0-1 试一试 Java

单选题 第 1 题 1 分

```
Integer a = new Integer(3);
Integer b = 3;
int c = 3;
System.out.println(a == b);
System.out.println(a == c);
```

控制台输出结果是?

A False; True

B False; False

C True; False

D True; False

正确答案：A

答案解析：

针对第四行，a 和 b 是两个不同的 object，在 heap 中指向不同的地址，

“==”判定对象等价性（将在 3.5 节继续讨论）。

这对第五行，a 作为一个 Integer 对象，将会首先被 auto-unboxing 为一个

int，然后再跟 c 比较。针对 int 类型的==，比较的是值。

单选题 第 2 题 1 分

```
String a = "c";
String b = "c";
System.out.println("a and b: " + a == b);
```

控制台输出结果是?

A a and b: True

B a and b: False

C True

D False

正确答案：D

答案解析：

+的优先级比==要高，先计算两个字符串+，然后再跟 b 比较。

单选题 第 3 题 1 分

```
public static void
        main(String args[]) {
                System.out.println(2.00 - 1.10);
}
```

控制台输出结果是?

A 0.9

B 0.90

C 不一定

D 以上都不是

正确答案：D

答案解析：

0.8999999999999999

自己查一下资料了解一下 Java 里如何处理 double 类型的数值

单选题 第 4 题 1 分

```
List<Integer> list = new Arraylist<>();
for (int i = -3; i < 3; i++)
        list.add(i);
for (int i = 0; i < 3; i++)
        list.remove(i);
System.out.println(list);
```

A [-3, -2, -1]

B [-2, 0, 2]

C Throws exception

D 以上都不是

正确答案：B

单选题 第 5 题 1 分

```
String s = " Hello";
s += " World";
s.trim();
System.out,println(s);
```

A `" Hello"`

B `"Hello"`

C `" Hello World"`

D `"Hello World"`

正确答案：C

1.1 软件构造的多维度视图

单选题 第 1 题 1 分

Memory dump 属于软件三维度视图的____

A Build-time 和 code-level view

B Run-time 和 moment view

C Run-time 和 moment view

D Moment 和 component-level view

正确答案：C

单选题 第 2 题 1 分

Execution stack trace 和 code snapshot 在软件三维度视图中的共性特征是

____

A 都是 moment view

B 都是 component-level

C 都是 run-time view

D 无共性

正确答案：C

单选题 第 3 题 1 分

Code Churn 和 AST 分别是____的视图

A Code-level, Run-time

B Build-time, Period

C Component-level, Code-level

D Period, Build-time

正确答案：D

多选题 第 4 题 2 分

Static linking 和 Dynamic linking 的区别在于____

A 前者发生在构造阶段，后者发生在运行阶段

B 前者的软件运行时需要库文件，后者不需要

C 同样的源代码，经过前者产生的代码尺寸比后者的要小

D 二者都是试图把外部库文化和开发者的代码链接在一起形成可执行文件

正确答案：A

多选题 第 6 题 2 分

以下说法正确的是____

A Code static analysis 是发生在 build-time

B Deployment 是把 build-time 的软件转换为 run-time 的软件的手段之一

C Files 随时间发生变化，产生各个不同版本，按时间连起来形成 period view

D 对软件的 profiling 和 tracing 均发生在 run-time

正确答案：ABCD

1.2 软件构造的质量指标

多选题 第 1 题 2 分

以下___是软件构造的 external quality factors?

A Correctness 正确性

B Extendibility 可扩展性

C Reusability 可复用性

D Ease of use 易用性

E Complexity 代码复杂度

F Understandability 代码可理解性

正确答案：ABCD

单选题 第 2 题 1 分

关于软件构造的质量指标，以下说法不正确的是____

A 健壮性刻画了软件能够恰当的处理 spec 范围之外的各类异常情况的能力

B 各项质量指标的优先级是等价的，在软件构造过程中要对它们做出全面优化

C 代码行数 LoC 是内部质量指标之一，但它可能对多项外部质量指标产生影响

D 程序的可复用性与程序的开发代价/运行效率直接存在折中

正确答案：B

多选题 第 3 题 1 分

Correctness 和 Robustness 的区别在于___

A 前者针对“需求”的正确实现，后者针对“需求”之外的其他情况的恰当实现

B 如果某函数输入参数应该是(0,100]范围的整数，当用户输入-1 时，此为前者应考虑的内容。

C 如果某函数的返回值应该是(0,100]范围的整数，当程序输出-1 时，此为后者

应考虑的内容。

D 一个 100%正确的程序一定是 100%健壮，反之亦然。

正确答案：A

单选题 第 4 题 1 分

说法不正确的是___

A LoC 和 code complexity 很高，并不代表一定有很差的 reusability 和 extendibility

B 对代码的时间/空间复杂度进行优化，可能带来其他 external quality factors 的降低

C 每向软件里增加一点功能，都要确保其他质量属性不受到损失

D 健壮性是唯一不能与其他质量指标进行 tradeoff 的质量指标

正确答案：D

多选题 第 5 题 2 分

HIT CS32207 关注的软件构造质量指标包括___

A Ready to change

B Safe from bugs

C Easy to understand

D Efficient to run

E Cheap for develop

F Easy to extend

正确答案：ABCDEF

2.1 软件过程与配置管理

多选题 第 1 题 2 分

以下__不是 agile development 敏捷开发过程的特征

A 线性过程

B 增量式过程

C 迭代过程

D 测试驱动开发（Test-Driven）

E 持续集成、持续交付

F V 字模型（确认/验证）

正确答案：A

单选题 第 2 题 1 分

关于软件配置管理 SCM 的说法，不正确的是___

A 用于追踪和控制软件开发过程中的变化

B 其基本管理单元是软件配置项 SCI，即开发过程中发生变化的基本单元

C 版本是为软件处于特定时刻（moment）的形态指派一个唯一的编号

D Git 是一种典型的集中式版本控制系统

正确答案：D

答案解析：

Git 是分布式版本控制系统

单选题 第 3 题 1 分

以下关于软件配置管理 SCM 和 Git 的说法，不正确的是___

A 软件配置项 SCI 是软件演化过程中发生变化和 SCM 管理变化的基本单元，不需再细分

B Git 中在本地机器上的.git 目录对应于 SCM 中的配置管理数据库 CMDB

C Git 中的 SCI 是“文件”，它有三种形态：已修改(modified)、已暂存(staged)、已提交(committed)

D Git 中两次相邻提交 v1 和 v2，若后者提交时间晚于前者，那么 Git 仓库中只记录 v2 中的文件相对于 v1 中的文件发生变化的代码行（增加和删除的代码行）

正确答案：D

填空题 第 4 题 2 分

用于将 GitHub 上的某个 Git 仓库设置为本地仓库的远程仓库的指令是 [填空 1] 。

用于将当前 staging area 中的文件写入 Git 仓库的指令是 [填空 2]。

只需要填写指令，仓库的具体名字等参数无需输入。

正确答案：

git remote add;git commit

多选题 第 5 题 2 分

针对 Git 仓库的 object graph，以下不正确的说法是__

A 它是一个有向图，边的方向指向产生时间较晚的 commit 节点

B 一个 commit 节点可以有 0 个、1 个、2 个、多个 parent 节点

C 一个 branch（分支）本质上相对于一个指向特定 commit 节点的“指针”

D 可以有两个不同的 branch 指向同一个 commit 节点

E git commit 指令相当于在 object graph 当前分支 HEAD 指向的 commit 基础上，派生出一个新的 commit 节点。

正确答案：AB

多选题 第 6 题 2 分

针对 Git 中 commit 节点的数据结构，说法不正确的是____

A 包含一个 tree，tree 中包含了一组指针，分别指向本次 commit 所修改的每一个文件。

B 若某 commit 相比其 parent 来说，某文件 f 未发生变化，则 f 在.git 中不会重复存储。

C 如果文件 f 在前后两个 commit 中相比只增加了一行代码，那么.git 只需要存储 f 的这一行的变化代码即可。

D 如果某个 commit 节点仅存在于远程服务器的 object graph，那么当本地向远程 git push 的时候，会出现错误提示。

正确答案：AC

答案解析：

选项 A 比较隐藏：tree 中包含的指针指向了本次 commit 中包含的所有文件，而非仅仅本次修改的文件。即使某文件未被修改，它也有一个对应的指针，只不过指向的就是其 parent 的对应文件。这其实就是选修 B 所描述的内容。

D 选项相信大家在 Lab1 中已经有过体验了：当远程仓库比较“新”的时候，本地仓库不能直接 push，会出错，必须先 fetch 下来，再跟本地的更新 merge 起来，统一 push 到远程仓库。

单选题 第 7 题 1 分

如果使用 https://xx.yy/zz.git 作为本地 Git 仓库的远程仓库，其别名叫 origin，若本地 git 仓库当前正在 master 分支上工作，那么以下能够最恰当的将本地仓库中的 master 分支的最新提交推送至远程仓库的指令是___

A git push origin master

B git pull origin master

C git fetch origin master;

git merge;

git push origin master

D git commit -m “fix bugs”;

git pull origin master

正确答案：C

答案解析：

这就是上一题 D 选项所描述的现象

2.2 软件构造工具

单选题 第 1 题 1 分

以下___用于在软件设计阶段描述设计思想和设计结果?

A Programming languages (e.g., C, C++, Java, Python)

B Modeling languages (e.g., UML)

C Configuration languages (e.g., JSON)

D Build languages (e.g., XML)

正确答案：B

单选题 第 2 题 1 分

以下关于软件构造过程各阶段的说法，不正确的有____

A Profiling 是 static code analysis 的一种典型形式

B Code review 的目的是发现代码中的潜在错误

C Refactoring 是在不改变代码功能的前提下重写代码，以消除 bug，提高质量

D Build 是将软件从开发态转化为可运行状态的过程

正确答案：A

单选题 第 3 题 1 分

以下___环节无需执行正在开发的软件?

A Code review

B Dynamic code analysis

C Debug

D Testing

正确答案：A

单选题 第 4 题 1 分

Dynamic code analysis / profiling 解决不了的问题是___

A 发现程序运行过程中的内存分配和占用情况

B 发现程序运行过程中每个类被实例化的数目，及其所占用的内存

C 发现程序潜在的性能瓶颈

D 发现程序中潜在的重复代码以便于抽取出来形成可复用函数/类

正确答案：D

多选题 第 5 题 2 分

以下___可纳入自动化 build 的过程?

A Compiling .java into .class

B Executing JUnit test cases

C Using Checkstyle tool to check if code follows Google’ s Java code style

D Packaging .class files into .jar file and deploying it to a remote server

正确答案：ABCD

多选题 第 6 题 1 分

以下说法，不正确的是___

A 常规的构造次序是：coding ->refactoring-> testing -> code review-> debugging ->build ->dynamic profiling

B 通过 code review 和 profiling 找出可能的 bug，通过 testing 找出真实的 bug，通过 debug 找出 bug 的根源

C 想嗯弘毅 spec 构造完备的测试用例，后续对代码的任何修改，都应重新运行测试用例

D Build 脚本是由配置语言书写，告知 build 工具如何一步一步完成自动化 build 任务

正确答案：A

7.5 Testing

多选题 第 1 题 2 分

你要为某个方法 A m(int b, String c)构造黑盒测试用例，那么设计和实现 JUnit 测试用例不需要依据的内容为___

A m()的 pre-condition（注：pre-condition 是该方法输入参数应满足的条件）

B m()的 post-condition（注：post-condition 是该方法执行后返回值应满足的条件）

C m()的内部实现代码

D 类 A 的等价性判断方法 A.equals()

正确答案：C

多选题 第 2 题 2 分

关于测试的说法，不正确的是___

A 再好的测试也无法证明代码里 100%不存在 bug

B 测试用例的数量越多，则越容易发现潜在的 bug

C 设计测试用例时，需要给出输入数据和期望的输出结果

D 测试用例具有优先级，应将最容易发现错误的用例先执行

E TDD 的思想是：先设计方法的 spec，然后根据 spec 设计测试用例，然后再写代码并确保代码通过测试

正确答案：B

多选题 第 3 题 2 分

Which of the following are good time to re-run all your JUnit tests?

A Before doing `git add/commit/push`

B After rewriting a function to make it faster

C When using a code coverage tool such as EclEmma

D After you think you fixed a bug

正确答案：ABCD

单选题 第 4 题 1 分

在使用等价类划分方法进行测试用例设计时，错误的是___

A 如果输入参数 a 在 spec 中仅被规定 a>10 且未说明 a<=10 应该如何，那么无需测试 a<=10 的情况

B 如果输入参数 a 在 spec 中仅被规定 a>10，那么仍然需要为 a<=10 的情况设计测试用例

C 采用笛卡尔积“全覆盖”策略进行测试用例设计，会导致用例数量多，测试代价高

D 采用“覆盖每个取值”的策略进行用例设计，测试代价低，但测试覆盖度可能较差

正确答案：B

多选题 第 5 题 1 分

Which of these techniques are useful for choosing test cases in test-first programming, before any code is written?

A Black box

B Regression

C Static typing

D Partitioning 等价类划分

E Boundaries 边界值

F White box

G Code coverage 代码覆盖度

正确答案：ADE

多选题 第 6 题 2 分

以下说法不正确的是___

A 如果发现了一个新 bug，需要返回到版本仓库中对之前的各个版本进行测试，已确认该 bug 最早是在哪个历史版本中引入的

B 如果某个 bug 已被正确修复并已通过测试，那么为了降低后续测试的代价，应将该 bug 对应的测试用例从测试库中删除

C 代码覆盖度 code coverage 是指所有测试用例执行后有多大百分比覆盖了被测程序的所有代码行

D 可以从被测代码中寻找依据来设计处于“边界”上的测试用例。

正确答案：B

单选题 第 7 题 1 分

以下关于 JUnit 的说法，不正确的是___

A 眸仿佛前标注着`@Test`，意味着它是一个测试方法。`@Test`是 Java 中的`annotation`

B 如果未通过测试，方法中的`assertXXX()`将抛出`AssertionError`

C 一个 Java 测试类可以定义全局属性并在`@Before`方法中对属性进行数据准备，在`@Test`方法中使用数据

D 如果一个 Java 测试类定义了多个`@Test`方法，那么它们按照在代码中出现的先后次序加以执行。

正确答案：D

答案解析：

D 这个选项是一个很有意思的问题，请查阅 Junit 相关资料了解一下在同一个 Java 测试类中多个@Test 方法的执行次序。

另外，@Before 和@After 方法与各个@Test 方法的执行次序如何，也请了解一下。

3.1(a) Data Type and Type Checking

多选题 第 1 题 2 分

Java 中的 Primitive Type（`int, char, boolean`等）和 Object Type（`String`, `Boolean`, `Calendar`等）的差异是___

A 前者在 stack 中分配内存，后者在 heap 中分配内存

B 使用前者的时空代价低，使用后者的时空代价高

C 前者是 immutable 的，后者是 mutable 的

D 前者和后者中的某些类型可通过 auto-boxing 进行自动转换，例如`int`和`Integer`

正确答案：ABD

单选题 第 2 题 1 分

Static type checking 和 dynamic type checking 的区别，不正确的是___

A 前者在编译阶段发生，后者在运行阶段发生

B 前者比后者更能带来程序的健壮性，因为可以在程序投入运行前就发现错误

C 前者能发现数组访问越界的错误，后者能发现 divide by 0 的错误

D 前者是关于“类型合法性”的检查，后者是关于“值的合法性”的检查

正确答案：C

多选题 第 3 题 2 分

以下代码段中，会在运行阶段发生错误的是___

A

```
int[] arr = new int[] {1,2};
arr[2] = 3;
```

B

```
int[] arr = new int[] {1.2};
arr[0] = "3";
```

C

```
String s = null;
System.out.println(s == null);
```

D

```
String s = null;
System.out.println(s.length());
```

正确答案：AD

单选题 第 4 题 1 分

以下__不是 static type checking 所能处理的编程错误

A 所调用函数的参数数目错误

B 调用一个指向 null 的对象的某个方法

C 函数的 return 语句返回的变量类型与函数声明中的返回值类型不匹配

D 赋值语句右侧的值类型与左侧的变量类型不匹配

正确答案：B

多选题 第 5 题 2 分

___不具备 static typing (即 static type checking)的能力

A Java

B C++

C Python

D Ruby

E Perl

F PHP

正确答案：CDEF

填空题 第 6 题 2 分

```
String a = "5" + 6;
System.out.println(a);
```

会在控制台打印出什么？ [填空 1]

可填写“编译错误”、“运行错误”，或具体输出的字符串，无需带引号。

```
int a = "5" + 6;
System.out.println(a);
```

此时的打印输出呢？ [填空 2]

正确答案：

56;编译错误

3-1(b) Mutability and Immutability

单选题 第 1 题 1 分

关于 mutable 和 immutable 的说法，不正确的是__

A 所有简单数据类型和所有相对应的封装类（Integer, Double, Boolean 等）都是 immutable 的

B 所有数组都是 mutable 的

C 使用 immutable 类型可以降低程序蕴含 bug 的风险，但其时空性能相对较差

D 一个 immutable 的类，一旦其 constructor 方法执行结束并产生了类的实例，则该实例的任何成员变量都不能够再被做任何修改

正确答案：D

单选题 第 2 题 1 分

针对 `final` 关键字，说法不正确的是___

A A `final` class declaration means it cannot be inherited.

