```

        System.out.println(com.compute(a, b));
    }
}

class Demo {
    public static void main(String[] args) {
        /**
         ComputeTool.calc((a,b)->a+b, 5, 6);
         ComputeTool.calc((a,b)->a-b, 5, 6);
         ComputeTool.calc((a,b)->a*b, 5, 6);
         ComputeTool.calc((a,b)->a/b, 5, 6);

        */
        ComputeTool.calc((a,b)->a+b, 5, 6);
        ComputeTool.calc(Demo::minus, 5, 6);
        ComputeTool.calc(Demo::multiply, 5, 6);
        ComputeTool.calc(Demo::divide, 5, 6);
    }
}

@FunctionalInterface
interface Compute {
    double compute(double a, double b);
}

```

```

F:\itstudy\develop\software\jdk1.8\bin\
11.0
-1.0
30.0
0.8333333333333334
Process finished with exit code 0

```

getClass()方法练习

定义两个类，然后分别创建对象，调用getClass方法

用“==”号比较它们的运行时Class对象是否相等，并说明原因

理解运行时类对象、类加载、类的对象的区别

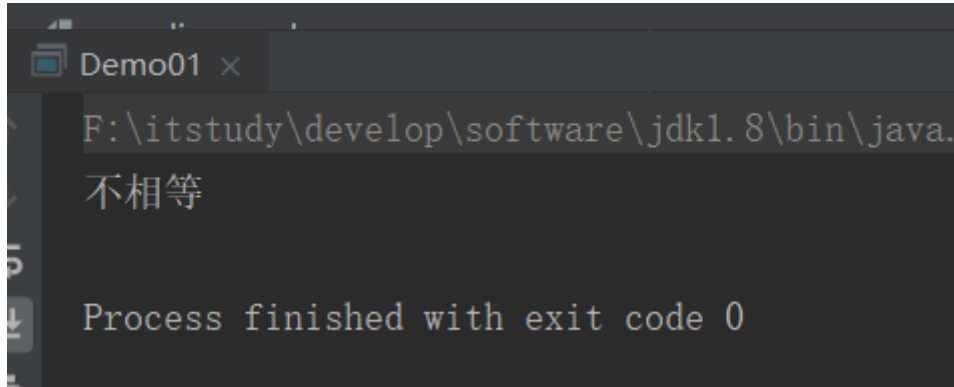
```

package com.cskaoyan.work.work15;

public class Demo01 {
    public static void main(String[] args) {
        A a = new A();
        B b = new B();
        Class aClass = a.getClass();
        Class bClass = b.getClass();
        if (aClass == bClass) {

```

```
        System.out.println("相等");
    }else {
        System.out.println("不相等");
    }
}
}
class A{}
class B{}
```



```
Demo01 x
F:\itstudy\develop\software\jdk1.8\bin\java.
不相等

Process finished with exit code 0
```

周末了，好好休息一下吧，有对象就出去玩一玩，没有就new一个吧

四、预习问题

预习的题目仅为预习提供思路，不用表现在作业中

- 预习Object类的clone()方法，思考以下问题
 - Object类的API自己尝试敲一敲
- 预习String类的
 - 可以自己敲一敲String的常用API