第一题

```
public static void main(String[] args) {
    int test = test(3,5);
    System.out.println(test);//
}
public static int test(int x, int y){
    int result = x;
    try{
        if(x<0 || y<0){
            return 0;
        }
        result = x + y;
        return result;
    }finally{
        result = x - y;
    }
}
```

8

第二题

```
public class Test02 {
    public static void main(String[] args) {
        try{
            return;
        }finally{
            System.out.println("finally");
        }
    }
}
```

finnally

第三题

```
public class Test03 {
    {
        System.out.println("a");
    }
    static{
        System.out.println("b");
    }
    Test03(){
        System.out.println("c");
    }
    public static String getOut(){
        try{
            return "1";
        }catch(Exception e){
            return "2";
```

```
        }finally{
            return "3";
        }
    }
    public static void main(String[] args) {
        System.out.println(getOut());
    }
}
```

b

3

第四题

```
public class Test04 {
    static int i = 0;
    public static void main(String[] args) {
        System.out.println(test());
    }
    public static int test(){
        try{
            return ++i;
        }finally{
            return ++i;
        }
    }
}
```

2

第五题

```
import java.io.IOException;
public class Test05 {
    public static void main(String[] args) {
        int a = -1;
        try{
            if(a>0){
                throw new RuntimeException("");
            }
            else if(a<0){
                throw new IOException("");
            }
            else{
                return ;
            }
        }catch(IOException ioe){
            System.out.println("IOException");
        }catch(Throwable e){
            System.out.println("Throwable");
        }finally{
            System.out.println("finally");
        }
    }
}
```

IOException

finally

第六题

```
public class Test06 {
    public static int fun(){
        int result = 5;
        try{
            result = result / 0;
            return result;
        }catch(Exception e){
            System.out.println("Exception");
            result = -1;
            return result;
        }finally{
            result = 10;
            System.out.println("I am in finally.");
        }
    }
    public static void main(String[] args) {
        int x = fun();
        System.out.println(x);
    }
}
```

Exception

I am in finally.

-1

第七题

```
public static int aMethod(int i)throws Exception{
    try{
        return i / 10;
    }catch(Exception ex){
        throw new Exception("exception in aMethod");
    }finally{
        System.out.println("finally");
    }
}
public static void main(String[] args) {
    try {
        aMethod(0);
    } catch (Exception e) {
        System.out.println("exception in main");
    }
}
```

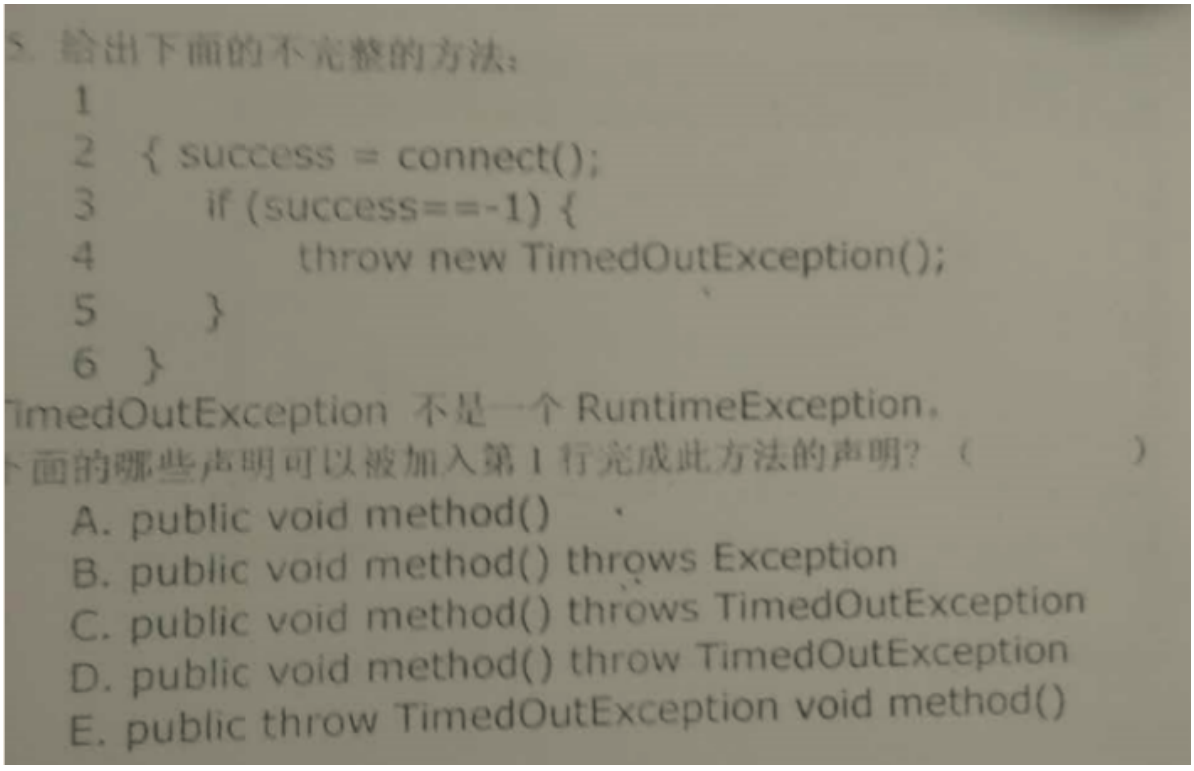
finally

第八题

```
/*
案例：
在一款角色扮演游戏中,每一个人都会有名字和生命值，角色的生命值不能为负数。
要求：当一个人物的生命值为负数的时候需要抛出自定的异常
操作步骤描述：
（1）自定义异常类NoLifeValueException继承RuntimeException
①提供空参和有参构造
②在有参构造中，需要调用父类的有参构造，把异常信息传入
（2）定义Person类
①属性：名称(name)和生命值(lifeValue)
②提供空参构造
③提供有参构造：使用setXxx方法给name和lifeValue赋值
④提供setter和getter方法：
在setLifeValue(int lifeValue)方法中，首先判断，如果 lifeValue为负数,就抛出
NoLifeValueException，异常信息为：生命值不能为负数：xx；
然后在给成员lifeValue赋值。
（3）定义测试类Test08
①使用满参构造方法创建Person对象，生命值传入一个负数
由于一旦遇到异常,后面的代码的将不在执行,所以需要注释掉上面的代码
②使用空参构造创建Person对象
调用setLifeValue(int lifeValue)方法,传入一个正数,运行程序
调用setLifeValue(int lifeValue)方法,传入一个负数,运行程序
③分别对①和②处理异常和不处理异常进行运行看效果
*/

public class NoLifeValueException extends RuntimeException {
    public NoLifeValueException() {
    }

    public NoLifeValueException(String message) {
        super(message);
    }
}

public class Person {
    private String name;
    private int lifeValue;

    public Person(String name, int lifeValue) {
        this.name = name;
        this.lifeValue = lifeValue;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getLifeValue() {
        return lifeValue;
    }

    public Person() {
    }
```