B A `final` variable means it always contains the same value/reference but cannot be changed

C A `final` method means it cannot be overridden by subclasses

D The contents in a `final Arraylist` cannot be changed during its lifecycle.

正确答案：D

填空题 第 3 题 4 分

```
String a = "a";
String c = a;
a += "b";
c += "c";

StringBuilder b = new StringBuilder(a);
StringBuilder d = b;
b.append("b");
d.append("c");
```

假设执行之后未进行任何垃圾回收，此时内存里共有 [填空 1] 个 `String` 对象和 [填空 2] 个 `StringBuilder` 对象

此时 `c` 的取值是 [填空 3]， `d` 的取值是 [填空 4]

正确答案：

3;1;ac;abbc

多选题 第 4 题 2 分

关于 immutable 和 mutable data type 的说法，正确的__

A 使用不可变类型，对其频繁修改会产生大量的临时拷贝

B 可变类型可直接修改其值而无需太多拷贝，从而提高效率

C 不可辩数据类型更“安全”，因为其值无法修改

D 使用可变类型做“全局变量”，可在不同模块之间高效率的进行共享数据读写

正确答案：ABCD

多选题 第 5 题 2 分

```
final List<String> l1 = new ArrayList<>();
List<String> l2 = new ArrayList<>();
1.      l1.add("a");
2.      l1.set(0, "b");
3.      l1 = l2;
4.      l2 = l1;
```

在上面标号 `1-4` 的四行中，无法通过 `static type checking` 的是____

A 1

B 2

C 3

D 4

正确答案：C

填空题 第 6 题 3 分

```
List<String> k = new ArrayList<>();
k.add("lab1 ends");

Irerator it = k.iterator();
System.out.println(it.hasNext());

it.next();
System.out.println(it.hasNext());

k.remove(0);
System.out.println(it.hasNext());
```

分别输出的是 [填空 1] 、 [填空 2] 、 [填空 3]

填写 `true` 或者 `false` 即可。

正确答案：

TRUE;FALSE;TRUE

答案解析：

第三个填空，输出 TRUE，这让人很费解，试着查阅资料了解一下原因。

单选题 第 7 题 1 分

```
List<String> t = new ArrayList<>();
t.addAll(Array.asList("a","b"));
Iterator<String> i = t.titerator();
while(i.hasNext())
        if(i.next() == "a")
                i.remove();
```

期望结果是 t 中只包含`"b"`。

以下说法正确的是____

A Static type checking 未通过

B 执行时出现 dunamic error，抛出异常

C 正常执行，但结果与期望不一致

D 正常执行，结果与期望一致

正确答案：D

3.2 Specification

多选题 第 1 题 2 分

两个方法具有“行为等价性(behavior equivalence)”，以下说法正确的是

____

A 站在客户端的视角看，它们实现相同的功能

B 站在客户端的角度看，它们可能展现出不同的性能

C 它们具有相同的规约(spec)

D 它们具有相同的实现算法和异常处理策略

E 其实是针对同一个 spec 来说是等价的。若对这个 spec 进行更改，这两个方法也许就不等价了

正确答案：ABCE

单选题 第 2 题 1 分

以下关于方法 spec 的说法，不恰当的是__

A 程序员针对给定的 spec 写代码，需做到“若前置条件满足，则后置条件必须要满足”

B 前置条件是对 client 端的约束，后置条件是对开发者的约束

C 若客户端传递进来的参数不满足前置条件，则方法可直接退出或随意返回一个结果

D 一个方法的前置条件比另一个方法的前置条件更强，二者的后置条件相同，则前者 spec 的强度比后者大

正确答案：D

填空题 第 3 题 3 分

在 Java 语法中，使用 [填空 1] （annotation）表达一个方法的 pre-condition，使用 [填空 2] 和 [填空 3] （annotation）表达方法的 post-condition。

正确答案：

@param;@return;@throws

多选题 第 4 题 2 分

以下说法正确的是___

A 除非在 post-condition 中明确声明过，否则方法内部代码不应该改变(mutate)输入参数

B 方法的 spec 描述里不能使用内部代码中的局部变量或该方法所在类的 private 属性

C 若在方法的 post-condition 中声明“client 端不能修改该方法所返回的数量”，可以减少该方法的潜在 bug

D 若为某方法设计 JUnit test case，在任何 test case 中对该方法的调用必须遵循其 pre-condition

正确答案：ABD

多选题 第 5 题 2 分

如果修改了某个方法的 spec 使之变弱了，那么可能发生的是__

A Client 调用该方法的代价变大了，即 client 需要对调用时传入该方法的参数做更多的检查

B 程序员实现该 spec 的难度增大了，自由度降低了

C 如果用椭圆面积表示 spec 的强度，那么该方法的椭圆面积增大了

D 该 spec 的实现方式更多了

正确答案：ACD

单选题 第 6 题 1 分

这两个 Spec 的强度有何关系?

```
static int findOneOrMore,FirstIndex(int[] a, int val)
 requires: val occurs at least once in a
 effects: returns lowest index i such that a[i] = val
static int findCanBeMissing(int[] a, int val)
 requires: nothing
 effects: returns index i such that a[i] = val, or -1 if no such i
```

A 前者强于后者

B 前者弱于后者

C 二者相同

D Not compatible

正确答案：D

3.3 Abstract Data Type (ADT)

填空题 第 1 题 4 分

类 `WordList` 有四个方法，根据其方法定义来确定其类型

```
1. public WordList(List<String> words)
2. public void unique()
3. public WordList getCapitalized()
4. public Map<String, Integer> getFrequencies()
```

使用 C、M、P、O 分别表示 Creator, Mutator, Producer, Observer

1 [填空 1] 2 [填空 2]

3 [填空 3] 4 [填空 4]

正确答案：

C;M;P;O

单选题 第 2 题 1 分

以下关于 ADT 的 RI 和 AF 的说法，不正确的是___

A ADT 的 Abstract 空间(A)中的某个值，在其 Rep 空间®中可能有多个值与其对应

B 若 ADT 的某个方法返回一个 mutable 的对象，并不一定表明该 ADT 产生了表示泄露

C 若 ADT 的任意 costructor 所构造出的 object 都满足 RI、每个 mutator 方法执行结束后都保持 RI 为真，那么该 ADT 的 RI 就始终为真

D 一个 immutable 的 ADT，其 rep 可以是 mutable 的

正确答案：C

答案解析：

针对 B 选项：通过 defensive copy 返回 mutable 对象，其实并未产生表示泄露

针对 C 选项：如果存在表示泄露，外部 client 就可以修改内部 rep 的值，就可能导致 RI 违反。

针对 D 选项：of course，只要没有 rep exposure 就没有问题。

多选题 第 3 题 2 分

关于 invariants、AF 和 RI，说法不正确的是__

A 如果一个 immutable 的 ADT 存在 rep exposure，那么就违反了该 ADT 的 invariants

B 如果在一个 mutator 方法内没有 checkRep()，那么 RI 就可能被违反了

C 两个 ADT 有相同的 rep 和相同的 AF，那么其 RI 一定相同

D 两个 ADT 有相同的 rep 和相同的 RI，但可能 AF 不同

正确答案：C

答案解析：

针对 A 选项：immutability 也是 invariants 的一部分，产生表示泄露就可能导致内部 rep 被修改，于是就不能保证 immutable 了，invariants 就被违反了。

多选题 第 4 题 2 分

用于检查 ADT 的 Rep Invariants 是否保持为真的 `checkRep()` 最好应该以下__类型的方法结束前调用。

A Mutator

B Creator

C Producer

D Observer

正确答案：ABCD

答案解析：

针对 D 选项：虽然 observer 方法不改变 rep，但还是强烈建议 return 之前要 checkRep()

多选题 第 5 题 2 分

```
class C {
        private String s;
        private String t;
        ...
}
```

它的可能的 Rep Invariants 会是__

A s contains only letters

B s.length() == t.length()

C s is the reverse of t

D s represents a set of characters

正确答案：ABC

多选题 第 6 题 2 分

ADT 的某个方法的 spec 需要以注释的形式放在代码中，在撰写这部分 spec 的时候，不能用到的信息包括__

A Parameters of operation

B Data type of return values of the operation

C Exceptions thrown by the operation

D Mathematical values in the Rep space ®

E Mathematical values in the Abstract space (A)

正确答案：D

答案解析：

Spec 要给 client 看，那么所有内部的东西都不能用。R 是 rep 的值空间，只能开发者自己了解。

多选题 第 7 题 2 分

在对 ADT 的方法进行 JUnit 测试时，以下说法正确的是____

A 如果某方法的返回值为 void，则无法为其撰写测试用例，因为无法 `assertEquals()`

B 对 constructor 方法，测试用例中需要在构造新对象之后调用 observer 方法确认构造结果是否正确

C 对 mutator 方法，测试用例中需要在该方法执行之后调用 producer 方法确认是否做了正确的 mutate

D 对 observer 方法，测试用例中需要使用其他三类方法构造一个对象，再执行该方法并判断结果是否正确

正确答案：BD

多选题 第 8 题 2 分

以下说法不正确的是__

A 只要有 public 的 field，就一定有表示泄露

B 只要有 非 final 的 field，就一定产生表示泄露

C 除了初始化，Immutable 的类中一定不能存在其他任何改变 rep 的方法

D `checkRep()`方法可能消耗大量计算，在程序投入实际运行的时候要注释掉

正确答案：BC

答案解析：

针对 A 选项：sure，所以迫不得已千万不要用 public

针对 B 选项：final 是来支持 immutable 的，与是否存在表示泄露无直接关系

针对 C 选项：可以有 beneficent mutation

针对 D 选项：assert 语句在投入真正运行的程序中是没有意义的，需要注释掉。当然，前提是开发者利用 checkRep()已经发现了所有违反 RI 的 bug 并修复了。

多选题 第 9 题 3 分

针对你设计的一个 ADT，不应该提供给 client 看的内容包括__

A AF

B RI

C Rep exposure safety argument

D Spec

E Testing strategy

F Rep

G Implementation

H Test cases

正确答案：ABCEFGH

3.4 Object-Oriented Programming (OOP)

多选题 第 1 题 2 分

关于 `static` 和 `final` 的说法，正确的是____

A 一个变量被声明为 `final`，意味着它在被首次赋值之后，其内容就不能再变化

B static 类型的方法，调用时无需创建类的实例对象，可直接通过类名使用

C 被声明为 final 类型的类，无法从中派生出子类

D 被声明为 final 类型的方法，无法在子类中被 override

E 类 A 的 static 方法中不能直接调用 A 的 instance 方法（而是要 new 一个 A 的对象再调用）；A 的 instance 方法中可以直接调用 A 的 static 方法

正确答案：BCDE

多选题 第 2 题 2 分

关于 Java interface 的说法，正确的是__

A 不能有 `static` 方法

B 不能有 `constructor`（构造器）

C 不能有 `final` 方法

D 不能有 `private` 方法

E 不能有 `fields`（属性）

正确答案：BCD

答案解析：

E 选项：可以有属性的，都是 public static final 的。

多选题 第 3 题 2 分

关于 class 和 interface 的说法，不正确的是___

A 一个接口可以 extends 一个或多个其他接口

B 一个类只能 implements 一个接口

C 一个类不能同时 extends 另一个类和 implements 一个接口

D 一个类 implements 了一个接口，意味着它必须要实现该接口中的所有方法

E 一个类除了实现其 implements 的接口中的方法，还可以增加新的方法

正确答案：BC

单选题 第 4 题 1 分

某方法的定义是 `public int getLength (List<String> list, boolean bFliter)`，以下___不是对该方法的合法重载

A `private int getLength(List<String> list, String regex)`

B `public Integer getLength(List<String> list)`

C `public int getLength(List<String> list) throws IOException`

D `public void getLength(List<Object> list, boolean bFliter)`

正确答案：D

多选题 第 5 题 2 分

关于 `Java OOP` 中 `override` 和 `overload` 的异同，以下说法不正确的是____

A 前者的参数列表不能改变，后者的参数列表必须发生变化

B 前者的返回值类型不可变化，后者的返回值类型可以变化

C 前者抛出的异常可以变化，后者抛出的异常不能变化

D 前者的类型检查发生在 run-time，后者的类型检查发生在 compile-time

E 在子类里 override 父类方法时，方法前面必须使用`@override`；overload 父类方法时，无需使用 annotation

正确答案：CE

答案解析：

E 选项：@override 不是必需的，但是强烈建议使用，可以帮助开发者理解方法的“源头”

多选题 第 6 题 2 分

```
class Car {
        public String refuel()
        { return "R"; }
}
class Tesla extends Car {
        public String refuel() {return "C";}
        public String refuel(double price)
        { return "P"; }
}
```

无法通过 static type checking 的是__

A

```
Car c = new Car();
c.refuel(10);
```

B

```
Car c = new Tesla();
c.refuel();
```

C

```
Car c = new Tesla();
c.refuel(10);
```

D

```
Tesla t = Tesla();
t.refuel(10);
```

正确答案：AC

答案解析：

Overload 由 static type checking 负责

多选题 第 7 题 2 分

```
class Car {
        public String refuel()
        { return "R"; }
}
class Tesla extends Car {
        public String refuel() {return "C";}
        public String refuel(double price)
        { return "P"; }
}
```

能获取内容为“C”的字符串是__

A

```
Car c = new Car();
c.refuel();
```

B

```
Car c = new Tesla();
c.refuel();
```

C

```
Car c = new Car();
((Tesla) c).refuel();
```

D

```
Tesla t = Tesla();
t.refuel();
```

正确答案：BD

答案解析：

Override 在 run-time 做 dynamic checking

单选题 第 8 题 1 分

类 A 和 B，B extends A，二者分别有一个 apply(A a)方法，具有不同的返回值类型。

假如

```
A a = new A();
B b = new B();
```

那么以下正确的是__

A `b.apply(a)`与`((B) b).apply(a)`不等价

B `b.apply(a)`与`((A) b).apply(a)`不等价

C `b.apply(a)`与`((B) b).apply((B) a)`不等价

D `b.apply(a)`与`((B) b).apply((A) a)`不等价

正确答案：C

答案解析：

选项 A：将 b 强转为(B)b，等于没转，b 的类型本来就是 B

选项 D：也是类似的，两个强转都是没转

选项 B：B 是 A 的子类，所以 b 必然是 A 的实例，所以将 b 转成 A(b)，本质上还是等价的。

选项 C：将 a 转(B) a，这个乡下强转做不到，该强转将会在运行时抛出异常 ClassCastException。

单选题 第 9 题 1 分

关于 OOP polymorphism（多态）的各选项中，___不是同义词

A Ad hoc polymorphism 和 function overloading

B Parametric polymorphism 和 generics（泛型）

C Subtype polymorphism 和 inclusion polymorphism

D Generics 和 overriding

正确答案：D

3.5 Equality

填空题 第 1 题 3 分

ADT 的 `equals()`需满足的三个性质是 [填空 1] 性、[填空 2] 性、 [填空 3] 性

请填写中文

正确答案：

自反;对称;传递

多选题 第 2 题 2 分

以下针对 ADT 等价性的说法，不正确的是____

A 如果对象 a 和 b 的 R 值被 AF 映射到相同的 A 值，则 a 和 b 等价

B 对对象 a 和 b 调用任何相同的方法，都会得到相同的返回值，则它们是等价

的

C 对象 a 和 b 不等价，那么该 ADT 中不应存在任何方法 op 使得

`a.op()=b.op()`

D 对象 a 和 b 是等价的，那么 a 和 b 的 rep 中每个 field 的值也一定是相等的

正确答案：CD

答案解析：

A 选项是从 AF 的角度判断等价性。

B 选项是从外部观察者的角度来判断等价性。

C 选项：是可以存在这样的操作的。

即使这个操作返回相同的结果，该 ADT 还应有其他操作会让 a 和 b 执行后的结果不同。

填空题 第 3 题 2 分

Java 中有两种不同的操作来验证对象的等价性，分别为==和 equals()，它们的“学名”是 [填空 1] 等价性和 [填空 2] 等价性。

请填写中文

正确答案：

引用;对象

单选题 第 4 题 1 分

某个 ADT 的 Rep 是：

```
private int no;
```

```
private String name;
```

它的 AF 是：

AF(no) == ID of a student

那么以下 `equals()`正确的是__

A return this.no == that.no

B return this.no == that.no && this.name == that.name

C return this.no == that.no && this.name.equals(that.name)

D return this.name.equals(that.name)

正确答案：A

填空题 第 5 题 2 分

针对 mutable 的 Java 类，有两种等价性，分别为 [填空 1] 等价性和 [填空 2] 等价性。

请填写中文

正确答案：

观察;行为

单选题 第 6 题 1 分

以下关于的等价性的说法，不正确的是____

A `object` 类缺省的 `equals()`是判断两个对象内存地址是否相等

B 若 a 和 b 满足 `a.equals(b)`为真，那么 `a.hashCode() == b.hashCode()`也应为真

C 若 `a.equals(b)`为假，那么 `a.hashCode() == b.hashCode()`可以为真

D 针对 mutable 的类，由于其 rep 在变化，直接使用==判断是否相等即可，无需 override `equals()`

正确答案：D

答案解析：

D 选项：这要取决于实际需求。例如 Java 中 List 和 Date 等 mutable 的类，都 override 了 equals()方法，实现观察等价性。

多选题 第 7 题 2 分

关于 hashCode()的说法，不正确的是__

B 具有相同 `hashCode()`返回值的两个对象 `a` 和 `b`，必须做到 `a.equals(b)`为真

C 若 `hashCode()`始终返回同一个值，则违反了 `Object` 对 `hashCode()`的要求

D 如果不 `override hashCode()`方法，`Object` 类缺省实现也可以满足要求

E 一个对象实例的 `hashCode()`返回值，在其生命周期内可以发生变化

正确答案：BCD

答案解析：

针对 C 选项：还是要看需求是什么，不能一概而论

针对 D 选项：mutable 对象的 hashCode 在其 mutator 方法执行后，就会发生变化，因此在 hash table 中的位置也可能随之发生变化。

单选题 第 8 题 1 分

```
Date d = new Date(2019,4,1);
Set<Date> hs = new HashSet<>();
```

```
Set<Date> ts = new TreeSet<>();
hs.add(d);
ts.add(d);

d.setYear(2020);

println(hs.contains(d));
println(ts.contains(d));
```

该代码执行之后会打印出____

A True; True

B True; False

C False; True

D False; False

正确答案：C

答案解析：

haahCode，顾名思义...

mutable 对象的 hashCode 变化之后，只会影响受 hashSet，不会影响

TreeSet

单选题 第 9 题 1 分

```
Date d = new Date(2019,4,1);
Date f = new Date(2019,4,1);

List<Date> al=new ArrayList<>();
List<Date> ll=new LinkedList<>();

al.add(d);
ll.add(f);

println(d.equals(f));
```

```
println(al.equals(ll));
```

该代码执行之后输出的是____

A True; True

B True; False

C False; True

D False; False

正确答案：A

答案解析：

Date 是 mutable 的，它实现的是观察等价性，所以 d 和 f 两个值等价。

List 也是 mutable 的，实现的也是观察等价性，所以判断其中每个元素的等价性，而 d 与 f 是等价的，故 al 和 ll 也是等价的。这个跟具体的类型（ArrayList 和 LinkedList）无关。

5-1 可复用性

单选题 第 1 题 1 分

Programming for reuse

Programming with reuse

二者的区别，不正确的是___

A for：开发可复用的软件；

with：用可复用的软件开发自己的软件

B with：开发可复用的软件；

for：用可复用的软件开发自己的软件

C for：难点在于抽象（abstraction），让开发出的软件能适应于不同但相似的应用场合

D with：难点在于适配（adaption），让自己的软件与来自外部的软件之间做好恰当的连接

正确答案：B

答案解析：

这道题你还错，说明本节课没听课...

有同学说“课堂上需要记忆的东西太多了”，我想说的是：不记忆点新东西，你的编程总是停留在之前写代码所养成的习惯中。要让课堂上这些一点一滴的东西逐步改变你的编程习惯。

单选题 第 2 题 1 分

Lab2 中，你开发了 `Graph<L>`，然后在 `FriendShipGraph` 中使用 `Graph<L>`表示人与人之间的社交网络，此为___

A Code level reuse

B Module level reuse

C Library level reuse

D System level reuse

正确答案：B

单选题 第 3 题 1 分

为了让你的 Lab2 具备可视化功能，你决定复用 Lab1 的 `Turtle Graphics`，于是在代码里加入 `import turtle.*`，然后在中用 `Turtle` 的相关类和方法执行图的可视化，此为___

A Code level reuse

B Module level reuse

C Library level reuse

D System level reuse

正确答案：C

答案解析：

潜藏的操作是你必须将 turtle 的 jar 包或.class 目录加入你的项目 path 中。该 jar 表示一个可复用的外部 library。

就如同：为了让你的程序具备 Junit 测试能力，你必须将 junit.jar 放入你的 path 里。

单选题 第 4 题 1 分

你在 GitHub 上搜索了某个 `ConvexHull` 的算法，将其代码复制到你的 Lab1 中，这属于

A Code level reuse

B Module level reuse

C Library level reuse

D System level ruse

正确答案：A

答案解析：

虽然不算“剽窃”，但如果你的软件投入商业用途，必须要遵循对方的开源许可协议

多选题 第 5 题 2 分

以下__技术对开发高可复用性的软件有积极意义

A 泛型/参数化，例如 `Graph<L>`中的`<L>`

B 使用 interface 定义操作，而非用 class 直接实现 op

C 设计和实现 abstract class

D 使用 override 和 overload

E 将 ADT 的 rep 设置为 `private` 和 `final`，并避免表示泄露

F 精心撰写符合要求的 spec 并生成 Java Doc

正确答案：ABCDEF

单选题 第 6 题 1 分

Framework 是一种典型的复用形态，它与传统的 API 复用存在区别，以下不正确的是____

A API 复用是将外部开发的 API 放到自己的代码中去调用，自己的代码是可执行程序的主体

B Framework 复用是将自己的代码填充到 framework 中，可执行程序的主体是 framework

C API 复用的学习周期短，framework 复用的学习周期较长

D API 复用的粒度大，framework 复用的粒度小

正确答案：D

5-2(1) Subtyping

单选题 第 1 题 1 分

她是谁?

向大师致敬!

A Margaret Hamilton

B Ada Lovelace

C Barbara Liskov

D Frances Allen

正确答案：C

多选题 第 2 题 2 分

Behavioral subtyping 必须要满足的条件，不包括以下__

A 子类型可以增加父类型中所没有的新方法

B 子类型 override 父类型的某方法，子类型方法需具备相同或更弱的 post-condition

C 子类型必须要具备与父类型相同或更弱的 invariants（不变量）

D 子类型 override 父类型的某个方法，不能比父类型方法抛出新的异常类型，但可比父类型方法抛出的异常更少

正确答案：BC

多选题 第 3 题 2 分

关于 Behavioral subtyping 的说法，不正确的是___

A 子类型 override 父类型某个方法，其返回值类型应该与父类型方法的返回值类型相同或者更具体（子类型）

B 子类型 override 父类型某个方法，其参数的类型应该与父类型方法的参数类型相同或者更具体

C 某个类是 immutable 的，它可以派生出一个 mutable 的子类

D 子类型 override 父类型某个 public 方法，子类型中该方法的可见性可以为 private

正确答案：BCD

答案解析：

C 选项：java 允许你这么设计，static type checking 也可以通过，但却是违反 LSP 的，因为 immutable 也是 invariant（不变量），子类型的 invariant 应该等于或强于父类型，而一个 mutable 的子类型就不再具备这个 invariant。

D 选项：把一个 public 方法 override 为 private 方法，那么如果用子类型的对象取代父类型对象，就无法调用这个操作了，所以无法取代，所以违反 LSP。

单选题 第 4 题 1 分

```
public class A {
        public Object a (String d) {
                return "";
        }
}
public class B extends A {
        @Override
        public String a (Object d) {
                return null;
        }
}
```

该段 `Java` 代码是否能通过 `static type checking`?

A 能

B 不能

正确答案：B

答案解析：

虽然方法 a 在类 A 和类 B 里符合 covariance 和 contra-variance，但因为 Java 不支持 contra-variance，这里的@Override 是不能成立的。

单选题 第 5 题 1 分

```
public class A {
        public Object a (String d) {
                return "";
        }
}
public class B extends A {
        public String a (Object d) {
                return null;
        }
}
```

该段 `Java` 代码是否能通过 `static type checking`?

A 能

B 不能

正确答案：A

答案解析：

方法 a 在类 A 和类 B 里符合 covariance 和 contra-variance，但因为 Java 不支持 contra-variance，于是编译器就把 B 中的 a 方法看作了 A 中 a 方法的 overload。

多选题 第 6 题 2 分

```
void print(List<Object> list)
{...}
```

以下___作为参数传递进去不是对它的合法调用?

A `List<Integer> a;`

B `List<?> a;`

C `ArrayList<Integer> a;`

D `List<? extends Object> a;`

E `List<Object> a;`

正确答案：ABCD

多选题 第 7 题 2 分

```
void print
(List<? extends Number> list)
{...}
```

以下___作为参数传递进去不是对它的合法调用?

A `List<Integer> a;`

B `List<?> a;`

C `ArrayList<Integer> a;`

D `List<? extends Integer> a;`

E `List<Object> a;`

正确答案：BE

5-2(2) 组合与委派、框架复用

单选题 第 1 题 1 分

关于 delegation 的说法，不正确的是___

A 某些功能不需要 ADT 自己来做，而是委托其他 ADT 来做（调用其他 ADT 的方法）

B 在类 1 的构造器里传入类 2 的对象，即可在类 1 里使用传入的对象调用类 2 的操作

C 为实现类 1 对类 2 的 delegation，需要将类 2 作为类 1 的 rep 的一部分（即类 2 是类 1 的 field）

D Delegation 发生在 object 层面而非 class 层面

正确答案：C

答案解析：

选项 C：不必需。可以动态传入类 2 的对象，然后调用其方法即可。

```
class A {
        ...
        void a(B b) {
                b.add();
        }
}
```

在上面的代码中，B 并未作为 A 的 rep 的一部分，只是在方法 a 中动态传入，然后调用 b 的 add()方法。

单选题 第 2 题 1 分

关于 Inheritance 和 delegation 的说法，不正确的是____

A 如果类 1 需要使用类 2 的大部分乃至全部方法，则最好将类 1 设计为类 2 的子类

B 如果类 1 只需要类 2 的小部分方法，则最好通过 delegation 实现对类 2 的

方法调用

C Delegation 发生在 object 层面，而 Inheritance 发生在 class 层面

D 用 A has a B 或 A use a B 表示 A 和 B 之间的 inheritance 关系，用 A is a B 表示二者之间的 delegation 关系

正确答案：D

答案解析：

D 选项：正好说反了

多选题 第 3 题 2 分

关于 Composite over inheritance 原则，以下正确的是___

A 若用继承，子类不得不继承父类的全部方法，但子类可能并不需要这么多方法

B 子类的不同对象可能具备不同的行为，若用继承实现，则不得不定义大量具有不同行为的子类，麻烦

C 若用继承，子类和父类之间的关系在编译阶段即硬性绑定，难以在运行时动态改变

D 若用 delegation，可隔离两个类之间的静态绑定关系，从而支持运行时实例之间关系的动态改变

正确答案：ABCD

单选题 第 4 题 1 分

若事物 o 既有 A 行为（例如“飞”），也有 B 行为（例如“叫”），但具体如何实现这些行为可能会变化。那么 ready for change 的 Java OOP 设计是__

