**FIT2099: Exam Excerpts**

**Index**

Keeping implementation details of classes private

Repeated code (Risk and Resolution)

JavaDocs for private attributes

Declared Exceptions vs Undeclared Exceptions

Writing Executable Preconditions

Why not to cope with violated preconditions

Access control for methods used in preconditions

Benefits of Writing Preconditions before Implementation

Benefits of Writing Postconditions before Implementation

Command-Query Separation, Relation to Design by Contract

Liskov Substitution Principle and Pre/Post Conditions

Interface in Java

Abstract Class in Java

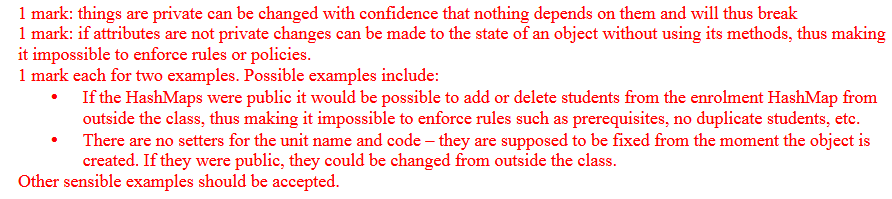
Interfaces vs Abstract Classes

Why Reduce Dependencies

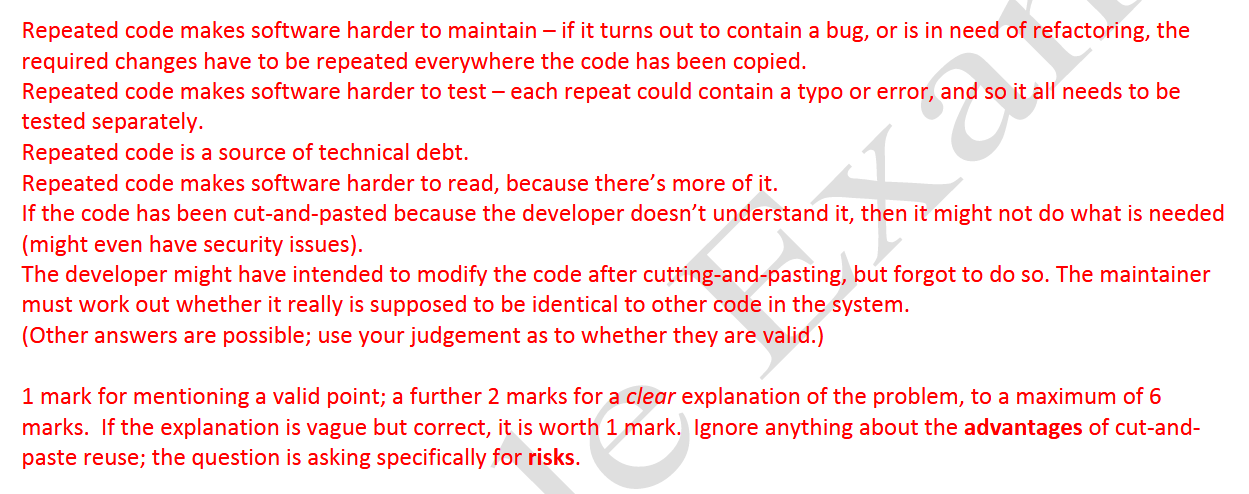
Encapsulation and Access Modifiers to control dependencies

What is good abstraction? Abstraction vs Abstract Classes

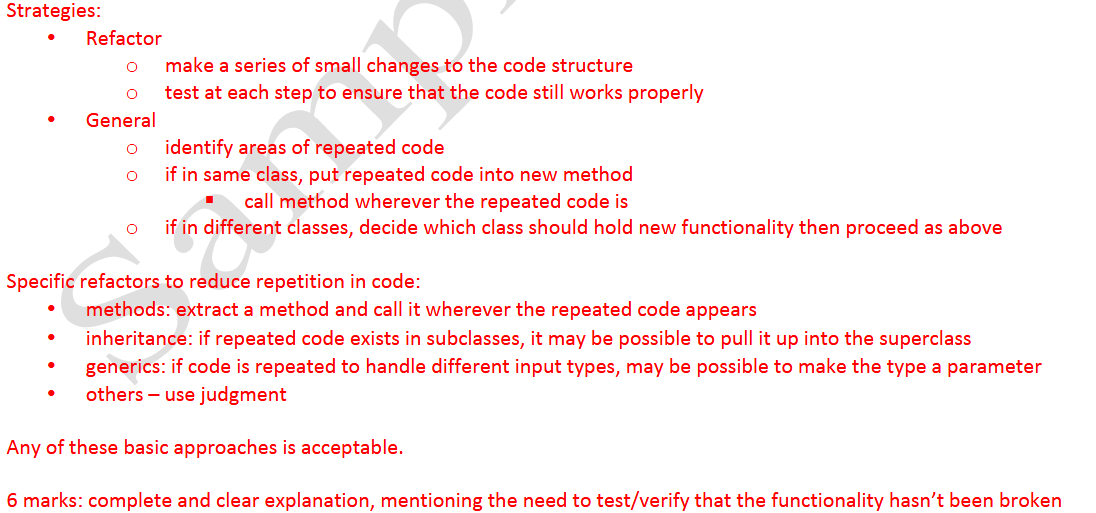
**Keeping implementation details of classes private**

****

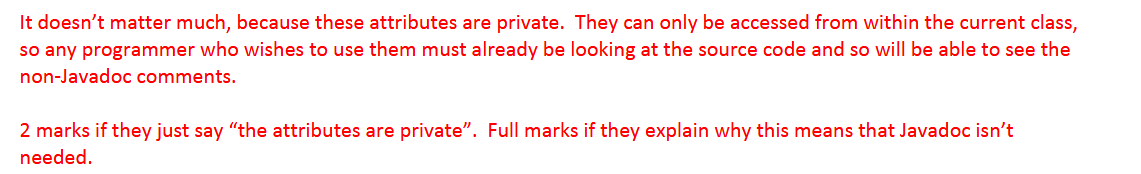
**Risks of repeated code**



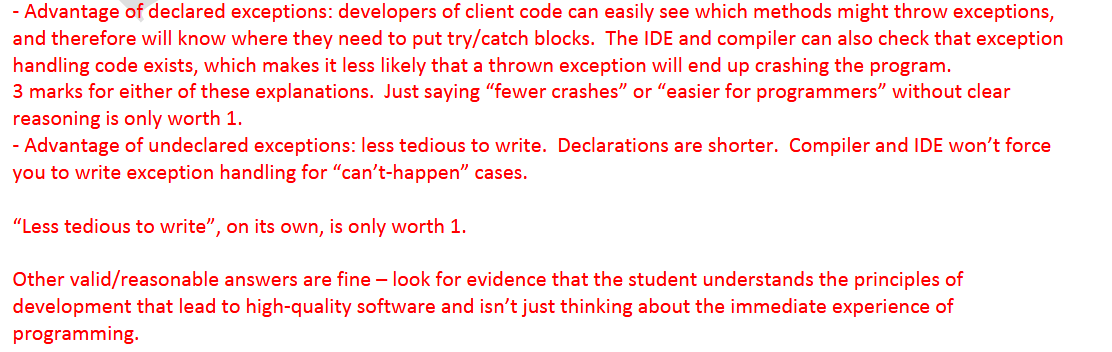
**Fixing repeated code**



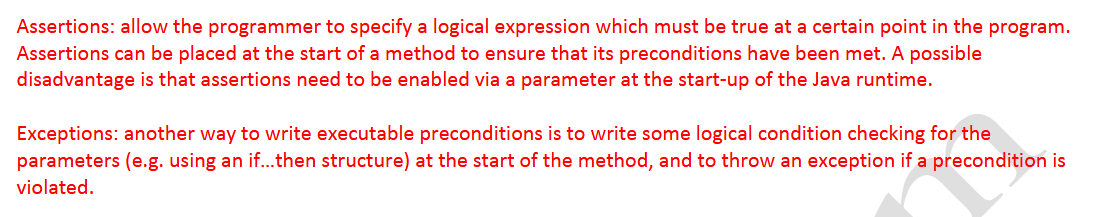
**JavaDocs for private attributes**

****

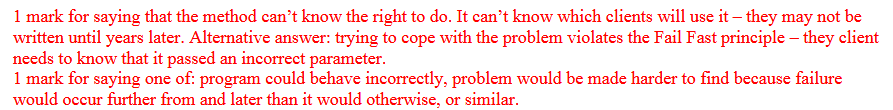
**Declared Exceptions vs Undeclared Exceptions**

****

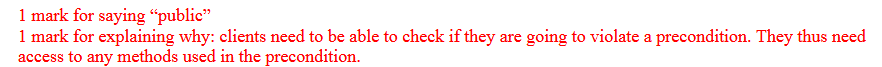
**Writing Executable Preconditions**

****

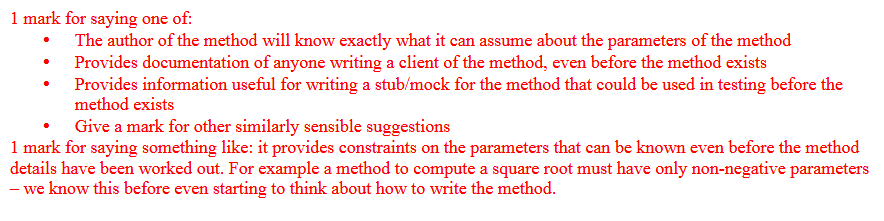
**Why not to cope with violated preconditions**



