**JAVA开发试卷**

应聘时间: 应聘岗位: 姓名:

|  |  |
| --- | --- |
| 注意事项 | 1测试时间限时60分钟。  2请在考卷以及答卷上填写所应聘岗位以及姓名  3测试结束后，请将考卷、答卷以及草稿纸一并交回 |

1. **JAVA基础**
2. 对于下面的代码，选择正确的执行结果 。

public class ExceptionTest {

class TestException extends Exception {

}

public void runTest() throws TestException {

System.out.println("runTest");

try {

throw new ExceptionTest.TestException();

} catch (Exception e) {

System.out.println("runTest Exception");

}

}

public static void main(String[] args) {

ExceptionTest test = new ExceptionTest();

test.runTest();

System.out.println("test");

}

}

1. 输出如下内容

runTest

test

1. 输出如下内容

runTest

runTest Exception

test

C. 输出如下内容

runTest

D. 编译不通过

答案 ：

1. 请解释Checked exception和Runtime exception的区别是什么。

　　\* Runtime exceptions:   
　　在定义方法时不需要声明会抛出runtime exception;   
　　在调用这个方法时不需要捕获这个runtime exception;   
　　runtime exception是从java.lang.RuntimeException或java.lang.Error类衍生出来的。   
　　\* Checked exceptions:  
　　定义方法时必须声明所有可能会抛出的checked exception;   
　　在调用这个方法时，必须捕获它的checked exception，不然就得把它的exception传递下去;   
　　checked exception是从java.lang.Exception类衍生出来的。

答案：

1. 判断某年度Y是否为闰年的方法是：
2. 如果Y是4的倍数，且不是100的倍数，则是闰年。
3. 如果Y是100的倍数，则只有当Y是400的倍数时，才是闰年
4. 其它条件下都是平年。

请根据这个规则，完成如下函数（在一行内完成）：

boolean isLeapYear(int year) {

((year ((year % 4 == 0 && year % 100 != 0) || year % 400 == 0); return ;

}

答案：

1. 下面的代码会输出内容是 。

public class X {

public static void main(String[] args) {

String foo = "ABCDE";

foo.substring(3);

foo.concat("XYZ");

System.out.println(foo);

}

}

答案：ABCDE

1. 下面哪个方法放在Sub类中会造成编译错误？

class Super {

public float getNum() { return 3.0f; }

}

public class Sub extends Super{ }

A. public float getNum() { return 4.0f; }

B. public void getNum() {}

C. public void getNum(double d) {}

D. public double getNum(float d) { return 4.0d; }

答案：

1. 根据下面的代码，请回答将会输出的结果是 。

public class Test {

   public static void main(String[] args) {

   StringBuffer a = new StringBuffer("A");

   StringBuffer b = new StringBuffer("B");

   operate(a, b);

   System.out.println(a + "," + b);

}

public static void operate(StringBuffer x, StringBuffer y) {

    x.append(y);

    y = x;

   }

}

答案：AB,B

1. 在方法中定义以下变量，请问以下哪个结果是正确的？

int index = 1;

int foo[] = new int[3];

int bar = foo[index];

int baz = bar + index;

A. baz 的值是 0

B. baz 的值是 1

C. baz 的值是 2

D. 抛出异常

E. 代码编译不通过

答案：A

1. 如下方法，将字符串从指定的编码转成另一种编码。请在空白处填写正确的代码

Public String translate (String str, String fromEncode, String toEncode) {

String tempStr = "";

try {

tempStr = new String( , );

   tempStr = tempStr.trim();

} catch (Exception e) {

   System.err.println(e.getMessage());

}

return tempStr;

}

答案：str.getBytes(fromEncode), toEncode

1. 列举jvm中，年轻区（young）、年老区（tenured）的垃圾回收器，并简要说明(\*)

答案：

1. **Collection**
2. 下面的程序输出是

public class A {

public int hashCode() { return 1; }

public boolean equals(Object b) { return true; }

public static void main(String args[]) {

Set set = new HashSet();

set.add(new A());

set.add(new A());

set.add(new A());

System.out.println(set.size());

}

}

答案：1

1. 有两个Integer类型数组，以下函数将两个数组合并，并且排序。请在空白处填写代码。

Integer[] merge(Integer[] arr1, Integer[] arr2) {

java.util.Collection<Integer> aCollection = new TreeSet<Integer>() ;

aCollection.addAll(Arrays.asList(arr1)) ;

aCollection.addAll(Arrays.asList(arr2)) ;

Integer[] arr3 = new Integer[aCollection.size()];

aCollection.toArray(arr3) ;

return arr3;

}

答案：

1. 如果需要对一个List中的对象频繁进行插入和删除操作，请问应该选择哪个实现类？为什么？(\*)

答案：LinkedList的内部实现是链表，它适合于在链表中间需要频繁进行插入和删除操作

1. **数据库**

员工表：（emp表）

|  |  |  |
| --- | --- | --- |
| empId | empName | empSal（员工薪水） |
| 1 | ABC | 1200 |
| 2 | BCD | 2000 |
| 3 | EFG | 3000 |
| 4 | HIJ | 1700 |

薪水级别表：（sal表）

|  |  |  |
| --- | --- | --- |
| minSal | maxSal | salClass（薪水级别） |
| 1000 | 1500 | 1 |
| 1500 | 2200 | 2 |

1. 请写出sql语句，查询id为1的员工所属的薪水级别。

答案：

1. 请写出sql语句，统计人数大于等于2人的薪水级别，和该级别的汇总人数，不属于任何薪水级别的员工不参与统计。

答案：

1. 请写出查询所有员工姓名和所属级别的sql语句(\*)。

答案：

1. **多线程**
2. 请说说Wait和Sleep的区别

答案：

sleep()方法，属于Thread类中的。而wait()方法，则是属于Object类中的。

sleep()方法导致了程序暂停执行指定的时间，让出cpu该其他线程，但是他的监控状态依然保持者，当指定的时间到了又会自动恢复运行状态。

在调用sleep()方法的过程中，线程不会释放对象锁。

而当调用wait()方法的时候，线程会放弃对象锁，进入等待此对象的等待锁定池，只有针对此对象调用notify()方法后本线程才进入对象锁定池准备

获取对象锁进入运行状态。

1. 请编写2个线程，线程1顺序输出1，3，5，……, 99 等奇数，每个数一行。线程2顺序输出2，4，6……100等偶数，每个数一行。最终的结果要求是输出为自然顺序：1,2,3,4,……99,100。

答案：

**public** **class** ThreadPrint {

**public** **boolean** flag;

**public** **class** OddClass **implements** Runnable {

**public** ThreadPrint tp;

**public** OddClass(ThreadPrint tp) {

**this**.tp = tp;

}

@Override

**public** **void** run() {

**int** i = 1;

**while** (i < 100) {

**synchronized** (tp) {

**if** (!tp.flag) {

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

i += 2;

tp.flag = **true**;

tp.notify();

} **else** {

**try** {

tp.wait();

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

}

}

}

**public** **class** EvenClass **implements** Runnable {

**public** ThreadPrint tp;

**public** EvenClass(ThreadPrint tp) {

**this**.tp = tp;

}

@Override

**public** **void** run() {

**int** i = 2;

**while** (i <= 100)

**synchronized** (tp) {

**if** (tp.flag) {

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

i += 2;

tp.flag = **false**;

tp.notify();

} **else** {

**try** {

tp.wait();

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

}

}

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

ThreadPrint tp = **new** ThreadPrint();

OddClass oddClass = tp.**new** OddClass(tp);

EvenClass evenClass = tp.**new** EvenClass(tp);

**new** Thread(oddClass).start();

**new** Thread(evenClass).start();

}

}

1. 解释什么是死锁，如何定位死锁(\*)

答案：

什么是死锁多个进程在运行过程中因争夺资源而造成的一种僵局，当进程处于这种僵持状态时，若无外力作用，它们都将无法再向前推进。

定位死锁

1 jstack pid

2 Found one Java-level deadlock:

1. **算法/设计**
2. 设计购物车，要求画图，期待写出class和operation，需考虑到以下功能[购物车: 失效性，添加，删除，修改，检查重复，总价]

答案：

1. 用递归实以下求和：现N+（N-1）+(N-2)…..+1,其中N>0

答案： **public** **static** **int** add(**int** i) {

**if** (i <= 0) {

**return** 0;

}

**return** i + *add*(i - 1);

}