A 定义一个接口，在其中为 A 和 B 分别定义一组方法，然后为 o 设计一个类，实现该接口

B 为 A 和 B 分别定义一个接口，各自分别定义自己的方法；为 o 设计类，将其具体行为 delegate 到 A 和 B 的具体实现类

C 为 o 设计类，在类中添加 A 和 B 的行为作为其一组方法

D 针对 A 设计一个类，针对 B 设计另一个类，然后让 o 的类同时继承 A 和 B 的类

正确答案：B

答案解析：

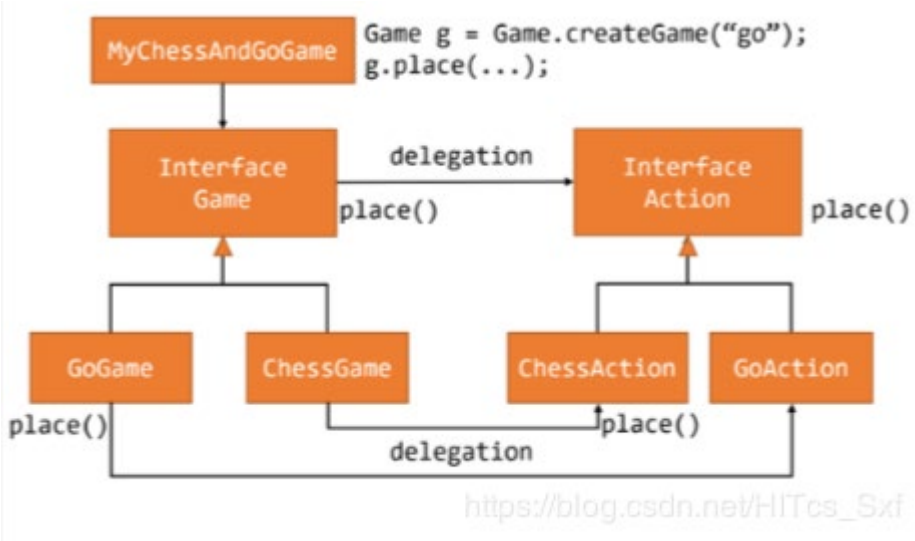
A 选项：如果 o 将来不再具备 A 或 B 的行为，那么需要修改该接口，会造成大量依赖该接口的类随之修改。

B 选项：将两类行为分别放在不同接口里，o 的类通过 delegation 调用两类接口的具体实现。这是 CRP 原理的恰当应用。

C 选项：对接口编程，而非对类编程。

D 选项：Java 不支持多重继承。

单选题 第 5 题 1 分



Lab2 的 P3 按照上述 UML 图进行设计，那么 `GoGame` 类的 `place()`方法（落子）的 spec 应该为____

A `void place(Action a, ...)`

B `void place(GoAction a, ...)`

C `void place(...)`

D `void place()`

...表示其他参数

正确答案：B

答案解析：

该题目的目的不是考核，而是帮你思考你的 Lab2 的 P3 设计是否有改进的空间。

记住软件构造的目标：

Ready for change

Safe from bugs

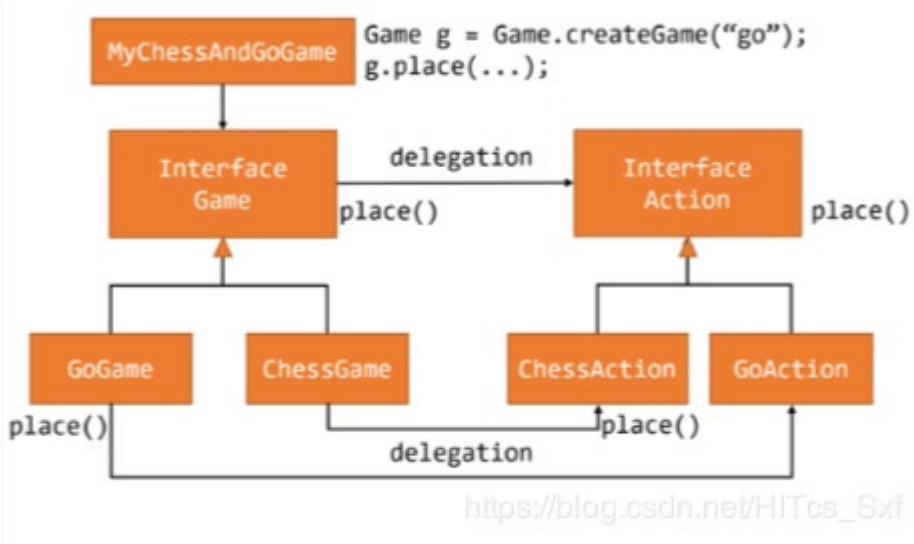
Design for reuse

Easy to understand

Efficient to run

第三章我们关注的是 safe from bugs，本章开始关注 design for reuse 和 ready for change，也是更难的东西。

多选题 第 6 题 2 分



Lab2 的 P2 按照上述 UML 图进行设计，这么设计的优点是____

A 客户端只需了解 `Game` 接口，无需获知其他任何信息

B 可容易的扩展其他棋类游戏（只需为 `Game` 和 `Action` 两个接口添加新的实现类）

C 如果要扩展支持真实的围棋规则，只需改 `GoAction` 类即可，最小化改变范围

正确答案：ABC

答案解析：

该题目的目的不是考核，而是帮你思考你的 Lab2 的 P3 设计是否有改进的空间。

记住软件构造的目标：

Ready for change

Safe from bugs

Design for reuse

Easy to understand

Efficient to run

第三章我们关注的是 safe from bugs，本章开始关注 design for reuse 和 ready for change，也是更难的东西。

单选题 第 7 题 1 分

关于黑盒与白盒的 framework 的说法，不正确的___

A 白盒框架里实现了若干方法，用户程序通过 override 这些方法即可改变框架的行为

B 在黑盒框架里，用户编写类来实现框架提供的接口，框架运行时动态调用用户写的类中的方法

C 白盒框架的基本原理是 inheritance，黑盒框架的基本原理是 delegation

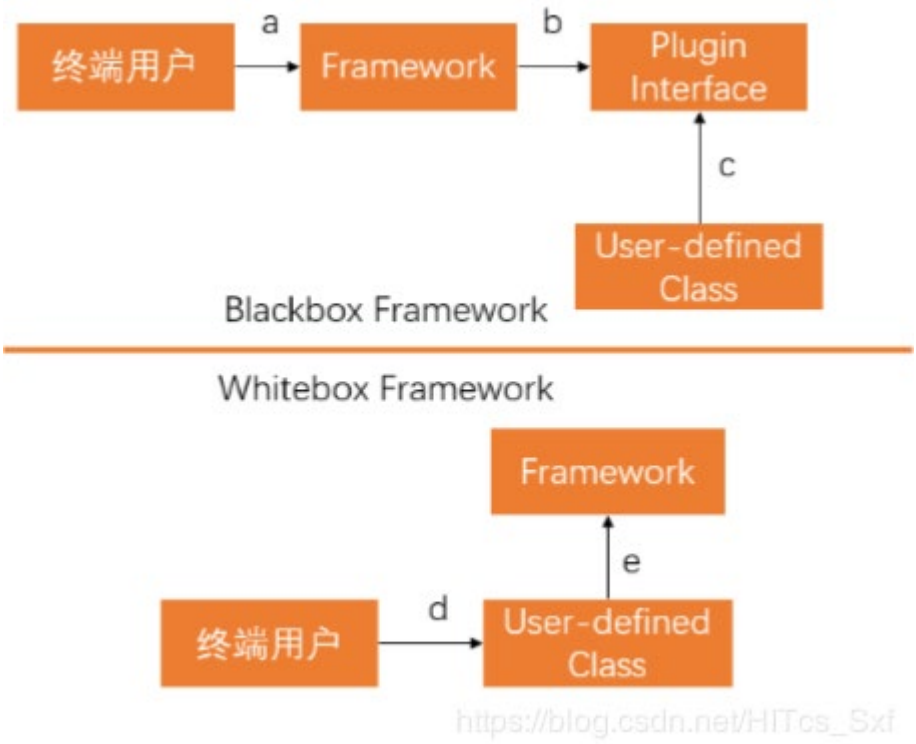
D 黑盒框架运行时主程序是用户写的类，白盒框架的主程序是框架本身的类

正确答案：D

答案解析：

没啥说的，看讲义吧

多选题 第 8 题 2 分



上图中表示 delegation 关系的是____

A a

B b

C c

D d

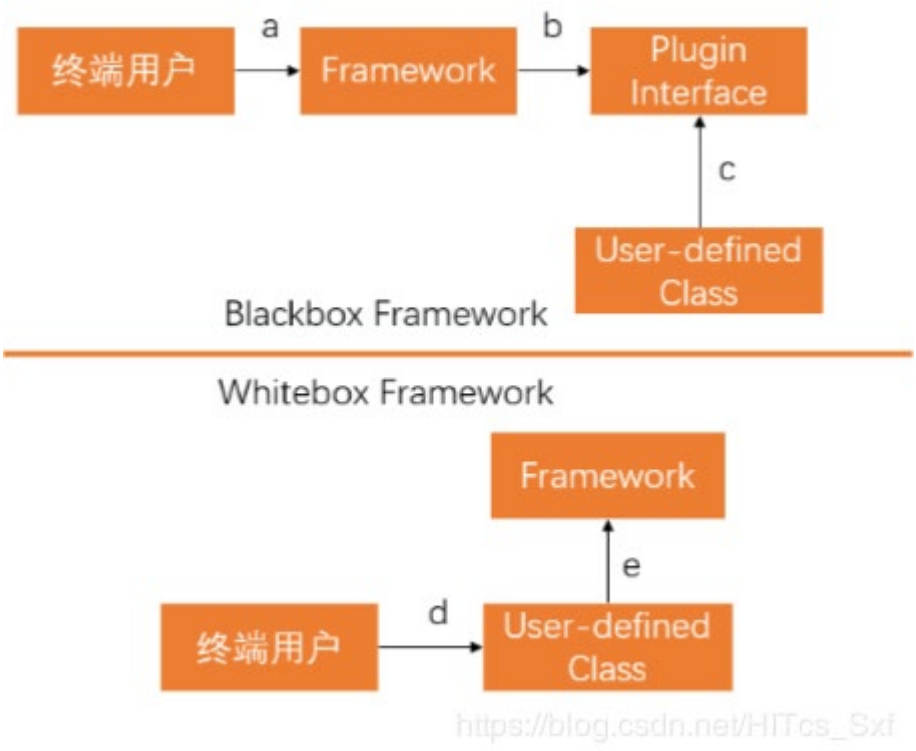
E e

正确答案：B

答案解析：

a 和 d，并非 ADT 之间的关系：这里的“终端用户”就是使用软件的真人了。

多选题 第 9 题 2 分



上图中表示继承或实现关系的是____

A a

B b

C c

D d

E e

正确答案：CE

答案解析：

c 是用户开发的具体类实现了 Plugin 接口，e 是用户定义的类继承了 framework 所提供的抽象类。

5-3 面向复用的设计模式

多选题 第 1 题 2 分

经典著作《Design Patterns: Elements of Reusable Object-Oriented Software》的作者是___

向前辈程序员致敬!

A Ralph Johnson

B Yoshua Bengio

C Yann LeCun

D Richard Helm

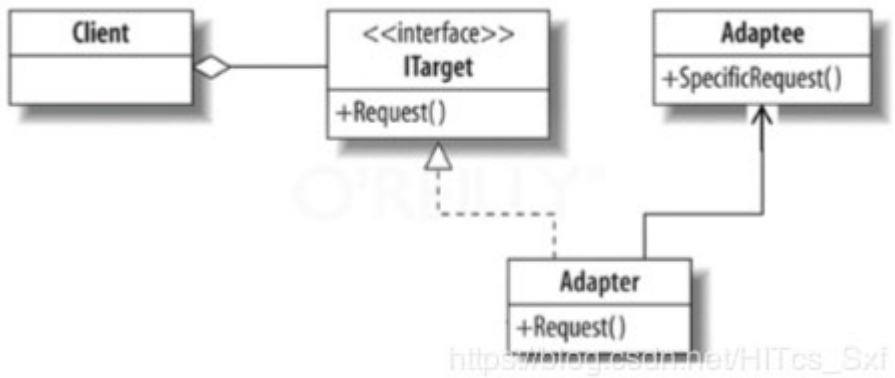
E John Vlissides

F Geoffrey Hinton

G Erich Gamma

正确答案：ADEG

单选题 第 2 题 1 分



关于 Adapter 模式的说法，正确的是___

A Adapter 类可提供被复用的方法，但与 Client 要求的 spec 不吻合

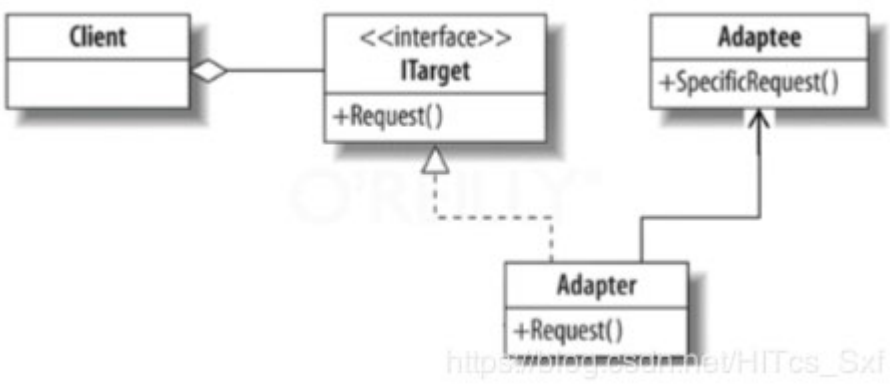
B Adaptee 类用于将 client 的请求转化为对 Adapter 类的方法的调用

C Adapter 类和 Adaptee 类之间的关系是 delegation

D Adapter 和 ITarget 之间的关系是 inheritance

正确答案：C

多选题 第 3 题 2 分



Adapter 模式为何要设计一个 `ITarget` 接口，而不是由 client 直接请求 `Adapter` 的 `Request()`方法?

