NATIONAL UNIVERSITY OF SINGAPORE

CS2103/T – SOFTWARE ENGINEERING

(Semester 1: AY2018/19)

Part 2

Time Allowed : 1 Hour

INSTRUCTIONS TO STUDENTS

- ☐ Please write your Student Number only. Do not write your name.
- ☐ This assessment paper contains **FOUR** questions and comprises **FIVE** printed pages.
- ☐ You are required to answer **ALL** questions.
- ☐ This is an **OPEN BOOK** assessment.
- ☐ You may **use pencils** to write answers.

STUDENT NO: _____

This portion is for examiner's use only

Question	Marks	Remarks
Q1	/5	
Q2	/5	
Q3	/5	
Q4	/5	
Total	/20	

Q1 [4+1=5 marks]

(a) [4 marks] Illustrate the class structure of the following code using a suitable UML diagram. In addition to the classes in the code, include the following classes:

- `ActionX`, `ActionY`: Inherits from the `Action` class.
- `ActionFactory`: creates `ActionX`, `ActionY` objects and adds them to the `History` object.

Show all attributes, methods, navigabilities, dependencies, and known multiplicities. Other optional notations can be used only when they add value.

```
/** A repeatable action to be performed by the app */
abstract class Action implements Repeatable{
    List<Task> tasks; //1 or more tasks that the action consists of
    abstract void perform();
}

interface Repeatable{
    void repeat();
}

class History{
    List<Repeatable> history;
    void add(Repeatable r){
        //..
    }
}
```

```
class Task{
    Repeatable owner; // the owner of this task
    Task previous; // the previous task, if any

    Task(Repeatable owner, Task previous){
        //..
    }
}
```

(b) [1 mark] Which *design pattern* (out of the ones covered in the module) is being used in the above design (a close match is acceptable – does not have to be an exact match)?

Answer:

Q2 [3+2=5 marks]

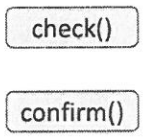
(a) [3 marks] Use a UML sequence diagram to illustrate the interactions caused by calling the `generate()` method given below. Ignore the `validate()` method.

```
class TaskList{
    Task generate(boolean hasExisting){
        if (hasExisting){
            return getTask();
        } else {
            return new Task();
        }
    }

    void validate(){
        check();
        confirm();
    }

    //...
}
```

(b) [2 marks] Assume the `validate()` method and the `generate()` method of the a `TaskList` object -- code given in part (a) above -- are run in parallel using some mechanism. Illustrate that scenario as a UML activity diagram. Use the partial diagram given below as the starting point.



Q3 [2+3 = 5 marks]

(a) [2 marks] The `isDigit(String s)` method returns `true` only if `s` is a single digit integer e.g., "4" or "9" (i.e., it should be a string of length 1). Suggest 5 more test inputs for this method where the expected result is **`false`**, in addition to the example test case given. Assume you are doing exploratory testing and give test inputs you think that are more likely to break the SUT i.e., this question evaluates your creativity as a tester, rather than your ability to apply theory.

	Input	Explanation
1	"10"	More than one digit
2		
3		
4		
5		
6		

(b) [3 marks] The `isNormal(Status s, int t)` method returns true if both the following conditions are satisfied:

- s is either OK or FINE (Status is an enumeration with values OK, FINE, BAD)
- t is in the range -5 to 99, including both values

Three test case for testing this method are given below. Give six more test cases that you will use to test this method in an efficient and effective manner.

Test cases that return **`true`**:

	s	t
1	OK	0
2		
3		
4		
5		

Test cases that return **`false`**:

	s	t
1	FINE	150
2	OK	-200
3		
4		
5		
6		
7		

Q4 [5 marks] For the following code, suggest how to fix 3 coding standard violations (w.r.t. the coding standard used in the module) and 3 other ways to improve the code quality. Use arrows and text to write your suggestions, similar to the example given.

```
// Converts the given code to a properly formatted info message
// @param code a string representing the status code
// @return null if code is an empty string.
// @throws InvalidCodeException if code is null
public String getinfo(String code) throws InvalidCodeException {

    log("In the getinfo method");

    if (code.isEmpty()) { return null; }

    if (code != null) {

        switch(code){
            case "0" :
                code = CODE_ZERO_RECEIVED;
                break;
            case "1" :
                code = CODE_ONE_RECEIVED;
                break;
            default:
                code = CODE_TWO_RECEIVED;
                break;
        }

        code = INFO_PREFIX + code + INFO_SUFFIX;
        return code;

    } else {
    // invalid code!
        throw new InvalidCodeException();
    }
}
```

Should be indented at the same level as the throws clause

--- End of assessment paper (Part 2) ---

BLANK PAGE