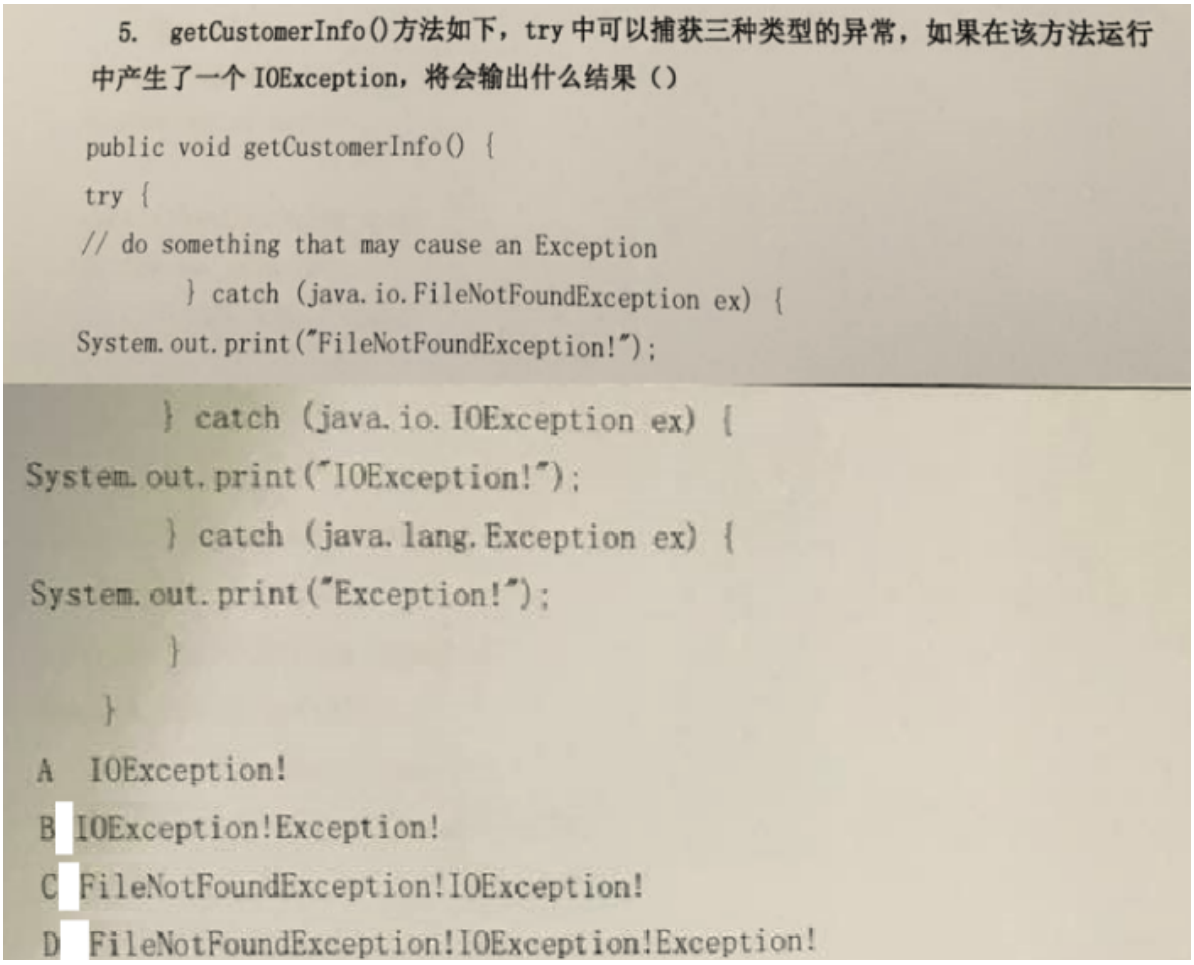
```
    public void setLifeValue(int lifeValue) {
        if(lifeValue < 0){
            throw new NoLifeValueException("生命值不能为负数："+lifeValue);
        }
        this.lifeValue = lifeValue;
    }
}

public class Test08 {
    public static void main(String[] args) {
        //Person person = new Person("jzsj", -2);

        Person person = new Person();
//        person.setLifeValue(20);
//        person.setLifeValue(-10);

        try{
            person.setLifeValue(20);
            person.setLifeValue(-10);
        }catch (NoLifeValueException e){
            e.printStackTrace();
        }
    }
}
```

第十题



B、C

第十一题



A

第十二题

1. 请描述异常的继承体系

```
Throwable
    -- Error
    -- Expection
        -- RuntimeException
        -- 非RuntimeException
```

2. 请描述你对错误(Error)的理解

JVM、硬件等错误，是不能手动处理的，产生时必定终止程序

3. 请描述你对异常(Expection的理解)

程序运行时产生的问题，可以对其进行手动处理，以保证程序继续运行

4. 请描述你对运行时异常(RuntimeException)的理解

由于在程序设计时高频率出现，所以Java对其进行了优化，不需要手动处理，可以自动上抛到JVM

5. throw与throws的区别

throw：用于在程序中主动抛出一个异常，将异常抛给方法本身

throws：用于将方法的方法体出出现的异常上抛，抛给调用者

6. 异常处理方式有几种,分别是什么?详细阐述每种方式对异常是如何处理的

两种：1. 将异常向上抛出　2. 对异常进行捕获

抛出：将异常主动向上抛出，交给上层调用者处理，如果一直不处理，将自动抛给JVM，结束程序并打印错误信息

捕获：通过try...catch...对异常进行捕获与处理，以保证程序的正常运行，并通过finally释放资源

7. 请列举常见异常，并说明产生原因。

```
NullPointerException//操作null对象
ArithmeticException//除0
ArrayIndexOutOfBoundsExceptio//下标越界
```