A Client 对接口编程，无需了解 `Adapter` 类

B `Adapter` 类可能发生变化，接口隔离了该变化，不会影响 client 代码

C `Adaptee` 类可能发生变化，需要隔离 client 与他的变化

D Client 无法直接构造 `Adapter` 的实例，故需要增加接口

正确答案：AB

单选题 第 4 题 1 分

关于 Decorator 设计模式的说法，不正确的是____

注：在以下选项中，A 代表装饰之前的类，B 代表装饰之后的类

A A 和 B 二者具有共同的祖先类或实现共同的接口

B B 中有一个成员变量，其类型为 A

C B 中的某些方法，通过 delegation 调用 A 的同名方法，并可扩展其他新功能

D B 中不可以增加 A 中所没有的方法

正确答案：D

多选题 第 5 题 2 分

```
Stack t = new SecureStack (
        new SynchronizedStack (
        new UndoStack(s)
);
```

针对 Decorator 设计模式上述形式的 client 代码，以下说法不正确的是____

A 第三行的 `s` 的类型是 `Stack` 或其子类型

B 第三行的 `s` 的类型是 `UndoStack`

C 第一行的 `t` 的运行时类型是 `SecureStack`

D 第一行的 `t` 和第三行的 `s`，其实是指向内存中同一个对象的两个不同 `alias`

正确答案：BD

单选题 第 6 题 1 分

针对 Façade 设计模式，以下说法不正确的是__

A 针对一个或多个已有的类，client 需要调用它们的多个方法，故该模式将这些调用统一封装为一个方法对外提供

B 该模式降低了 client 端使用已有类的代价，减少了 client 与已有类之间的耦

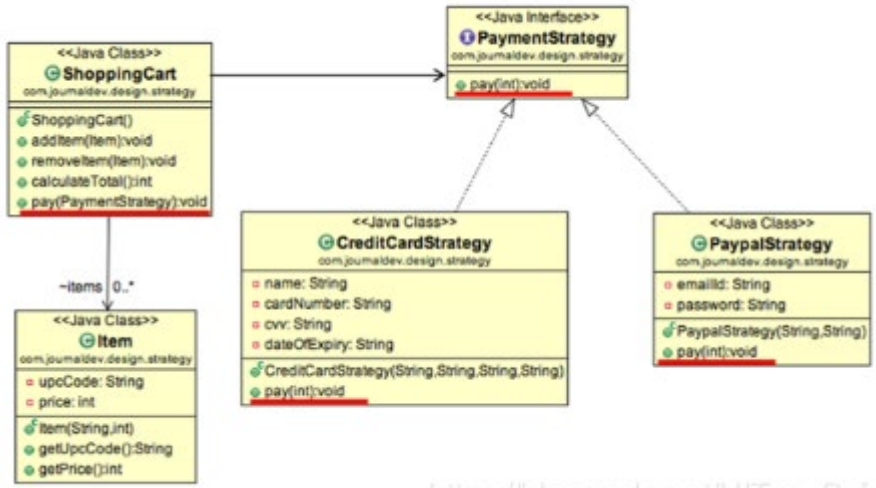
合度

C 该模式所构造的封装类中的方法一般用 static 类型

D 虽然减少了耦合度，该模式仍然需要 client 端显式构造出被封装类的实例

正确答案：D

单选题 第 7 题 1 分



上图 Strategy 模式，说法不正确的是__

A `ShoppingCart` 将 `pay()`的职责 delegate 给了 `PaymentStrategy`

B `PaymentStrategy` 是接口，client 的 `pay` 实际执行的是其某个实现类的 `pay()`

C Client 类不需要了解 `PaymentStrategy` 的任何实现类即可使用其 `pay()`

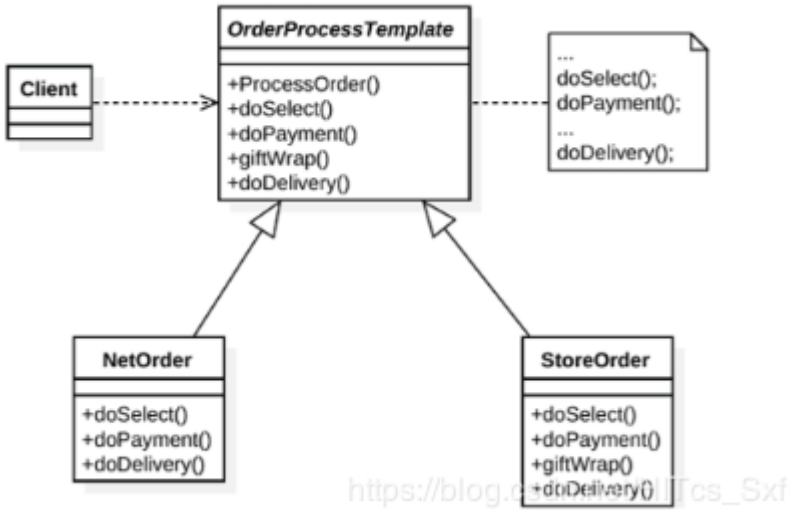
D 若要扩展新的 `pay()`策略，只需为 `PaymentStrategy` 增加新的实现类

正确答案：C

答案解析：

针对 C 选项：client 必须要知道自己希望使用的是哪个 pay 策略，然后构造其对象，传入 pay(PaymentStrategy p)方法，才能调用 p 的 pay()方法。

多选题 第 8 题 2 分



上图显示的 Template 模式，不正确的是__

A 上图的 `OrderProcessTemplate` 是“模板”，可以是接口，也可以是抽象类

B `ProcessOrder()` 是模板方法，包含了对一系列操作的调用

C 如果 `OrderProcessTemplate` 中没有某个 `doXXX()` 方法的实现，则在 `NetOrder` 和 `StoreOrder` 中必须要实现它。

D Client 可以更改对一系列 `doXXX()` 方法的调用次序。

正确答案：AD

多选题 第 9 题 2 分

关于 Iterator 模式，不正确的是____

A 为了让你的类 A 具备 iteration 的能力，需要 A 实现 `Iterator` 接口

B 为了让你的类 A 具备 iteration 的能力，还需要构造一个实现 `Iterable` 接口的类作为符合 A 特定需求的迭代器

C Java 里 `Iterable` 接口只有一个方法 `iterator()`，它返回一个迭代器对象

D Java 里 `Iterator` 接口有三个方法 `hasNext(), next(), remove()`

正确答案：AB

6-1 可维护性

单选题 第 1 题 1 分

涉及以下三个场景的软件维护活动分别称为___

1 将 iOS 10 下的 App 升级，使之能在 iOS 11 下运行；

2 用户报告一个功能 bug，对该 bug 进行修复并重新发布；

3 为改善性能，发布一个补丁

A 预防性、完善性、纠错性

B 适应性、纠错性、完善性

C 完善性、适应性、预防性

D 纠错性、预防性、适应性

正确答案：B

多选题 第 2 题 2 分

关于软件维护和演化的说法，正确的是___

A 在设计与开发阶段就要考虑将来的可维护性，而不是等到软件发布之后再考虑

B 对软件代码的频繁变化，可能导致软件复杂度的增加和质量的下降

C 软件开发的大部分成本来自于维护阶段

D 软件维护是软件开发团队中“运维工程师”的工作

正确答案：ABC

多选题 第 3 题 2 分

以下___可提高软件的可维护性

A 高内聚、低耦合

B 每个类的责任应尽可能单一

C 遵循 LSP 原则设计类

D 抽象的模块不应依赖于具体的模块

E 避免类产生“表示泄露”

F 降低代码复杂度，提高代码可读性

正确答案：ABCDEF

答案解析：

A 显然，这是可维护性的最基本的原则

B “单一责任原则”SRP

C LSP 原则

D OCP 原则

E 信息隐藏原则

F 显然

多选题 第 4 题 2 分

关于可维护性的说法，不恰当的是___

A 开发中使用的类的数量越多，就越难以保护

B 对新功能的扩展所影响的已有代码范围越小，可维护性就越高

C 客户端调用类的某个方法，方法所需的参数越多，将来维护的代价可能就越大

D 类结构里使用的 overload 和 override 越多，维护的代价就越高

正确答案：A

答案解析：

A 类的多少与可维护性无直接关系

B 这就是“可维护性”追求的目标

C 接口要尽可能小，参数多，交互复杂，维护困难

D overload 和 ovverride 过多，影响代码的可理解性，的确会造成一定程度的维护性下降

单选题 第 5 题 1 分

关于“开放-封闭原则”（OCP）的说法，不正确的是___

A 能够通过在已有类结构上增加新的类来进行扩展

B 对类结构进行扩展的时候，不应或应尽可能少的变化已写好的代码

C 为支持 OCP，客户端不应该直接使用具体的类，而应尽可能使用接口

D 客户端使用接口来调用方法，即可避免声明特定实现类的对象

正确答案：D

单选题 第 6 题 1 分

关于面向可维护性的 OO 设计原则的说明，不正确的是__

A 对 client 的需求分类，将 client 共同使用的一组方法放到同一个接口里，其他方法分离出去

B 如果 client 代码里出现了对某个具体实现类的调用，最好引入接口，使 client 避免直接接触类

C 尽可能降低类之间的耦合度，而类之间的 delegation 越多，耦合度可能就越大。

D 当发生需求变化时，首先考虑对原有代码进行修改，然后再考虑增加新类满足之

正确答案：D

6-2 面向可维护性的设计模式

单选题 第 1 题 1 分

关于 Factory Method 模式，说法不正确的是__

A 如果不想在 client 代码中指明要具体创建的对象类型时，使用该模式比较合适

B 定义一个接口，其中包含工厂方法，然后在它的不同实现类中完成不同子类型的对象实例的创建

C 工厂方法可以被实现为静态的，即静态工厂方法

D 使用工厂方法创建的对象实例是接口类型，在 client 代码里使用 `instanceof` 也无法得知其具体类型

正确答案：D

多选题 第 2 题 2 分

Factory Method (FM) 和 Abstract Factory (AF)模式，错误的是__

A AF 模式为了解决客户端创建遵循固定搭配规则的多个不同类型的对象实例的麻烦

B AF 模式在创建每个类型实例的时候，本质上是使用了 FM 模式

C Client 不能仅使用 AF 的某个具体实现类来创建单个对象实例，必须要同时创建多个不同类型的对象实例

D Client 可用多个 FM 来取代 AF 模式，可达到同样的目的

E 如果要改变不同类型的对象实例的固定搭配，只需扩展新的抽象工厂实现类即可

正确答案：CD

答案解析：

针对 C 选项：AF 的实现类里提供了多个工厂方法，分别创建不同类型的对象实例，Client 完全可以只调用其中一个而忽略其他。当然这么做就失去了抽象工厂方法的意义。

多选题 第 3 题 2 分

关于 Builder 模式，错误的是__

A Client 要的不是一堆零散的 objects，而是一个完整的对象（由若干小的 objects 组成）

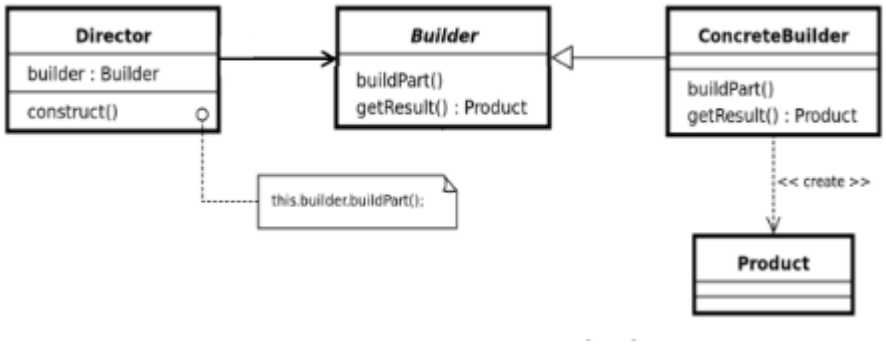
B Client 不需了解各部分怎么创建、怎么组合，这些细节完全隐藏在 Director 类当中

C Builder 在构造复杂对象内部各部分时，需要遵循严格的构造次序

D 可以支持多种不同的产品构建，只要为每一个产品实现一个具体的 builder 类即可

正确答案：C

单选题 第 4 题 1 分



在上图所示 builder 模式中，负责将各个“部分”组装在一起的方法是_

A `Builder` 的 `buildPart()`

B `Builder` 的 `getResult()`

C `ConcreteBuilder` 的 `getResult()`

D `Director` 分的 `construct()`

正确答案：D

答案解析：

`construct()`是负责调用每一个“部分”的 build 方法，意即将一个对象的各个部分逐步构造出来。而 `getResult()`是获得组装后的结果（单一对象）。

单选题 第 5 题 1 分

关于 Bridge 模式的说法，不正确的是____

A 通过 delegation+inheritance 建立两个具体类之间的关系，避免具体类之间的直接“接触”

B Bridge 模式中两个具体对象之间的关系是在编译阶段就建立起来，在运行阶段实现 delegation

C Bridge 模式中，两个 ADT 对象之间的关系是永久保存的

D Bridge 实现了两棵 inheritance 继承树，两棵树中的不同子类之间的关系根据 client 的具体需求在 client 代码中动态确定

正确答案：B

单选题 第 6 题 1 分

关于 proxy 模式，说法不正确的是__

A 目的是在 `client` 代码和具体被访问的类代码之间建立隔离

B `Client` 实际请求的 `Proxy` 类和实际执行任务的类，二者实现同一个接口或具有相同的父类

C `Client` 代码必须知道 `Proxy` 类的名字并 `new` 一个 `Proxy` 对象实例，否则无法使用其功能

D 可在 `Client` 请求的 `Proxy` 类中实现 `lazy initialization`、`lazy loading` 等，降低执行具体任务的类的负担

正确答案：C

答案解析：

C 选项：当然可以使用 factory method 将 proxy 隐藏起来，对 client 不可见。

单选题 第 7 题 1 分

关于 composite 模式，不正确的说法是__

A 该模式用来解决“递归包含”的问题，程序运行时的多个对象构成一棵树型结构

B 树型结构中的所有节点，实现相同的接口或继承自相同的类，符合 LSP 原则

C 每个节点对象具有一个集合类型的 field，保存其下层的节点对象，集合内元素数量>0

D 该树型结构是在运行时根据 client 代码动态构建起来的

正确答案：C

单选题 第 8 题 1 分

关于 observer 模式，不正确的是___

A 该模式在 `1` 个 `subject` 对象和一组 `observer` 对象之间建立 `1` 对多的 `delegation` 关系

B `Observer` 对象调用 `subject` 对象的 `attach()`方法，将自己加入到 `subject` 对

象的“粉丝”队列中

C 当 `subject` 对象的状态变化时，它回调各个 `observer` 对象的 `update()` 方法

D 在 `Java` 中，各具体 `observer` 类要继承自 `Java Observer` 类

正确答案：D

单选题 第 9 题 1 分

关于 visitor 模式，不正确的是__

A 该模式可将一个 ADT 定义的某些特定操作从原 ADT 实现类中抽取出来，单独放在 visitor 实现类中

B 给原 ADT 传递进不同的 visitor 实现类的实例，即可对 ADT 对象实例做不同的操作，但无需改变原 ADT 的代码

C 被 visit 的 ADT，需要提供 accept(Visitor v)这样的方法，便于将一个外部 visitor 实例传递进来

D 被 visit 的 ADT，可以提供诸如 accept (Visitor v1, Visitor v2)的方法，即可支持对 ADT 做两个外部操作的组合

正确答案：A

答案解析：

A 选项：形式上类似于该选项所述，但这并非是 visitor 模式的本意

D 选项：一下子扩展两个操作？其实是可以实现的。

多选题 第 10 题 2 分

以下关于 Java 中的`-able` 和`-ator` 的说法，不正确的是__

A `Comparator<T>`是个接口，其类要实现的方法是 `compareTo(T a)`

B `Comparable<T>`是个抽象类，子类需要实现的方法是 `compare (T a1, T a2)`

C `Iterable<E>`是个接口，`iterator()`是其类要实现的方法

D `Iterator<E>`是个抽象类，其子类要实现的方法有 `hasNext(), next(), remove()`

E `Observable` 是个抽象类，其子类要实现的方法之一是 `notifyObservers()`

F `Observer` 是个抽象类，其子类要实现的方法之一是 `update(...)`

正确答案：ABDF

6-3 可维护性构造技术

单选题 第 1 题 1 分

关于 State 设计模式，说法不正确的是__

A 使用 delegation 机制，将状态转换的行为委派到多个独立的 State 类中去完成

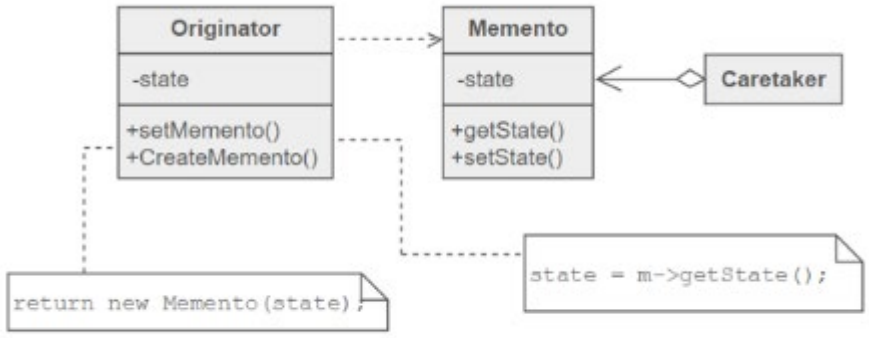
B 各具体 State 类分别代表对象能达到的某种状态，并负责完成在该状态下所能进行的状态转换，向客户端返回某个新状态的实例

