Mid-Term Exam (AY2022) of "Introduction to OOA OOD and UML" -- The Answer Sheet

ID: 26001904581 **Name: Tian Xiaoyang**

(1) Date/Time: 2022.05.26 10:45 ~ 12:10
(2) Please fill in your answers in this file.
(3) Please submit your answer sheet (named "Your ID number_ Mid-term_your name.pdf") by **2022.05.26 12:10** under "Mid-term Exam" in the Assignments of our course page on Manaba +R
(4) You can leave the classroom early if you finish all of the questions, even though, you have to stay in the classroom until 11:50 or later.

Q1:
Unified modeling language (UML) is a system of notations used to draw diagrams of software concepts.

Q2:
An object in the real world is an entity whose identity can be characterized by attribute, behavior, state, etc.
States/attributes are a real-world object's current characteristics.
Behaviors are its changing characteristics in time.

An object in software is a model representation of a real-world object. Its identity can also be characterized by state, attribute, behavior, etc.
States are called a software object's attributes, fields, variables.
Behaviors in software objects are called methods, operations, functions.

When modeling a real-world object in software object, only states/attributes and behaviors we are interested in will be modeled.

Q3-1:
(a)
(d)
Object0 is the source, object1 is the target

Q3-2:
(b)
(d)
Object3 is the source, object2 is the target

Q3-3:
(c)

Q3-4:
(a)

Q4-1:
Fields:
Radius: double
Color: string
“Color” is a field of “Circle” class. Its visibility is “protected”, meaning that it’s Visible within the defining class, to all classes in the same package, and to all subclasses of the defining class.
Type is “string”.
Initial value is “red”.

Q4-2:
Methods:
getRadius():double
getColor():String
setRadius(radius:double):void
setColor(color:String):void
toString():String
getArea(radius:double):double

“getRadius()” is a method. Its visibility is public, meaning that it can be accessed or redefined by objects/instances of any classes. Its return value type is “double”. It has no parameters.
“getColor()” is a methods. Its visibility is public, meaning that it can be accessed or redefined by objects/instances of any classes. Its return value type is “String”. It has no parameters.

Q4-3:
“getter” and “setter” methods are needed because accessors allow us to centralize accesses to fields, making maintenance easier.
Accessors make it easier for a compiler to optimize.

Q5:
A specific object from a specific class is called “instance”.

Q6:
(a)
(d)

Q7:
The getter methods and the setter methods are separated in class diagram. This makes the diagram visually unorganized, uneasy to read and could cause mistakes while editing the diagram.
Both “ID” and “name” fields are not initialized with any values. Initializing fields with some values can make the diagram look better. And it is a good habit while creating a class to initialize fields.

Q8:
“Point3D” class access and redefine all methods within its own class.
Including: “getZ()”, “getY()”, “Point3D(x : int, y : int, z : int)”
It can also access public methods from “Point2D” class since it inherits the class.
However, since fields “x” and “y” are private within class “Point2D”, methods “Point2D(x : int, y : int)”, “setX(x : int):void” and “setY(y : int):void” can only be accessed by “Point2D” itself since they require private fields.
So class “Point3D” can access “getX(): int”, “getY() : int”, “toString() : String”.

Q9-1:
WaterMonster class attack method implementation

```
Public class WaterMonster extends Monster{
    Public String attack(){
            Return “Attack with water!”;
    }
}


Public class StoneMonster extends Monster{
    Public String attack(){
            Return “Attack with stone!”;
            }
    }
```

Q9-2-1:
Output for “person.teach()”
Message printout: “Person can teach”

Q9-2-2:
Output for “another_person.teach()”
“Person can teach”
“Teacher can teach on a school”

Q9-2-3:
Output for “teacher.teach()”
“Teacher can teach in a school”

Q10: