Software Engineering 2020/2021
Resit Exam Questions

— Do not turn this page before the official start of the exam! —

First Name, Surname: _____

Student ID: _____

Program: Bachelor Data Science and Artificial Intelligence

Course code: KEN1520

Examiner: C. Seiler and T. Pepels

Date/time: Tuesday, June 29, 2019, 9:30–11:30

Format: Closed book exam

Allowed aides: Pens, simple (non-programmable) calculator from the DKE-list of allowed calculators.

Instructions to students:

- The exam consists of 6 questions on 6 pages (excluding the 1 cover page(s)).
- Fill in your name and student ID number on each page, including the cover page.
- Answer every question at the reserved space below the questions. If you run out of space, continue on the back side, and if needed, use the extra blank page.
- Multiple choice questions that allow for multiple answers: We will award a fraction of the points available for each correct answer selection and subtract an equivalent fraction for incorrect answer selection. The lowest score for a question is 0.
- Ensure that you properly motivate your answers.
- Do not use red pens, and write in a readable way. Answers that cannot be read easily cannot be graded and may therefore lower your grade.
- You are not allowed to have a communication device within your reach, nor to wear or use a watch.
- You have to return all pages of the exam. You are not allowed to take any sheets, even blank, home.
- If you think a question is ambiguous, or even erroneous, and you cannot ask during the exam to clarify this, explain this in detail in the space reserved for the answer to the question.
- If you have not registered for the exam, your answers will not be graded, and thus handled as invalid.
- **Good luck!**

The following table will be filled by the examiner:

Question	Points	Score
Testing	18	
Version Control	16	
Abstraction Functions and Representation Invariants	16	
Software Management	6	
Software Architecture	20	
UML	24	
Total:	100	

Question 1 (Testing) (18 points)

Given this specification:

```
/**
 * @param val value by which this BigInteger is to be divided
 * @return a BigInteger whose value is (this / val)
 */
public BigInteger divide(BigInteger val);
```

(a) (4 points) How many input parameters do you need to consider for testing? Justify your answer.

(b) (10 points) Draw partitions where each input parameter should correspond to one axis. Clearly indicate the partitions and the boundary values between partitions.

(c) (4 points) How many test cases do we need to implement for the Cartesian product testing strategy? Justify your answer.

Question 2 (Version Control) (16 points)

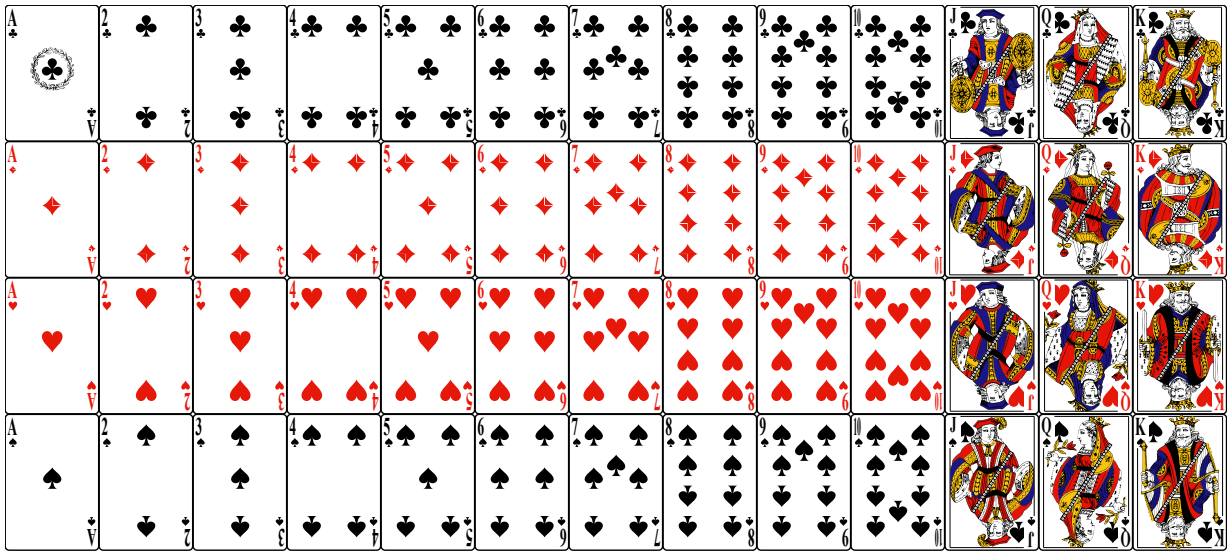
(a) (6 points) Arrange this itemized list according to its correct order, and explain each step in your own words.

- A: Send changes back to remote location using git push
- B: Check out the current version of the master branch
- C: Create new commit out of all the staged changes using git commit
- D: Stage changes using git add
- E: Copy the object graph from remote location to .git folder
- F: Create an empty local directory .git

Step 1: ?
Step 2: ?
Step 3: ?
Step 4: ?
Step 5: ?
Step 6: ?

(b) (10 points) How does git store the history of your source code? What data structure does it use? Make a drawing and explain in words what each component of your drawing means.

Question 3 (Abstraction Functions and Representation Invariants) (16 points)



Consider this Abstract Data Type (ADT):

```
/**
* Represents a card from a deck of poker cards
* A standard 52 card deck has 4 suits: clubs, hearts, spades, or diamonds
* Every suit contains twelve cards with values: ace, 2, ..., jack, queen, or king
*/
public class PokerCard {

    private int suit;  // suits are 0, ..., 3
    private int value; // values are 0, ..., 11

    public Dice(int suit, int value) {
        this.suit = suit;
        this.value = value;
    }
    // rest of the implementation not shown
}
```

(a) (4 points) What is the domain of the abstraction function for this ADT?

(b) (4 points) What is the range of the abstraction function for this ADT?

(c) (4 points) What is the representation invariant for this ADT?

(d) (4 points) How would you implement the representation invariant in Java?

Question 4 (Software Management) (6 points)

The Agile manifesto describes 12 "principles" of software engineering. Look at the statements below, most of them are not related to Agile principles, but two of them are part of the Agile manifesto, mark the **two statements** that are part of the Agile manifesto:

- ◯ Good requirements at the start of a project are key to success.
- ◯ The quality of software depends on the quality of development tools.
- ◯ Working software is the primary measure of progress.
- ◯ A good team bases its client communication on different roles.
- ◯ Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- ◯ Delivered software is Agile if it adheres to strictly predefined requirements.

Question 5 (Software Architecture) (20 points)

(a) (5 points) Which statement below best describes a "service" in Service Oriented and Microservices architectures?

- ◯ It is the model part of a Model, View, Controller architecture.
- ◯ A service is responsible for monitoring the application's requirements.
- ◯ A service is a part of the architecture that implements—as a separate, fully functional entity—a part of the functionality of the software.
- ◯ A service is a piece of software that runs inside a virtual machine, a cloud service, or a Docker file.
- ◯ A service is a REST API that provides a single interface that allows a user to connect to the software.

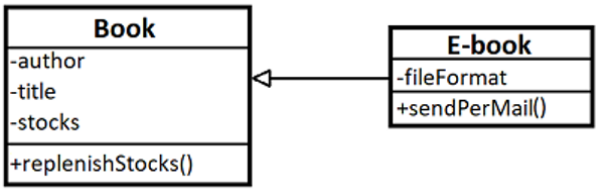
(b) (5 points) In your own words and/or using an example, show how Polymorphism can be used to write a loosely coupled application. If you cannot think of an example application, remember the shapes in the SVG assignment.

(max 5 sentences, you may also write source code with your explanation, source code does not count towards the sentence limit)

(c) (5 points) In your own words, explain how using REST API's in your software increases separation of concerns.

(max 5 sentences)

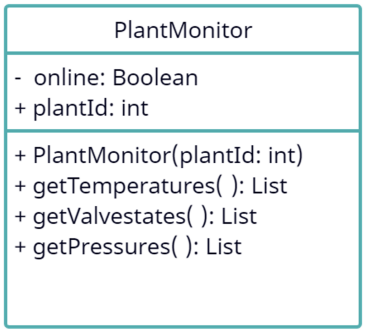
(d) (5 points) Look at the small class-diagram below and note that the E-book class inherits the Book class. Which SOLID principle is violated here and why? The SOLID principles are: Single responsibility, Open-closed, Liskov substitution, Interface segregation and Dependency inversion.



(max 4 sentences)

Question 6 (UML) (24 points)

A large chemical plant needs new a monitoring controller. A plant consists of several large machines that have sensors which need to be monitored. These sensors can report values such as pressure, temperature, valve states (open/closed) and so on. There is already a system in place that you have to interface with which collects all these values from different parts of the plant. You can access this system by using a class named PlantMonitor. For a given plantId denoted by an integer, you can use the getTemperatures, getValvestates and getPressures of all devices in the plant.



These values will be Lists of values of the appropriate datatype (double, boolean, and double, respectively).

(a) (14 points) Your job is to create an alert messaging system. This system should monitor a plant by using the PlantMonitor class above and notify other systems when:

- A temperature value is above or below two standard deviations of the 6-hour average.
- A valve state changes from open to closed.
- A pressure value is above or below one standard deviation of the 2-hour average.

When an alert is generated any number of other systems (that you do not control) should be notified. These are not systems that are part of your program, but rather screens, LEDs and sound-alarms inside a plant's control room that will use your system's classes to listen to alerts.

Because you do not know beforehand the systems that will use your monitoring application you decide to implement the system using the Observer pattern (see cheat sheet at the end of the exam). This way many different observers can listen and consume messages that are generated by your system. The observers should only be notified in case one of the alerts shown above is generated.

Draw a UML class diagram of the system using the Observer pattern. Your diagram should contain classes or interfaces that observers can inherit or implement, also include an example observer (for instance an AlarmBell class). Your diagram should also include a class that monitors and reports values that exceed a threshold. You should only include public fields and methods that are relevant to the application.

If you make any assumptions in your model, please write them down (max 3 sentence). If you feel like your model needs explanation, you may do so in addition to the assumptions in no more than 3 sentences.

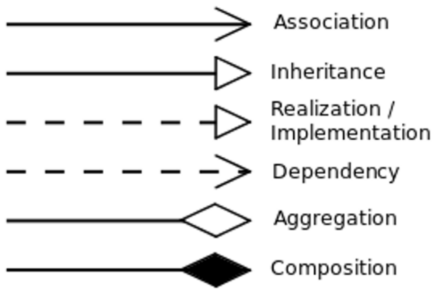
(b) (10 points) The PlantMonitor class should be accessed only once for each running instance of the software. Therefore, it should be rewritten as a Singleton. The developer has no idea how to do this, so they have come to you for assistance.

Draw a UML diagram where you change the original implementation of the PlantMonitor class into an implementation that uses the Singleton design pattern.

(no cheat sheet for this pattern is provided, remember that the singleton pattern ensures that only one instance of the class exists)

Cheat Sheet

UML



Observer Pattern