C 避免了在 ADT 内部代码中使用 if/else 结构实现状态转换，支持将来状态可能的扩展

D ADT 自己需要提供一个类似于 `accept(State s)`的方法来判断当前状态是否为最终状态

正确答案：D

单选题 第 2 题 1 分



关于 Memento 模式，不正确的是_

A `Originator` 是需要备忘的类

B 一个 `Memento` 对象代表“备份”的一次历史状态

C `Originator` 类利用 `Memento` 和 `Caretaker` 类管理一系列历史状态，随时进行备份和恢复

D 备忘和恢复的动作由 client 端代码发起，而非由 `Originator` 类和 `Memento` 类发起

正确答案：C

单选题 第 3 题 1 分

```
class Caretaker {
        List<Memento> ms = new ArrayList<>();
        public void addMemento(Memento m) {
                ms.add(m);
        }
        public Memento getMemento(int i) {
                return ms.get(i);
        }
}
```

如果 client 端想恢复离当前时刻最近的倒数第 i 次备份的状态，上述代码中的“？”应该是___

A `i`

B `i-1`

C `ms.size()-i`

D `ms.size()-i+1`

正确答案：C

单选题 第 4 题 1 分

```
class Caretaker {
        List<Memento> ms = new ArrayList<>();
        Memento getMemento(int i) {
                return ms.get(?);
        }
        Memento repeat(int i) {
                return ms.get(??);
        }
}
```

Client 用`getMemento(i)`恢复倒数第 i 次备份状态后，想调用`repeat(i)`来撤销本次恢复，即重新回到恢复前的状态，那么上述代码中的“??”应该是___

A `ms.size()-i`

B `i+1`

C `ms.size()-1`

D 以上 ABC 均无法做到

正确答案：D

单选题 第 5 题 1 分

关于 Table-driven construction，说法不正确的是___

A 将代码中复杂的 `if-else` 和 `switch-case` 语句从代码中分离出来，通过查表完成

B 表驱动编程的运行效率会远低于传统的 `if-else` 和 `switch-case` 方式，需要折中考虑

C 待查的“表”，既可以是代码中的 `final` 变量，也可以是外部文件或数据库（程序运行时从文件或库读入）

D 好处是当规则发生变化时，业务逻辑代码无需修改，提高了可维护性

正确答案：B

答案解析：

B：如果表是在代码中的 fianl 变量，其效率要高于 if/else 语句。

多选题 第 6 题 2 分

以下关于 Grammar 的说法，不正确的是___

A 语法驱动编程适用于从外部读入遵循特定格式的文本数据并加以解析的场景

B Regular grammar 是指 grammar 被简化之后可以表达为一个不包含任何非终止节点的产生式

C 正则表达式未必总是 regular grammar

D Parse tree 是 parser 根据特定 grammar 对某个字符串进行解析之后得到的结果

E 在 grammar 的操作符中，|的优先级高于*和+

正确答案：CE

单选题 第 7 题 1 分

```
F :: = B? E N* M
B :: = >
E ::= : | ; | 8
N ::= - | ^
M ::= D | O | P
```

以下字符串__符合该语法定义?

A `:-^[`

B `B:^D`

C `:---D`

D `<3`

正确答案：C

单选题 第 8 题 1 分

以下哪一对正则表达式所表示的字符串内容是等价的___

A

```
(cb*c)*
c(b*c)*
```

B

```
\?[a-z]+(y*by*)*
\?\w+(y)*(by?)*
```

C

```
(a|b|c*)*(d?)
(a|b|c*)*(c*|d)
```

D

```
([^abc]+x*)+
^a[bc]([^abc]*x*)+x
```

正确答案：C

单选题 第 9 题 1 分

在 Java 正则表达式中，\d 和\w 分别代表__

A `[0-9] [a-zA-Z_0-9]`

B A whitespace character `[^0-9]`

C `^[a-zA-Z_0-9] [ \t\n\x0B\f\r]`

D `[^0-9] [^\s]`

正确答案：A

多选题 第 10 题 2 分

以下__字符串是符合下列正则表达式的合法字符串?

```
(\.[a-z]+\.)+[a-z]+(:[0-9]+)?
```

A `hit.edu.cn`

B `hit.ed.u.c.n:88880?`

C `.hit.ed.u.cn:088888`

D `.hit.ed.u.cn`

正确答案：C

更正：四个选项都不对。按照语法的结构，字符串分为三部分，其中第 1 部分在一组字母前后分别匹配一个“点”，按照这个规则，选项 C 和 D 中的“.hit.”先被匹配出来，后面跟的“ed”被语法中的第 2 部分匹配，然后“ed”后头的“."就没法匹配了。

单选题 第 11 题 1 分

从语法的角度看，Java 的编译器 javac 相当于__，与 Java 源代码相对应的 AST 相当于__

A Parser generator, Grammar

B Parser, Grammar

C Parser, Parse tree

D Parser generator, Parse tree

正确答案：C

7-1 健壮性与正确性

单选题 第 1 题 1 分

关于 robustness 和 correctness 的说法，不正确的是__

A 即使用户输入是不符合 spec 的，需要优雅的提示错误、合理终止或继续运行

B “对自己的代码要保守，对用户的行为要开放”，前者针对的是 robustness，后者针对的是 correctness

C 程序员应总是认为 client 总是可以遵循 spec，这是面向 robustness 的编程思想

D 健壮性编程倾向于直接报错（error），正确性编程则倾向于容错（fault-tolerance）

正确答案：A

//更正：题干应为“正确的是”

填空题 第 2 题 3 分

（1）你在 Lab1 编写从外部文件读入数据来构造 magic square 的程序。你发现读入的文件中，两个数字之间并非总是按 spec 期望的\t 分割，于是你决定，只要遇到非\t 分割的数字，就直接退出。这是面向 [填空 1] 的编程。

（2）你要求用户输入 16 位数字的信用卡号码，结果你检测到用户输入了 17 位。你决定截取前 16 位数字作为用户输入，进入后续处理。这是面向 [填空 2] 的编程。

（3）你在 Lab3 里读取外部文本文件，发现文件里小于 10000 的数字也采用了科学计数法，不符合 spec。为此你决定接受这种输入，程序自动将其转为常规数字形式。这是面向 [填空 3] 的编程。

可以填写：正确性、健壮性、不合理

正确答案：

正确性;不合理;健壮性

多选题 第 3 题 2 分

关于 robustness 和 correctness 的说法，不正确的是__

A Make your external interface robust, and make your internal code correct

B 前者是为了让用户 easy to use，后者是为了让开发者 easy to develop

C 过分追求健壮性会让用户不愿意使用你开发的 ADT

D 过分追求正确性会让开发者写代码变得过于复杂

正确答案：CD

多选题 第 4 题 2 分

以下更适用于使用面向 robustness 进行开发的场景包括_

A 嫦娥四号探月飞船上的控制软件

B 新浪娱乐上的明星动态跟踪与搜索功能

C iPhone XS 上的 App 授权管理功能

D 波音 737 MAX 8 上的 MCAS 防失速系统

E HIT CS32207 的 Lab1-Lab6，有几个随意更改你输入文件内容的 TA

正确答案：BE

答案解析：

D：根据目前新闻报道，很多专家认为：目前的波音 737 MAX 8 上的 MCAS 防失速系统，读取了错误的传感器数据，在未加其他校验机制的情况下，就启动下压机头的动作——对 correctness 的追求不够

多选题 第 5 题 2 分

关于 error、fault、failure 的术语区分，说法不正确的是__

（咬文嚼字没什么意思，但还是需要尽量明白“行话”是啥）

A 你将 `a.equals(b)`写成了 `a==b`，这是一个 error

B 程序运行时抛出了一堆红色的 stack trace，这是一次 fault

C 你在 `checkRep()`中用 `assert xx` 检查 RI，这是为了避免在 ADT 代码中引入 failure

D 你通过 debug 定位了 bug 所在，这是为了找出潜藏的 fault

E 你修复了有 bug 的代码，这是消除了 error

正确答案：BC

多选题 第 6 题 2 分

用于支持 robustness 的构造方法包括___

A 在代码中使用 `assertion`

B 在各方法最后使用 `checkRep()`

C 保证 ADT 不出现 `rep explosure`

D 检查 `pre-condition`，一旦发现违反，告知 `client`，要求 `client` 重新输入数据

E 检查 `pre-condition`，一旦发现违反，抛出异常，结束程序运行

F 检查 `pre-condition`，一旦违反则抛出 `Exception`

正确答案：D

填空题 第 7 题 1 分

Java 中用来处理错误和异常的最高层次的接口是 [填空 1]

只需填写接口名，无需带包名。

正确答案：Throwable

填空题 第 8 题 2 分

用于使 ADT 具备排序功能的 Java 接口或抽象类是 [填空 1] 或 [填空 2]

只需填写接口/抽象类名，无需带包名。

正确答案：

Comparator;Comparable

7-2 错误与异常处理

多选题 第 1 题 2 分

以下___可归属 error 的范畴?

A Java 虚拟机无法为对象分配足够内存，导致内存溢出

B 因为磁盘故障导致文件不可读出

C 因为使用了超过数组长度的下标来读取数组中的数据，导致无法读出

D 运行时无法找到所需要的 class 文件

正确答案：ABD

多选题 第 2 题 2 分

关于 Java 异常处理的说法，不正确的是__

A `RuntimeException` 是由程序员在代码里处理不当造成的，是代码中的潜在 bug

B 除 `RuntimeException` 之外，`Exception` 类的其他子类型异常均是程序员所无法控制的，一旦发生就只能等着程序退出

C `RuntimeException` 及其子类所定义的异常，称为 unchecked exception

D 除 `RuntimeException` 及其子类之外的其他 `Exception` 均可归结为 checked exception

正确答案：B

多选题 第 3 题 2 分

从异常名字来判断，__属于 checked 类型的异常

A `NullPointerException`

B `NumberFormatException`

C `FileNotFoundException`

D `NegativeArraySizeException`

E `ClassCastException`

正确答案：C

多选题 第 4 题 2 分

关于 checked 和 unchecked 异常，说法不正确的是__

A 前者需要 try/catch，后者不需要/不必要/不应该 try/catch

B 前者可以自定义新的异常类型，后者不可以

C 前者一旦发生，无法补救；后者一旦发生，程序员可以采取措施补救

D Client 更希望开发者使用后者；而开发者若倾向于使用前者，则会让代码变

得更简洁更易读

E 前者将随常规代码一并交付客户使用，后者则不应出现在交付代码中

正确答案：BCDE

答案解析：

B：二者都可以定义新类型，分别继承自不同的父类

D：client 希望使用 checked 异常，因为可以提示更清晰的错误信息并加以补救，避免程序的直接退出；开发者若倾向于使用 checked 异常，则会让代码变得更复杂（各种各样的 try/catch）。相比较而言，使用 unchecked 异常会让代码简洁易读（只需要在发生异常的现场 throw new XxxException(...)即可）

E：unchecked 虽然无法恢复，但如果代码中有遗留问题，不可避免的抛出 unchecked 异常，不存在交付不交付给用户一说

单选题 第 5 题 1 分

以下说法正确的是__

A 如果父类型中的某个方法声明中未 `throws` 任何 checked 异常，那么子类型中 override 该方法时，在 override 方法内部抛出或捕获的任何异常必须被在方法内捕获并处理

B `RuntimeException` 一定不会出现在方法定义的 `throws` 关键字之后，也一定不会出现在方法内的 `throw` 关键字之后

C 如果某个方法中包含了 `throw new xxException(...)`这样的代码，则该方法的 spec 种一定会包含 `throws xxException`

D 某行代码 `w` 调用了一个方法 `methodA`，若 `methodA` 的 spec 中使用 `throws` 抛出了一个 checked exception，那么该行代码 `w` 必须要被包含在 `try-catch` 块中

正确答案：A

答案解析：

A 选项：如果不理解它为何正确，考虑 LSP

B 选项：RuntimeException 是 unchecked 异常，但这跟 throws 和 throw 关系词没关系，可以出现在它们后头，表示抛出异常。

C 选项：除非 xxException 是 checked exception，否则不需要出现在方法 spec 中

D 选项：不一定，此行代码所在的方法可以继续将捕获到的该异常往其调用的 client 端抛出，即 w 所在的方法 spec 中也有 throws 同样的异常

这道题目比较晦涩，各位仔细体验一下。（虽然这里描述的大部分规则都是由 static type checking 帮你做了，但是你还是要理解其原理）

单选题 第 6 题 1 分

以下场景最适合使用 checked exception 的是___

A 某个方法中，判断 client 传递进来的参数是否符合方法的 `pre-condition`

B 某个方法中，在 `return` 语句之前，判断该方法所在类的 RI 是否满足

C 在读取文件的过程中，发现磁盘上找不到该文件

D 访问数组时越界，例如 `a[-1]=0`

正确答案：C

答案解析：

RI, pre-condition, post-condition，最好的处理方式是 assertion，目的是 fail fast，因为这是编程错误（要么是 client 的错，要么是 developer 的错）

数组越界，也是同样的道理

只有 C 选项，不是程序员的锅，那就老老实实 try/catch 吧。

多选题 第 7 题 2 分

关于 Java `try/catch/finally` 的语法，以下说法不正确的是__

A 如果 `try{...}`中第 2 行代码抛出了异常，则第 2 行之后至 try 的“}”之间的代码不会再被执行

B 如果 `try{...}`中的代码抛出了异常但没有被后面的任何 `catch{...}`所捕获，则 `finally{...}`中的语句不会被执行

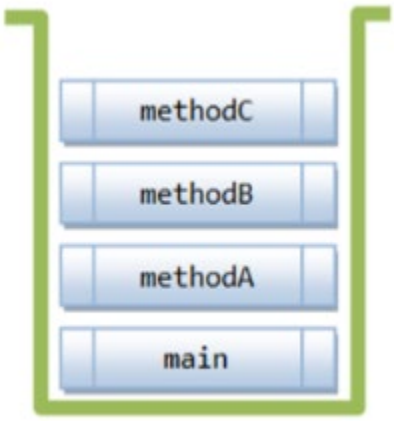
C `finally{...}`中的语句不能再抛出并捕获异常了，即其中不应再出现 `throw/try/catch` 语句

D 如果采用 `try(...){...}`的语法，则不再需要 `finally{...}`了

正确答案：BC

单选题 第 8 题 1 分

某时刻的 JVM stack 如右图所示，且 C 方法执行中抛出一个 checked 异常 XX。以下说法不正确的是__



A 若 C 未捕获未处理 XX，则 XX 被传递给 B

B 若 C 未捕获未处理 XX，则 B 的方法声明中有 throws XX 的字样，则 XX 被传递给 A

C 若 A、B 的方法声明中都没有 throws XX 的字样，则 XX 在 B 中被捕获被处理

D 若该异常 XX 被传递给 main 且 main 也没处理它，则程序将被终止，console 打印出血红的 stack trace

正确答案：C

多选题 第 9 题 2 分

```
try {
        ...
} catch(XXException e) {
        e.printStackTrace();
        print(e.getMessage());
}
```

针对上述代码的 code review，说法正确的是__

A 很正常的代码，用 `try/catch` 捕获和处理 checked 异常

B `printStackTrace` 和 `print` message 有重复，一个就 OK 了

C 意义很小，除了打印输出 stack trace 和错误提示，没有对异常做任何恢复处理

D 如果没法恢复这个 XX 异常，最好在 print 下一行加上 `throw YYException(...)`，YY 是个 unchecked 异常

正确答案：CD

单选题 第 10 题 1 分

以下关于异常处理的“位置”的说法，恰当的是__

A 如果某行代码抛出 checked 异常，最好在该代码所在的方法里捕获该异常并进行恢复处理

B 如果某行代码抛出 checked 异常，最好不在该方法里处理它，而是将其 throws 给调用该方法的 client 加以处理

C 如果某行代码抛出 checked 异常，如果想要 client 采取其他行动才能恢复异常，则不要在该代码所在方法里处理

D 如果某行代码抛出 unchecked 异常，但你不想让程序直接退出，最好应该将其 throws 给调用该方法的 client 加以处理

正确答案：C

答案解析：

关键看恢复 checked 异常需要什么信息、需要提示给用户什么信息。如果这些信息在本方法里无法获取，或者本方法里能获取的与异常相关的信息对用户来说无价值，则将异常往上传递。

Unchecked 异常，最好不要 throws 向上传播。它代表着你程序代码的问题，现场解决（赶紧修改代码来消除它）。——想想你们体育课系统里的 ArrayIndexOutOfBoundsException，居然在 UI 中出现了

7-3 断言与防御式编程

多选题 第 1 题 2 分

以下_是 defensive programming 的有效手段

A 利用 Java 的 static type checking

B 尽可能让 ADT 为 immutable 的，rep 为 `final` 和 `private`

C 在方法的第一行进行参数合法性检查，是否符合 spec 里的 pre-condition

D 在方法的 `return` 之前调用 `checkRep()`

E 使用 unchecked exception 发现代码中的潜在问题

F 先写 spec，再设计测试用例，最后写代码，每次代码更新都 regression test

正确答案：ABCDEF

多选题 第 2 题 2 分

以下关于 assertion 的说法，不正确的是_

A assert 主要用于开发阶段，帮助开发者发现代码中潜在错误

B 在代码中某位置加入 assert 语句，用于判定程序运行到此应满足的条件。若为假，意味着代码出错

C assert 后面可以跟一个参数（判定条件），也可以有第二个参数（message）

D 若代码中的 assert 语句太多，会严重影响代码运行性能，所以要慎重使用

E assert 语句一旦 false，程序绝对无法恢复，只能退出

正确答案：D

多选题 第 3 题 2 分

以下__场合适合于使用 assertion?

A `checkPep()`

B 检查 pre-condition 是否违反

C 检查 post-condition 是否违反

D 在程序流程不应该执行到的地方

E 检查外部文件是否存在

F 检查用户通过键盘的输入数据是否合法

G 检查网络是否连通

正确答案：ABCD

单选题 第 4 题 1 分

关于 Assertion 和 Exception 的异同之处，正确的是___

A 断言是为了应对 robustness，异常处理是为了应对 correctness

B 如果开发者能预料到某些特殊情况可能发生，对其用断言加以应对

C 如果开发者希望某些特殊情况绝不能发生，对其用异常处理加以应对

D 开发阶段用断言尽可能消除 bug，在 release 版本里用异常处理机制处理漏掉的错误

正确答案：D

多选题 第 5 题 2 分

从防御式编程的角度看，如果 client 调用某个方法的时候输入了不符合 pre-condition 的参数，以下做法不恰当的是__

A 按照 spec 的定义，开发者可以采取任何行动来处理不合法的输入参数

B 对不合法的输入参数做智能化处理，使之转化为合法的输入参数，再继续计算

C 直接使用 assert 语句将不合法的输入参数挡在之外，给 client 呈现错误信息提示

D 抛出异常，向 client 展示参数为何不合法、何为合法参数

正确答案：AB

填空题 第 6 题 2 分

用于使 ADT 具备迭代遍历功能并对 client 提供迭代器对象的 Java 接口或抽象类是 [填空 1] 和 [填空 2] 。

只需填写接口或抽象类名，无需带包名。

正确答案：

Iterator;Iterable

填空题 第 7 题 2 分

用于支持实现观察者模式的 Java 接口或抽象类是 [填空 1] 和 [填空 2]

只需填写接口或抽象类名，无需带包名。

正确答案：

Observer;Observable

7-4 Debugging

单选题 第 1 题 1 分

Bug 这个术语是_最先使用的?

A Ada Lovelace

B Barbara Liskov

C Grace Hopper

D Margaret Hamilton

正确答案：C

单选题 第 2 题 1 分

关于 assertion, exception handling, debugging, testing 这四种技术之间的关系，不恰当的说法包括__

A 利用断言进行防御式编程，发现代码中的潜在错误；无法预防的问题，通过异常处理在运行时解决

B 防御式编程无法规避所有 bug，需通过测试发现其他 bug

C 通过 assertion、checked exception 和 testing 找出的 bug 需通过 debug 进行错误定位并消除之

D 这四种方法都不是万能的，交付的代码中仍然可能隐藏着大量的 bug

正确答案：C

答案解析：

Checked exception 是能预料发生并能处理的，不算是 bug。Unchecked exception 是用来帮忙找出不希望发生的情况，即代码里的 bug。

多选题 第 3 题 2 分

为何 debug 占用了几乎 50%的开发时间，以下__是可能的原因?

A 出错的症状所在的代码行跟造成出错的“原因”代码行可能相隔很远，甚至不在同一个源文件

B 由计时/定时等原因导致的某些 bug，时间不同了，bug 出现的症状就可能不同

C 多线程程序中因为线程交错执行（interleaving）的随机性，导致每次执行的次序不同，bug 症状可能时隐时现

D 因为外部的动态性，导致多次执行时难以复现完全一样的执行环境或者用户输入

正确答案：ABCD

单选题 第 4 题 1 分

在代码中可能出现的代码行之后加入 `print` 或 `log` 语句，输出某些变量的值，根据这些值判断是否有 bug。该 debug 策略称为__

A Instrumentation

B Divide and Conquer

C Program Slicing

D Delta Debugging

正确答案：A

单选题 第 5 题 1 分

两个测试用例，测试结果分别是“通过”和“不通过”。

那么，利用 EclEmma 代码覆盖度工具观察两个测试用例所覆盖的代码行，找出二者的差异，bug 很可能就隐藏在差异的代码行里。

该 debug 策略称为__

A Instrumentation

B Divide and Conquer

C Program Slicing

D Delta Debugging

正确答案：D

单选题 第 6 题 1 分

如果发现某函数输出的某变量 a 与期望不一致，那么对该函数代码中可能导致变量 a 的值发生变化的一部分代码进行仔细分析，以发现潜藏 bug。该 debug 策略称为___

A Instrumentation

B Divide and Conquer

C Program Slicing

D Delta Debugging

正确答案：C

单选题 第 7 题 1 分

通过各种 debug 策略找到 bug 根源之后，对 bug 进行修复。以下说法最不恰当的是__

A 在修改代码之前，要将之前的代码 commit 到仓库里

B 每次最好只针对一个 bug 进行代码修改

C 针对一个 bug 修改代码之后，要重新运行之前发现该 bug 的所有测试用例，已确认该 bug 是否被完全修复

D 针对一个 bug 的代码修改，如果没有彻底修复该 bug，最好不要将修改的代码 commit 到仓库

正确答案：C

多选题 第 8 题 2 分

以下__能够为开发者发现 bug 根源所在提供有效帮助

A git diff v1 v2

B Memory heap dump

C Print stack trace

D `System.out.println()`

E `java.util.logging`

F Set breakpoints and use step-by-step debugger

G End users’ bug report

正确答案：ABCDEFG

8-2 内存性能与垃圾回收

多选题 第 1 题 2 分

关于内存管理的三种模式（static、stack、heap），不正确的是__

A Static 在编译阶段为各变量分配内存，不支持运行时变量扩展内存，但支持运行时为新变量分配内存

B Stack 因为结构简单，不支持存储结构复杂的变量（例如数组、对象等）

C 在 heap 中分配内存的变量，其内存大小在运行时可动态扩展或收缩

D Static 和 stack 两种模式都不支持对变量所占内存的动态回收

E Java 中 heap 用于复杂结构的数组或对象等变量的内存分配，而简单数据类型的变量在 stack 中分配内存

正确答案：ADE

单选题 第 2 题 1 分

类 A 有一个静态成员变量 int a 和一个非静态成员变量 Date b，它的某个方法内使用了一个局部变量 Calendar c。JVM 在为 a、b、c 三个变量分配内存时，分别在____中分配

A Stack / Stack / Heap

B Stack / Method Area / Stack

C Heap / Stack / Stack

D Method Area / Heap / Heap

正确答案：D

多选题 第 3 题 2 分

关于 GC 的说法，不正确的是_

A GC 根据对象的“活性”（从 root 的可达性）来决定是否回收该对象的内存

B 如果某个对象内部不再有指向其他对象的 reference，则该对象一定是 dead，会被 GC

C 在 Java 中，GC 完全依赖于 JVM，程序员无法通过代码来主动改变某个对象的“活性”

D Defensive copy 策略大大增加了 GC 的负担，相当于用性能换取安全性

E 程序员在代码中人工实现 GC，若不恰当，可能导致内存泄露

正确答案：BC

单选题 第 4 题 1 分

Java 中针对 heap 的各种 GC 策略，以下说法不正确的是_

A Mark-sweep 检查内存中各对象是 live 还是 dead，对 dead 对象做出标记，进而将其清理

B 每次进行 Mark-compact 策略的 GC 之后，heap 中可被再分配的区域是连

续的

C Copy 是四种策略中唯一不需要检查对象 live/dead 的 GC 策略

D Reference counting 相对于其他三种 GC 策略，内存中 dead 对象的 zombie time 最短

正确答案：C

单选题 第 5 题 1 分

针对 JVM 内存管理和 GC 的说法，不正确的是__

A 相对于 old generation space 中对象的存活时间，young generation space 中对象的存活时间更短

B 对存活时间非常长的对象，经历多次 GC 后，在 heap 中位置的迁移路线是 eden→(S0 或 S1)→(S1 或 S0)→...→old generation space

C Old generation space 的 GC 是 full GC，当 young generation space 满了的时候触发执行

D 若针对 young generation space 的 minor GC 的发生频度很高，则需扩大该区域的内存大小

正确答案：C

单选题 第 6 题 1 分

```
-Xms512m
-Xmx1024m
-XX:MetaspaceSize=128m
-XX:MaxMetaspaceSize=192m
-XX:NewSize=128m
-XX:MaxNewSize=256m
```

```
-XX:SurvivorRatio=8
-XX:MaxHeapFreeRatio=90
```

按上述参数配置，Old Generation 和(S0+S1)的最大值分别是__

A 1024m, 25.6m

B 768m, 51.2m

C 896m, 51.2m

D 576m, 25.6m

正确答案：C

单选题 第 7 题 1 分

```
-Xms512m
-Xmx1024m
-XX:MetaspaceSize=128m
-XX:MaxMetaspaceSize=192m
-XX:NewSize=128m
-XX:MaxNewSize=256m
-XX:SurvivorRatio=8
-XX:MaxHeapFreeRatio=10
```

- 1 按上述参数配置，若当前 young 和 old generation 的尺寸分别为 128m 和 384m，在以下_的时刻，JVM 会自动增加 young generation 的尺寸?

A S0+S1 的总占用超过 96m

B Eden 的总占用超过 56m

C Old generation 的占用超过 350m

D S0+S1+Eden 是总占用超过 120m

正确答案：D

多选题 第 8 题 2 分

```
-XX:+DisableExplicitGC
-XX:+PrintGCDetails
-XX:+UseConcMarkSweepGC
-XX:ParallelCMSThreads=12
-verbose:gc
-XX:+HeapDumpOnOutOfMemoryError
-XX:+PrintGCTimeStamps
-Xloggc:../../logs/gc-console.log
```

按以上参数配置，以下不正确的是_

A 程序运行时可将 GC 过程计入日志文件，并记录历次 GC 时间戳

B 使用串行的 GC 策略进行 GC，GC 时会短暂停止程序执行

C 开发者可手工在程序代码中使用 System.gc()以提升 GC 性能

D 使用 12 个 GC 线程，降低了 GC 对程序执行性能的影响

正确答案：BC

8-4 动态性能分析方法与工具

多选题 第 1 题 2 分

以下关于 Dynamic Program Analysis 的说法，不正确的是___

A 根据程序执行的过程与结果，分析代码在时空性能方面所展现出的性质

B 对程序执行的性能没有影响

C 可用来发现程序中的“热点”语句，即哪些语句被频繁执行

D 只需执行一次，即可比测试更容易发现程序中的性能弱点和 bug

E 能够发现程序中各种不同类型的 object 分别占用了多少内存空间

正确答案：BD

多选题 第 2 题 1 分

关于 profiling 的三种策略，说法不正确的是__

A 采用代码注入的策略，对程序性能的度量最为准确

B 采用 Instrumented VM 的策略，需针对不同的 VM 分别使用不同的 profiling 工具

C 采用 Sampling 的策略，对程序执行性能的影响最大

D 采用 Instrumented VM 和 Sampling 的策略，均不需对程序代码进行修改

E 采用代码注入的策略，不可能对目标代码进行动态注入

正确答案：ACE

多选题 第 3 题 2 分

关于 Java 提供的若干 profiling 工具，说法不正确的是___

A 有工具可以获取 Java 程序运行时 JVM 管理的各 heap 区域的动态占用情况

B 有工具可以得知指定的 Java 程序所采用的 GC 策略

C 有工具可以对正在运行的 Java 程序的 JVM 内存配置进行参数的动态设置

D 即使不使用这些工具，当 Java 程序抛出 `OutOfMemoryError` 时，JVM 也能够自动导出内存溢出时刻的 heap dump

E 有工具可以获取当前时刻 Java 程序主线程的 call stack 的状态

正确答案：CD

单选题 第 4 题 1 分

```
jstat -gcutil 21891 250 7
  S0     S1     E      O      M     CCS    YGC
  0.00  97.02  70.31  66.80  95.52  89.14      7
  0.00  97.02  86.23  66.80  95.52  89.14      7
  0.00  97.02  96.53  66.80  95.52  89.14      7
 91.03   0.00   1.98  68.19  95.89  91.24      8
 91.03   0.00  15.82  68.19  95.89  91.24      8
 91.03   0.00  17.80  68.19  95.89  91.24      8
 91.03   0.00  17.80  68.19  95.89  91.24      8
```


从该结果看，在第一次 GC 前后，suivivor space 的占用比例增加了__，old generation 的占用比例增加了_

A `91.03%, 1.98%`

B `-5.99%, 1.39%`

C `-94.55%, -97.02%`

D `91.03%, 0.37%`

正确答案：B

多选题 第 5 题 2 分

不能够导出某个 Java 进程的 heap dump 文件的是__

A `jmap`

B `jcmd`

C `jconsole`

D `jstack`

E `jhat`

F Eclipse Memory Analyzer

正确答案：DEF

多选题 第 6 题 2 分

使用 profiling 工具来监视你的 Lab3 程序时，发现 heap 中出现了大量的 PhysicalObject 对象实例，占用了大量内存。可能的原因是___

A 你的 Lab3 将 `PhysicalObject` 设计为 `mutable`，并对其采取了防御式拷贝策略以避免表示泄露

B 本次运行读入了一个大文件，故构造 `CircularOrbit` 对象时不得不构造大量 `PhysicalObject` 对象

C 导出 heap 的时刻之前较长时间没有进行 GC，故大量无活性的 `PhisicalObject` 仍处于内存中

D 你的 Lab3 对 `PhysicalObject` 的生成（new）采用了“静态工厂”模式，导致 JVM 无法获取各 `PhysicalObject` 对象的活性而无法及时 GC

正确答案：ABC

多选题 第 7 题 2 分

对代码进行 dynamic profiling，不需要在__时候进行

A ADT 的初始版本完成后（包括完成了 Rep、方法、AF、RI、Spec、各方法的代码）

B ADT 测试完成后（根据 spec 设计测试用例，用 JUnit 执行测试用例并获得结果后）

C 每次向 Git 进行一次 commit 之前

D ADT 迭代开发结束，除性能之外的其他外部和内部质量指标的优化均已经达到期望

E 交付用户之前，发现程序运行缓慢，与期望不符

正确答案：ABC

8-5 面向性能的代码调优

多选题 第 2 题 2 分

以下关于代码调优的说法，不正确/不恰当的是__

A 代码行数越少，代码的执行性能倾向于更好

B 每写完一个 method 的代码，最好对其性能进行优化，确保时空复杂性优化

C 直到软件开发完全结束、所有其他质量指标均已满足期望，再进行代码调优

D 每次进行代码调优前，必须要使用 profiling 工具进行性能监控和度量

E 每次代码调优之后、修改代码提交 Git 仓库之前，都需要进行 regression testing

正确答案：AB

多选题 第 2 题 2 分

关于 Singleton 设计模式，说法不正确/不恰当的是__

A 符合该模式的 ADT，不应具有 constructor，以避免 client 创建多个实例

B 符合该模式的 ADT，其 rep 中应具有一个 `static` 的 field，其类型是该 ADT 本身

C 符合该模式的 ADT，应具备一个方法 `getInstance()`，在该方法内进行该

ADT 的 new 操作

D State 设计模式中，每个 State 的具体子类，可遵守 Singleton 模式进行实现

正确答案：AC

答案解析：

A 选项：每个类都有 constructor，只不过该模式下的 constructor 是 private 的，不能让 client 调用

C 选项：除非用 lazy load 的策略，否则 getInstance 方法中只需直接返回 rep 中的单例对象

多选题 第 3 题 2 分

使用 Flyweight 模式实现文本编辑器中对各个“字母”对象的复用，以下说法不正确的是__

A “字母"的内容（例如 a、b）是内特性，“字母"在编辑器中展示的字体、字号、颜色是"外特性”

B 即使"字母” 对象的字体/字号/颜色可变化，但"字母" ADT 仍应该是 immutable 的

C "字母"ADT 的 rep 种需包含其内容、颜色、字体、字号等属性

D 只需要 26 个大写字母和 26 个小写字母的对象实例，即足够 client 在编辑器中使用"字母"的功能

E 客户端代码中需要维护某些数据来管理或计算每个"字母"对象在文档中不同位置的字体、字号、颜色等。

正确答案：C

多选题 第 4 题 2 分

关于 Prototype 模式，说法不正确的是__

A 遵循该模式的 ADT，要实现 `Cloneable` 接口并 override `Object` 的 `clone()`方法

B `Object` 的 `clone()`方法，缺省实现的是 shallow copy 而非 deep copy

C 若在调用 ADT 的 `clone()`时抛出 `CloneNotSupportedException`，则意味着该 ADT 没有 `override` `Object` 的 `clone()`

D 在进行对象构造时，相比于直接使用构造器来说，使用 `clone()`的代价较小、构造时间更短

正确答案：CD

单选题 第 5 题 1 分

关于 Object Pool 的说法，不正确的是_

A 不采用该模式时，如果一个对象实例不再有活性，将对象实例加入 pool，相当于强行保持其活性而不会被 GC

B 当 client 需要一个对象实例时，先到 pool 中去获取，使用完之后，再归还回去

C 原本可被 GC 的对象，现在要留在 pool 中，导致内存浪费，但节省了频繁的 new 和 GC 的时间

D 不能把不同类型的对象实例放在同一个 pool 中进行管理，需要分别设定不同的 pool

正确答案：D

多选题 第 6 题 2 分

关于 Singleton、Flyweight、Object Pool 设计模式之间的异同，说法不正确的是__

A Singleton 相当于把每个 class 做成了 pool，其中包含唯一的对象实例，且是 static 的

B Flyweight 相当于只使用一个对象实例来表示一组具有相同内属性但不同外属性的对象

C Flyweight 模式维持一个 pool，pool 中包含一组具有不同内属性的对象实例

D 这三种模式的 client 使用某个对象实例时，均需要从 pool 中申请获得实例、用完归还给 pool

E 目的都是为了降低 new 对象实例时的时间代价，降低了程序运行时内存消耗

正确答案：DE

多选题 第 7 题 3 分

以下__能够减少创建 object 的数量，降低 GC 的代价

A 方法中的临时变量尽可能使用 primitive 数据类型（e.g., double），减少使用其对象数据类型（e.g., Double）

B 除非迫不得已（ADT 安全性要求），不要使用防御式拷贝

C 对频繁使用的对象用 object pool 进行 canonicalization，哪怕是类似于 Integer 这样的“小”对象

D 尽量使用 `String a = "foo"`的方式来定义常量字符串，避免 `new String("foo")`

E 尽可能使用类的静态工厂方法进行对象创建，避免直接用 new

F 在为 ADT 设计 rep 的时候，除非迫不得已，最好用简单数据类型

正确答案：ABCDEF

10-1 Concurrency

多选题 第 1 题 2 分

以下_是计算机系统中的 concurrency 现象?

A 手机上的一个 App 通过 5G 网络访问云端数据

B 四核 CPU，执行多道程序

C 使用 Observer 设计模式的 Java 程序，其中 Subject 类和 Observer 类的执行

D 一亿人同时登陆 12306 网站抢购春运火车票

E 使用 `JVM` 参数`-XX:+UseConcMarkSweepGC`启动的程序，运行时进行 GC

F 同一个 Java 程序内的两个线程，共享一个 mutable 的 Java 对象

正确答案：ABDEF

多选题 第 2 题 2 分

关于 process 和 thread 的说法，不正确的是_

A 多个 process 之间不共享内存，而多个 thread 之间可共享内存

B CPU 的某个核，在特定时间点只能运行单一 process，但可并行执行多个 thread

C 手机上的 App 通过 5G 网络访问云服务器的资源，手机上和服务器上运行的是不同的 thread 而非不同的 process

D 一个 process 可以包含多个 thread，一个 thread 只能在一个 process 里运行

正确答案：BC

多选题 第 3 题 2 分

关于如何创建 thread 的说法，不正确的是__

A 从 `Thread` 类派生子类 `A`，创建线程时`(new A()).start()`

B 类 `A` 实现 `Runnable` 接口，并实现其 `run()`方法，创建线程时`(new Thread(new A())).run()`

C

```
new Thread(new Runnable(){
        public void run() {...};
}).start();
```

D

```
new Thread(new Runnable(){
        public void run() {...};
}).run();
```

正确答案：BD

多选题 第 4 题 2 分

关于 time slicing, interleaving 和 race condition 的说法，正确的是__

A Time slicing 由 OS 决定，但程序员可在代码中进行若干有限度的控制

B 如果某程序执行结果的正确与否依赖于 time slicing，那么意味着程序执行中产生了 race condition

C 程序 interleaving 执行的基本单元是 Java 代码行

D 同一个并发程序的多次执行中的 time slicing 可能完全不同，因此 bug 很难复现，将此类 bug 形象的成为 Bohrbugs

正确答案：AB

多选题 第 5 题 2 分

关于 Java Thread 的 `sleep()`和 `interrupt()`，不正确的是__

A 若线程的 `run()`代码中包含 `Thread.sleep(100)`，意味着该线程停止执行 100ms，CPU 交给其他线程/进程使用

B 线程 t1 中包含代码 `t1.interrupt()`，意味着执行完该语句后 t1 被停止，不会再获得 time slicing

C 线程 t1 在 `sleep()`期间可捕获到其他线程发来的中断信号并抛出 `InterruptedException`异常

D 线程若捕获了抛出 `InterruptedException`异常，则自动终止执行

正确答案：BD

多选题 第 6 题 2 分

```
Thread t = new Thread(new Runnable(){
        public void run(){
                try{
                        print("a");
                        ...
                        Thread.sleep(200);
                }catch(InterruptedException e){
                        print("b");
                }
                print("c");
        };});

t.start();
...
t.interrupt();
```

某类的 `main()`代码如上所示，执行后可能的输出结果为___

A ab

B ac

C bc

D abc

正确答案：BC

//更正：BD

填空题 第 7 题 3 分

如果希望让线程 t 执行结束之后再执行其他线程，即让其他线程暂停，需要调用 `t.[填空 1]();`

用于“检测当前线程是否收到其他线程发来的中断信号”的 Thread 静态方法是 `Thread.[填空 2]()`

在线程类的 `run()`方法中有以下代码，如果希望收到中断信号后终止线程执行，则 TODO 位置的语句应该是 [填空 3]

```
try{
        ...
        Thread.sleep(1000);
        ...
} catch(InterruptedException e){
        TODO;
}
```

正确答案：

join;interrupted;return

10-2 线程安全

多选题 第 1 题 2 分

以下关于 ADT ThreadSafe 的说法，不正确的是__

A 任何对外发布的 ADT 都必须要做到 thread safe

B 不管怎么 time slicing 和 interleaving，一个线程安全的 ADT 对象的 RI 要始终为真

C 在不同的机器上、不同的 OS 上执行关于该 ADT 的程序，都不应该出现 RI 被违反的情况

D 任何 immutable 的 ADT 都是 threadsafe 的

E 做到 ThreadSafe 与否，只与 ADT 自己的内部 rep 和内部方法实现有关，与 client 是否应遵循特定条件有关

正确答案：AD

答案解析：

A 选项：不一定，例如 JDK 里就有不少在多线程程序里不安全的 ADT。这种情况可以在 spec 里写清楚即可。但是，最好提供与之相对应的 threadsafe 的 ADT。

B、C、E 选项都是在表达 ThreadSafe 的一个重要约束：threadsafe 与你的程序在什么软硬件环境下运行、被谁的程序所调用，没有关系。

D：beneficent mutation

多选题 第 2 题 2 分

关于 Strategy 1: Confinement 的说法，不正确的是__

A 多线程之间不要共享 mutable 的数据对象

B 各线程内部不应使用 mutable 的数据对象

C 只要 ADT 的 rep 里存在 static 类型的变量，该 ADT 在任何多线程场景下都无法做到 threadafe

D 只要 ADT 的 rep 里存在 mutable 类型的变量，该 ADT 在任何多线程场景下都无法做到 threadsafe

正确答案：BCD

答案解析：

A 选项是该策略的核心思想

B 选项：线程内部当然可以用 mutable 的对象，只要它们被限定只在该线程中使用（不和别的线程共享，局限在自己的 thread stack 中），即符合该策略

C 选项：如果该 static 属性是 immutable 的且不存在其他 mutable 的属性（或者这些 mutable 的属性通过其他策略确保的安全性），那么可以做到线程安全

D 限选：见 C 选项的解释

多选题 第 3 题 2 分

针对 Strategy 2: Immutability，不正确的说法是__

A 该策略思想是：多线程之间共享数据时，使用不可变数据类型和不可变引用，以避免多线程之间的 race condition

B 如果多线程共享的是 mutable 的数据类型，可以通过在线程中禁用其 mutator 方法来达到 threadsafe

C 如果 ADT 的 rep 里所有属性都是 `final` 的，那么它在任何多线程场景下都可以做到 threadafe

D 如果 ADT 的 rep 里存在 `public` 类型的属性，那么它就无法确保做到 threadsafe

正确答案：BC

答案解析：

B 选项：threadsafe 不能要求 client 作什么，一定是你 ADT 的职责

C 选项：光 final 还不够，指向的对象还得是 immutable 的。否则多线程就可以对其值的修改产生 race condition

D 选项：public 属性意味着 rep exposure，那线程就可以修改其 rep，就可能产生 race condition

多选题 第 4 题 2 分

针对 Strategy 3: Using Threadsafe Data Types 策略，不正确的是__

A 如果必须要用 mutable 的数据类型在多线程之间共享数据，要使用线程安全的数据类型

B 若在 ADT 的 rep 里使用集合类，最好使用

`SynchronizedMap/Syn..List/Syn..net`

C 若能做到 B 选项中的说法，该 ADT 在任何多线程场景下都可以做到 threadsafe

D 即使在线程安全的集合类上，使用 iterator 也是不安全的

正确答案：C

单选题 第 5 题 1 分

```
public class P{
        private static P p = null;
        // invariant: only one P object
        // should be created

        public static P getInstance() {
1)      if(p == null) {
2)              p = new P();
                }
3)      return P;
        }
        ...
}
```

有两个线程，其 `run()`方法都是在调用 `getInstance()`。

如果线程 `1` 正在执行语句 `1`，线程 `2` 正在执行语句 `3`，那么两个线程之间是否会产生 `race` 并导致 `invariant` 被违反?

A YES

B NO

正确答案：A

多选题 第 6 题 2 分

```
public String t = new String();
private List<String> ol = new ArrayList<>();
private final List<String> ls = Collections.synchronizedList(ol);
```

某 `ADT` 的 `Rep` 如上所示，该 `ADT` 在被用于多线程场景时的 `threadsafe`，以下说法不正确的是___

A `t` 不存在线程安全风险，因为它是 `String` 类型，是 `immutable` 的

B `ol` 存在线程安全风险，因为 `List<>`不是线程安全的数据类型

C `ls` 不存在线程安全风险，因为它用 `synchronizedList` 做了封装

D `ls` 存在线程安全风险，可能在该 `ADT` 的某方法中存在对 `ls` 的多个方法的调用，可能产生 `race`

正确答案：AC

多选题 第 7 题 2 分

线程 `1`:

```
if ( ! lst.isEmpty()) {
        String s = lst.get(0);
        ...
}
```

线程 2：

```
if ( lst.size() >=1) {
        lst.remove(0);
        ...
}
```

其中 lst 是利用 Collections.synchronizedList()得到的 List 对象

A 因为 lst 是线程安全的数据类型，故两个线程不存在 race condition

B 线程 1 的 isEmpty()和线程 2 的 size()之间的 race condition 造成未期望的后果

C 线程 1 的 isEmpty()和线程 2 的 remove()产生 race condition 造成未期望的后果

D 线程 1 的 get()和线程 2 的 remove()产生 race condition 造成未期望的后果

正确答案：CD