Quiz 1

Deadline	N/A
Latest Submission	Friday, 23 August 2019 at 9:44AM
Raw Mark	9.00/9.00 (100.00%)
Late Penalty	N/A
Final Mark	9.00/9.00 (100.00%)

Question 1 (1 mark)

What is the output of the code below?

```
public class Employee {
    private String name;
    private float  salary;

    public Employee(String name, float salary) {
        this.name = name;
        this.salary = salary;
    }

    public Employee changeName (Employee e, String name) {
        e.name = name;
        return e;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public static void main(String[] args) {
            Employee e = new Employee("bob", 30000);
            e.name = "Fred";
            e = e.changeName(e, "Bob");
            System.out.println(e.getName());

            Calendar hireDate = null;
            e = new Director("sam",10,hireDate);
            Manager m = new Director("sussan",10,hireDate);
            Admin a = new Admin("ad",10);
        }
}
```

(a) ○	Fred
(b) ◉	Bob
(c) ○	null
(d) ○	empty string

✔ Your response was correct.
Mark: 1.00

Question 2 (1 mark)

Suppose the following two classes are defined:

```
public abstract class Figure {...}
public class Rectangle extends Figure {...}
public abstract class 3DFigure extends Figure {...}
```

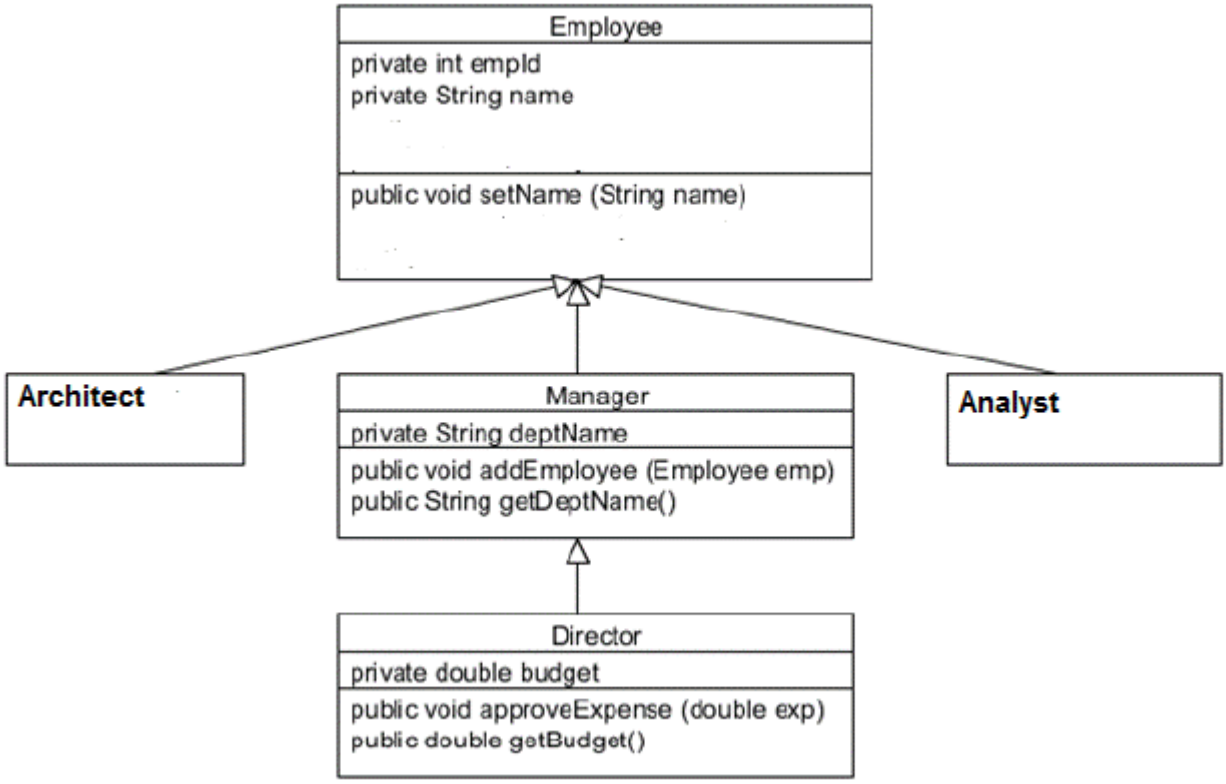
Which of the following instantiations is *not* valid?

(a) ○	Rectangle r = new Rectangle(…..);
(b) ○	Figure f = new Rectangle(….);
(c) ◉	Figure f = new 3DFigure(…);

✔ Your response was correct.
Mark: 1.00

Question 3 (1 mark)

Using the class diagram below, examine the code that follows and choose which of the following statements below does *not* compile?




```
Employee e = new Director();
Analyst a = new Analyst();
Manager m = new Director();
```

(a) ◉	e.addEmployee()
(b) ○	m.addEmployee(a);
(c) ○	((Director)m).approveExpense(10000)

✔ Your response was correct.
Mark: 1.00

Question 4 (1 mark)

The code below produces a compilation error. Examine the code and choose the fix that will enable the classes to compile

```
public class Account {
    private double balance;
    public Account (double balance) { this.balance = balance; }
    // other getter and setter for balance
}
public class Savings extends Account {
    private double interestRate;
    public Savings(double rate) {
        this.interestRate = rate;
    }
}
```

(a) ○	Call the setBalance method of the Account from Savings
(b) ○	Change the access of interestRate to public
(c) ○	Add a no-arg constructor to class Savings
(d) ◉	Replace the constructor in Savings with one that calls the constructor of Account using super.