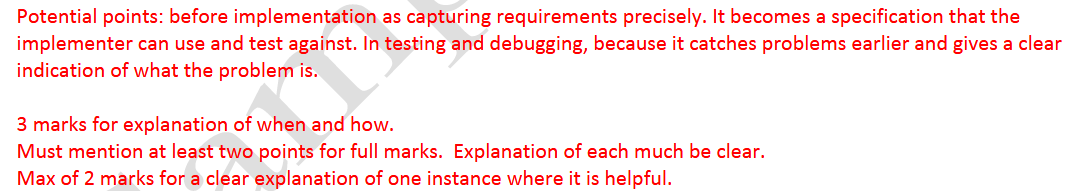
**Access control for methods in preconditions**

****

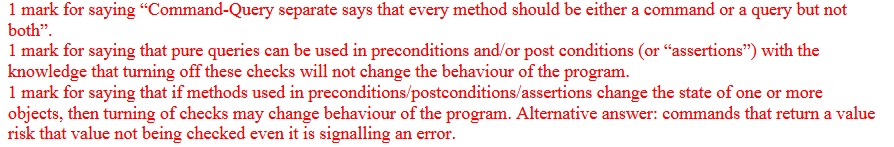
**Benefits of Writing Preconditions before Implementation**

****

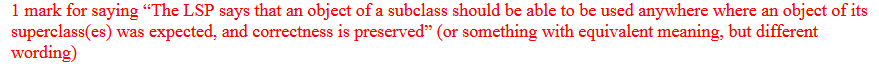
**Benefits of Writing Postconditions Before Implementation**

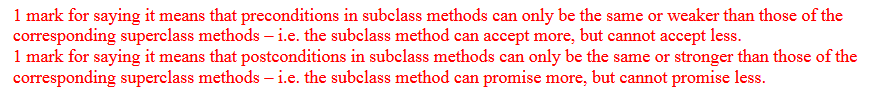
****

**Command-Query Separation, Relation to Design by Contract**

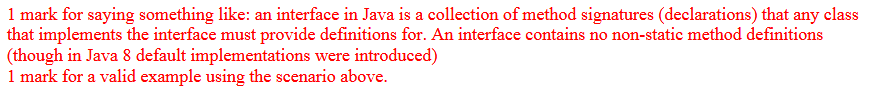


**Liskov Substitution Principle and Pre/Post Conditions**

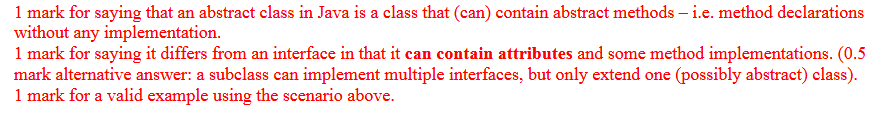
****

****

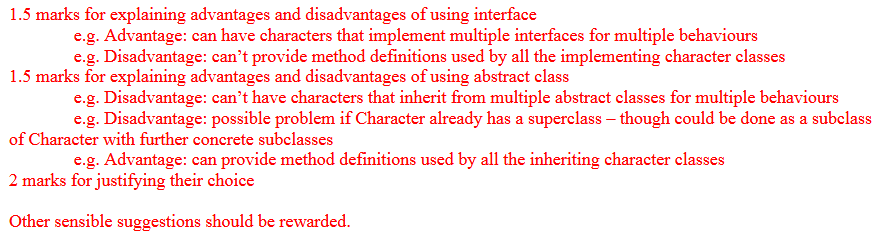
**Interface in Java**

****

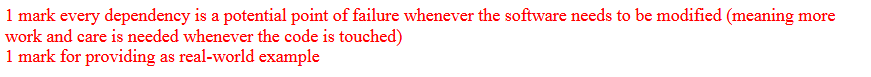
**Abstract Class in Java**

****

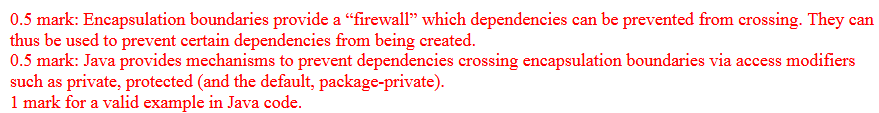
**Interfaces vs Abstract Classes**

****

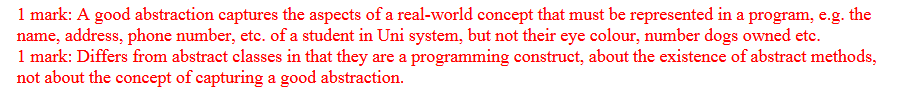
**Why Reduce Dependencies**

****

**Encapsulation and Access Modifiers to control dependencies**

****

**What is good abstraction? Abstraction vs Abstract Classes**

****