✔ Your response was correct.
Mark: 1.00

Question 5 (1 mark)

Which of the following statements is *untrue* about an “immutable” class ?

(a) ○	All attributes must be private to prevent access from outside the class
(b) ○	Have a constructor that enables an object to be instantiated the first time with values
(c) ○	An object instance cannot be changed after it is created
(d) ◉	Provide only setter and getter methods to access the attributes from outside the class

✔ Your response was correct.
Mark: 1.00

Question 6 (1 mark)

An abstract method must *not* have:

(a) ◉	a method implementation

(b) ○	a return value
(c) ○	method parameters
(d) ○	a protected access modifier

✔ Your response was correct.
Mark: 1.00

Question 7 (1 mark)

Which of the following is *untrue* about interfaces and inheritance?

(a) ◉	A class can extend multiple interfaces
(b) ○	An interface can extend multiple interfaces
(c) ○	A class can extend another class and implement multiple interfaces
(d) ○	All methods in an interface are implicitly abstract, unless provided with a default implementation

✔ Your response was correct.
Mark: 1.00

Question 8 (1 mark)

Which of the following statements is *untrue* about method overriding?

(a) ○	Constructors cannot be overridden
(b) ○	If a static method in the base class, is redefined in the sub-class, the later hides the method in the base class
(c) ○	In method overriding, run-time polymorphism ensures that instantiated, the call to any method in the base class will be resolved to the correct method, based on the run-time type of the object instantiated.
(d) ◉	During method overriding, the overridden method in the sub-class can specify a weaker access modifier

✔ Your response was correct.
Mark: 1.00

Question 9 (1 mark)

Choose the *incorrect* statement

(a) ○	The principle of least knowledge reduces dependencies between objects and promotes loose coupling

(b) ◉	The code below is a good example of the principle of least knowledge
	`Driver driver = car.getDriver()`
	`Address driverAddress = driver.getAddress()`

(c) ○	According to the principle of least knowledge, accessing the methods on objects returned by a method call is invalid
(d) ○	The principle of least knowledge states that accessing methods of objects passed in as parameters or instantiated inside the method is valid

✔ Your response was correct.
Mark: 1.00

✔ Submit

Quiz 2

Deadline	N/A
Latest Submission	Friday, 23 August 2019 at 9:48AM
Raw Mark	7.00/7.00 (100.00%)
Late Penalty	N/A
Final Mark	7.00/7.00 (100.00%)

Question 1 (1 mark)

Which of the following statements is FALSE in relation to generics?

(a) ○	A generic class used without type arguments is known as a *raw type*
(b) ○	You cannot instantiate an array of a generic type using new operator e.g., T[] anArray = new T[100]
(c) ◉	Consider the following method, someMethod(Box<Number> n) { /*…. */}, this method can take in a Box<Integer> or Box<Double>

✔ Your response was correct.
Mark: 1.00

Question 2 (1 mark)

Consider the following code:

```
public interface xyz {
void abc() throws IOException;
}
public interface pqr {
void abc() throws FileNotFoundException;
}
public class Implementation implements xyz, pqr {
// insert code
{ /*implementation*/ }
}
```

Which of the following statement(s) can you insert in place of "// insert code" comment?

(a) ○	public void abc() throws FileNotFoundException, IOException
(b) ○	public void abc() throws IOException
(c) ◉	public void abc() throws FileNotFoundException

✔ Your response was correct.
Mark: 1.00

Question 3 (1 mark)

A design pattern used to enhance a component's functionality dynamically at run-time is:

(a) ○	Composite Pattern
(b) ◉	Decorator Pattern
(c) ○	Abstract Factory Pattern
(d) ○	Observer Pattern

✔ Your response was correct.
Mark: 1.00

Question 4 (1 mark)

- Which Design Pattern should you use when....
  - there is a language to interpret, and you can represent statements in the language as abstract syntax trees.

(a) ○	Singleton
(b) ○	State
(c) ◉	Composite
(d) ○	Factory

✔ Your response was correct.
Mark: 1.00

Question 5 (1 mark)

Identify the pattern used in this code:

```
LineNumberReader lnr = new LineNumberReader(
                       new BufferedReader(
                       new FileReader("./test.c")));
String str = null;
while((str = lnr.readLine()) != null)
System.out.println(lnr.getLineNumber() + ": " + str);
```

(a) ○	Strategy
(b) ○	State
(c) ○	Factory
(d) ◉	Decorator

✔ Your response was correct.
Mark: 1.00

Question 6 (1 mark)

Which of the following exceptions must be handled by a try-catch block or declared?

(a) ○	NullPointerException
(b) ◉	MalformedURLException
(c) ○	ClassCastException
(d) ○	ArithmeticException

✔ Your response was correct.
Mark: 1.00

Question 7 (1 mark)

Which of the following statements is NOT true?

(a) ◉	The Builder Pattern is a violation of the law of demeter
(b) ○	Decorators provide a flexible alternative to inheritance for extending functionality.
(c) ○	The observer pattern provides a design where subjects and observers are loosely coupled

(d) ○	The Factory Method Design Pattern uses inheritance to solve the problem of creating objects without specifying their exact object classes

✔ Your response was correct.
Mark: 1.00

✔ Submit

Quiz 3

Deadline	N/A
Latest Submission	Friday, 23 August 2019 at 9:51AM
Raw Mark	5.00/5.00 (100.00%)
Late Penalty	N/A
Final Mark	5.00/5.00 (100.00%)

Question 1 (1 mark)

Which one of the following is not a code smell?

(a) ◉	Classes that not only passively store data, but also methods to operate on the data
(b) ○	Large conditional logic blocks
(c) ○	Methods making extensive use of another class

✔ Your response was correct.
Mark: 1.00

Question 2 (1 mark)

A class that isn't doing enough work to justify its maintenance is an example of code smell

(a) ○	Data Class
(b) ○	Inappropriate intimacy
(c) ○	Data Clumps
(d) ◉	Lazy class

✔ Your response was correct.
Mark: 1.00

Question 3 (1 mark)

When code has sets of variables usually passed together in multiple places, this is an example of the code smell

(a) ○	Duplicated code
(b) ○	Inappropriate intimacy
(c) ◉	Data Clumps
(d) ○	Lazy class

✔ Your response was correct.
Mark: 1.00

Question 4 (1 mark)

Which of the following statements about design pattern is NOT true?

(a) ○	The collections.sort() method is a good example of the strategy Pattern
(b) ◉	The Java IO makes use of the composite pattern
(c) ○	The Java collection framework makes use of the Iterator Pattern

✔ Your response was correct.
Mark: 1.00

Question 5 (1 mark)

An online camping store, sells different kinds of camping equipment. Items selected by the customer are added to a shopping cart. When a user clicks on the checkout Button, the application is required to calculate the total amount to be paid. The calculation logic for each item type varies, and we want to move all the calculation logic to one separate class, to decouple the different items from the calculation logic applied on them. As the application iterates through the disparate set of items of the shopping cart, we apply the price computation logic in the class to each item type. Which of the following patterns would be useful to design this scenario?

(a) ○	Strategy Pattern
(b) ○	Decorator Pattern
(c) ○	Iterator Pattern
(d) ◉	Visitor Pattern

✔ Your response was correct.
Mark: 1.00

Quiz 3

Deadline	N/A
Latest Submission	Friday, 23 August 2019 at 9:51AM
Raw Mark	5.00/5.00 (100.00%)
Late Penalty	N/A
Final Mark	5.00/5.00 (100.00%)

Question 1 (1 mark)

Which one of the following is not a code smell?

(a) ◉	Classes that not only passively store data, but also methods to operate on the data
(b) ○	Large conditional logic blocks
(c) ○	Methods making extensive use of another class

✔ Your response was correct.
Mark: 1.00

Question 2 (1 mark)

A class that isn't doing enough work to justify its maintenance is an example of code smell

(a) ○	Data Class
(b) ○	Inappropriate intimacy
(c) ○	Data Clumps
(d) ◉	Lazy class

✔ Your response was correct.
Mark: 1.00

Question 3 (1 mark)

When code has sets of variables usually passed together in multiple places, this is an example of the code smell

(a) ○	Duplicated code
(b) ○	Inappropriate intimacy
(c) ◉	Data Clumps
(d) ○	Lazy class

✔ Your response was correct.
Mark: 1.00

Question 4 (1 mark)

Which of the following statements about design pattern is NOT true?

(a) ○	The collections.sort() method is a good example of the strategy Pattern
(b) ◉	The Java IO makes use of the composite pattern
(c) ○	The Java collection framework makes use of the Iterator Pattern

✔ Your response was correct.
Mark: 1.00

Question 5 (1 mark)

An online camping store, sells different kinds of camping equipment. Items selected by the customer are added to a shopping cart. When a user clicks on the checkout Button, the application is required to calculate the total amount to be paid. The calculation logic for each item type varies, and we want to move all the calculation logic to one separate class, to decouple the different items from the calculation logic applied on them. As the application iterates through the disparate set of items of the shopping cart, we apply the price computation logic in the class to each item type. Which of the following patterns would be useful to design this scenario?

(a) ○	Strategy Pattern
(b) ○	Decorator Pattern
(c) ○	Iterator Pattern
(d) ◉	Visitor Pattern

✔ Your response was correct.
Mark: 1.00