Information

Important information:

- This final exam is two hours (120 minutes) long.
- The test is worth a total of 100 marks.
- There are no optional questions.
- Unanswered questions will result in 0 marks.
- Despite penalties, the total number of marks for a question cannot be less than 0.
- Check carefully the number of marks allocated to each question. This suggests the amount of time you should spend on the question.
- Questions may not be in order of increasing difficulty.
- You might receive partial marks.
- For coding questions we include example tests. However, due to the number of tests, not all tests are shown to you.
- You receive two 10% penalties then 20% penalties on all coding questions for incorrect attempts.
- **Non-coding questions are single attempt only.** Retries will not count towards your grade, even though it will allow you to resubmit.
- You will not see your final results immediately.
- There may be additional instructions on a question so pay attention to the question description.
- This is an open book assessment item.
- **Academic integrity*** must be observed at all times.
- You are permitted to use an IDE (e.g., Eclipse) and the Java API on your own computer.
- No form of collaboration is permitted.
- We reserve the right to require an additional assessment if we suspect unethical behavior during this exam.
- There are two types of coding questions:
  - **Classes:** submit only Java class(es), no individual methods. No imports, package declarations or main methods are required.
  - **Methods:** submit only Java method(s), no surrounding class is required.
- **You may consult the Java API if you would like to use methods from the standard Java classes.** Do not supply any imports or package statements when you submit a question (even if you need them to work in Eclipse). You have access to all classes in java.lang and java.util (we supply these for you). If your solution uses classes not included in these packages, you will need to refer to them by their full name (e.g., java.text.Format), rather than importing java.text and referring to the type as just Format.

*** Academic integrity** is a principle at UC whereby both staff and students agree to act honesty, fairly, ethically and with respect for each other in teaching and learning.

For some students, there may be increased temptation to cheat and engage in dishonest academic practices such as:

- **Plagiarism/Self-plagiarism –** using someone's ideas and information without acknowledging them as the source or self-plagiarism where someone attempts to submit their own writing to two different assessments to gain credit twice;
- **Collusion –** copying the work of someone else or allowing someone else to copy your work without disclosing this with the intent to deceive;
- **Impersonating/Ghost writing –** having another person or commercial organisation impersonate you and complete an assessment item on your behalf; and/or
- **Fabrication –** 'inventing' data for example in a lab report or from a publication.

Cheating and academic dishonesty will **not** be tolerated at UC. If you are suspected of engaging in a dishonest academic practice this will result in disciplinary action being taken against you such as receiving a fine or being suspended/expelled from your studies.

DECLARATION

By starting the exam, I confirm that I have read and understood the rules of the exam and the Academic integrity principle. Also I understand that the exam is considered confidential and I must not discuss the exam materials until the exam is over for all students. I also understand that failure to comply with these requirements may mean that the matter will be referred to the Head of Department, Dean or Proctor as appropriate for disciplinary action.

Question **1**

Not yet answered

Marked out of 6.00

Given the following software requirements, indicate which of them are quality requirements (also known as "non-functional requirements") or functional requirements. Note that all requirements are independent from each other and not for one system.

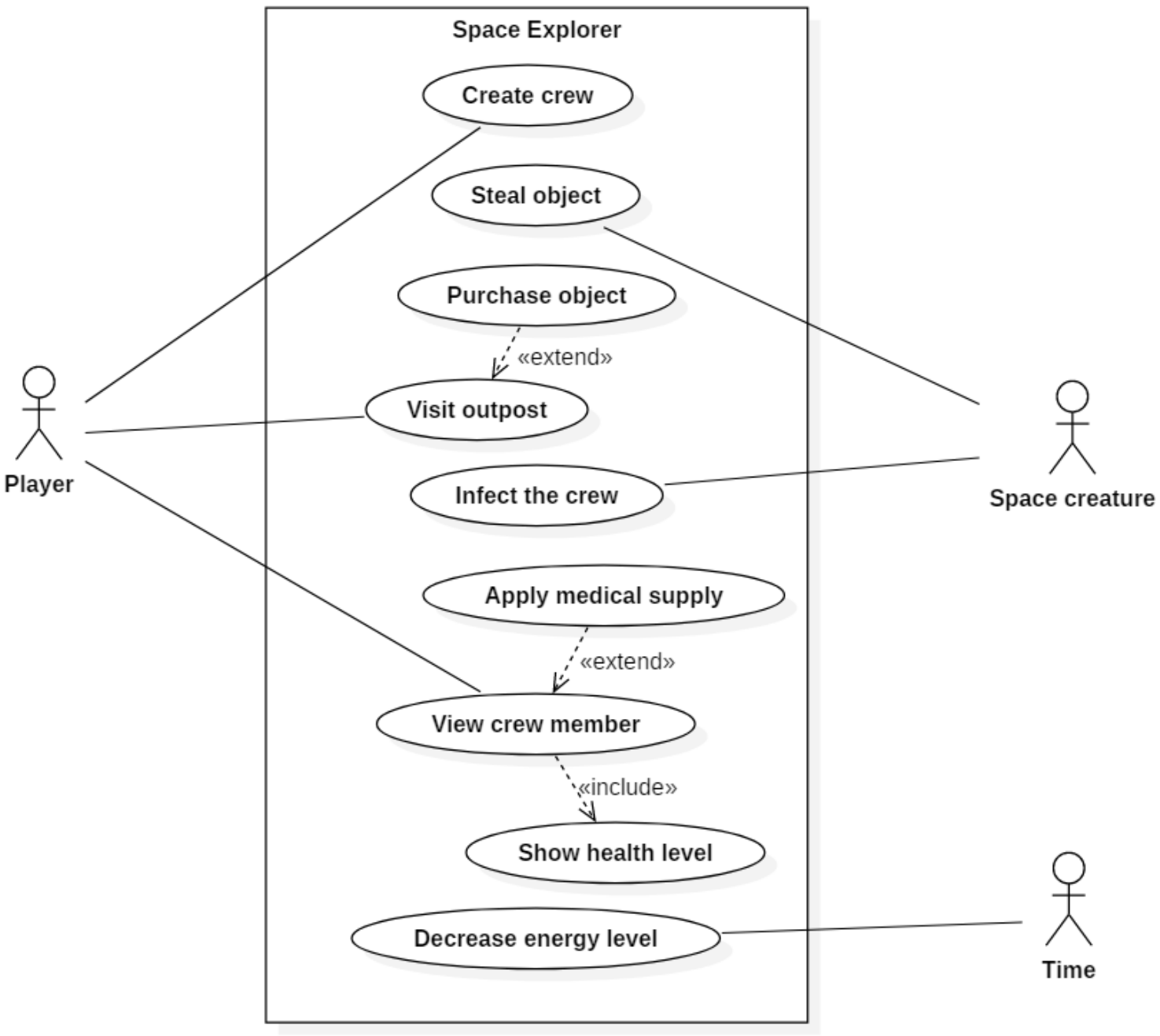
The software should be able to create new types of reports according to the Health Ministry requirements and/or the clinicians' demands.	**Functional**
The new software system to be installed in the main branch of Independence Bank, which operates 120 branches, will not be unavailable, on average, more than 10 minutes per month during the bank's office hours.	**Non-functional**
The organ donors' registration system should run on Windows 7, 10 and Linux operating systems.	**Non-functional**
The programming will adhere to the company coding standards and guidelines.	**Non-functional**
A software development team has been asked to develop a set of modules for the operation and control of a hotel swimming pool for hotel guests and members of a sports club. These modules will include: - Entry checks based on membership cards. - Maintaining records of visits. - Processing of membership renewal letters.	**Functional**
The City Council is planning to allow citizens access GIS (Geographic Information System) information through the Internet. The software should allow users to view changes in the city maps. Users won't be able to view maps that show city development plans in progress.	**Functional**

Question **2**

Not yet answered

Marked out of 8.00

Given the UML diagram below, indicate which of the following statements regarding use cases are true or false, or cannot be answered since we don't have enough information. Note that some statements are generic while some refer to this specific use case diagram.



Use cases can implement other use cases.	**False**
Use cases can extend other use cases.	**True**
"Steal object" can be executed only once.	**Not enough information**
When use case "Visit outpost" is executed, use case "Purchase object" is always executed too.	**False**
"Space creature" is an external system rather than a user.	**Not enough information**
Time can't be an actor.	**False**
Inheritance is not allowed between actors.	**False**
Textual use cases would focus on capturing blue sky scenarios.	**False**

Question **3**

Not yet answered

Marked out of 9.00

Scrum is a method focused on **Project Management** activities, while XP (eXtreme Programming) is a method for **Programming** activities.

Scrum defines three specific roles for teams. These specific roles are **Product Owner, Scrum Master, Development Team Members**.

Kanban helps with **scheduling** tasks in **agile** projects.

Kanban offers a workflow displayed on a board with cards for each **backlog item**, the work in progress is **strictly limited**.

Question **4**

Not yet answered

Marked out of 6.00

Unmanaged risks become **threats** to a project. In order to handle risks, we need to **prioritize** them to decide which risks to handle when.

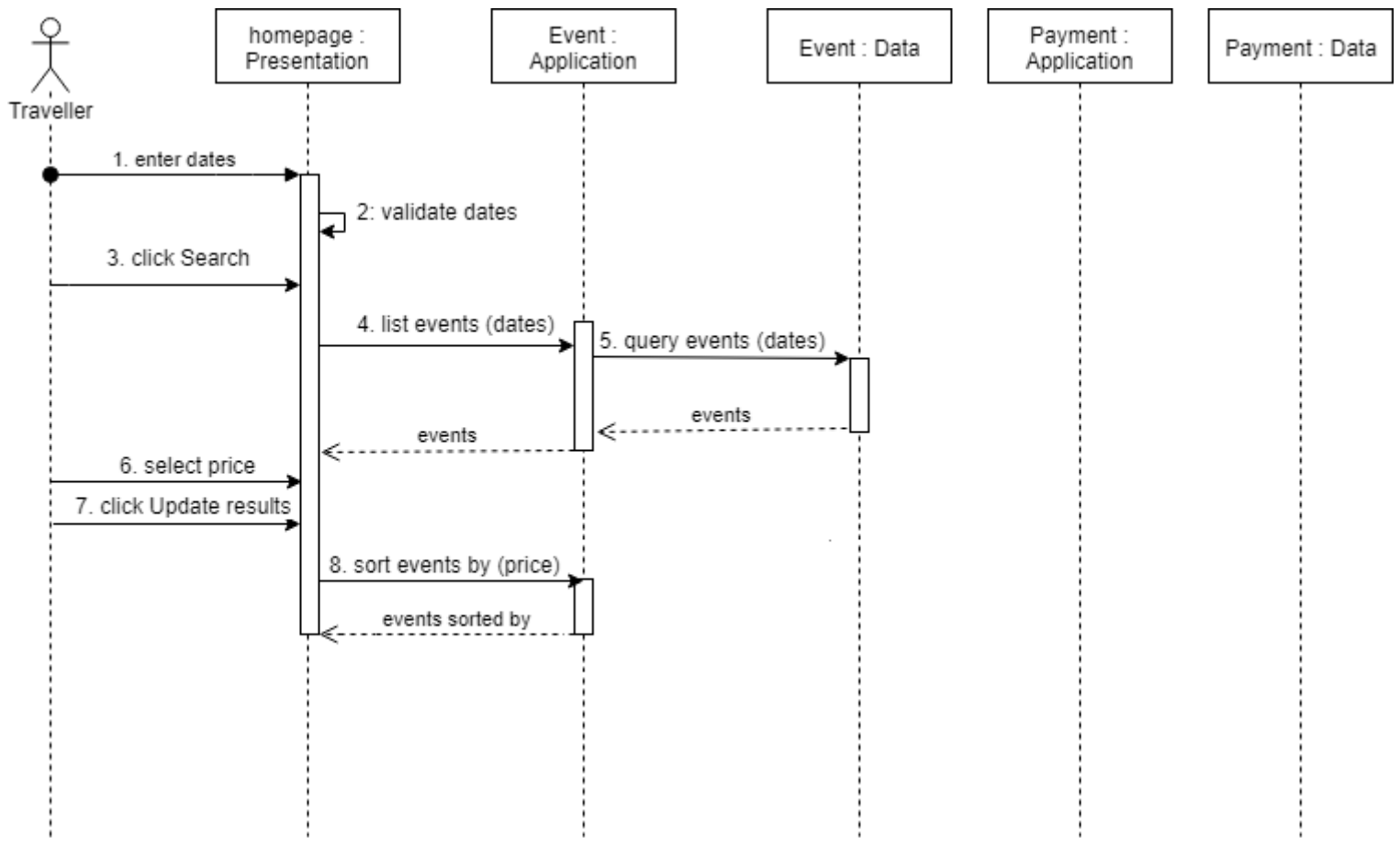
Once identified, we can think about risk **mitigation** strategies. For each risk we also need to understand how **likely** a risk becomes a threat to project success and the potential **impact** it can cause.

Question **5**

Not yet answered

Marked out of 6.00

Identify the pattern or patterns represented in the following UML Sequence diagram.



Select one or more:

- [ ] a. Client-Server
- [ ] b. Observer
- [x] c. 3-tier
- [ ] d. Adapter
- [ ] e. Façade
- [ ] f. Model-View-Controller

Question **6**

Not yet answered

Marked out of 9.00

List and explain (in bullet points) **three design decisions** that you made (or should have made) in your course project (farm simulator) and how they impacted your project, design, code, etc.
Your answer should include a discussion of good design principles, design patterns, architectural patterns or best software engineering practices.
Three marks will be awarded for listing and discussing each design decision.

Question **7**

Not yet answered

Marked out of 2.00

VCSs enable parallel development and effective management of simultaneous updates.

Select one:

- [x] True
- [ ] False

Question **8**

Not yet answered

Marked out of 2.00

VCSs allow for tracing all changes to the baseline system.

Select one:

- [x] True
- [ ] False

Question **9**

Not yet answered

Marked out of 2.00

VCS are the right place to store files containing sensitive information like system logs and passwords.

Select one:

- [ ] True
- [x] False

Question **10**

Not yet answered

Marked out of 2.00

A distributed VCS distributes sections of the source code between the developers. This means, none of the developers has a full copy of the source code.

Select one:

- [ ] True
- [x] False

Question **11**

Not yet answered

Marked out of 8.00

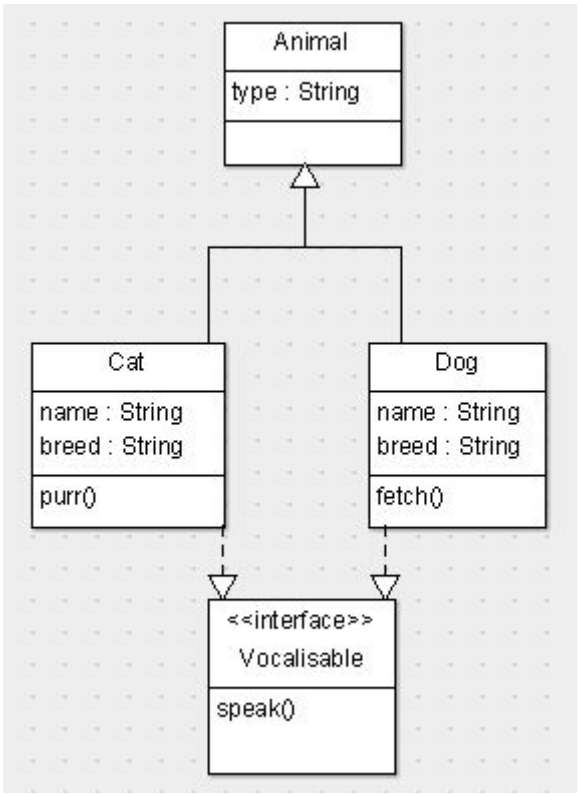
List and explain (in bullet points) the steps you followed to perform the black-box testing of your course project (farm simulator). Furthermore, discuss how effective or ineffective your approach was by discussing two disadvantages or advantages of your testing strategy.

Question **12**

Not yet answered

Marked out of 3.00

Consider the following simplified UML class diagram (we omit visibility of variables and return types of methods):



Consider the method **private void addCats()**:

```
/**
 * Creates instances of Animal, Cat and Dog
 * and an ArrayList<Cat>.
 */
private void addCats(){
        Animal myAnimal = new Animal();
        Cat myCat = new Cat("Kitty", "Persian cat");
        Dog myDog = new Dog("Woofer", "Golden Retriever");
        ArrayList<Cat> cats = new ArrayList<Cat>();
        //code selected below will go here
}
```

Based on the given simplified UML class diagram and code snippet above, select all code statements that can be added to addCats() without errors:

Select one or more:

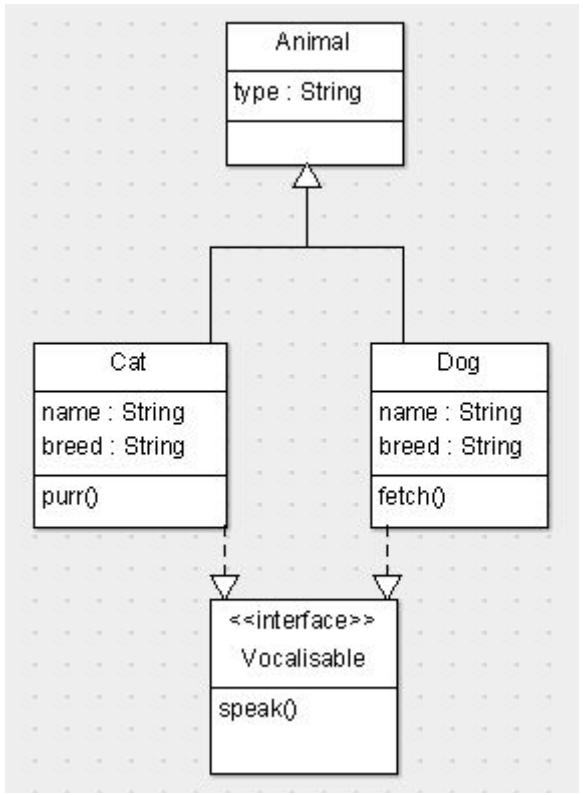
- `cats.add(myDog);`
- `cats.add((Cat) myDog);`
- `cats.add(myAnimal);`
- `cats.add((Cat) myAnimal);`
- [X] `cats.add(myCat);`

Question **13**

Not yet answered

Marked out of 3.00

Consider the following simplified UML class diagram (we omit visibility of variables and return types of methods):



Consider the method **private void addAnimals()**:

```
/**
 * Creates instances of Animal, Cat and Dog
 * and an ArrayList<Cat>.
 */
private void addAnimals(){
        Animal myAnimal = new Animal();
        Cat myCat = new Cat("Kitty", "Persian cat");
        Dog myDog = new Dog("Woofer", "Golden Retriever");
        ArrayList<Animal> animals = new ArrayList<Animal>();
        //code selected below will go here
}
```

Based on the given simplified UML class diagram and code snippet above, select all code statements that can be added to addAnimals() without errors:

Select one or more:

X
```
animals.add(myDog);
```

X
```
animals.add(myCat);
```

X
```
animals.add(myAnimal);
```

Question **14**

Not yet answered

Marked out of 2.00

What is a *collection*?

Select one:

- a. A subsystem of the Java garbage collector.
- X b. A high level construct in Java that details how data are stored.
- c. A variable that exists in all classes implicitly.

Question **15**

Not yet answered

Marked out of 4.00

Select from following **property declarations** (i.e., declarations of variables at class level) the legal statements (or those that would compile). You may assume that there are no other property declarations in the class.

Select one or more:

- [X] a. public char t = 'T';
- [ ] b. protected convenientInteger;
- [X] c. private int length;
- [X] d. public double salary$ = 106000.005;
- [ ] e. public static char true = 'T';
- [ ] f. private length = width = 4;
- [X] g. private int length, width = 4;
- [X] h. public double salary$ = 106000;
- [ ] i. public int salary = 106000.005;
- [X] j. private int length = 6, width = 4;

Question **16**

Not yet answered

Marked out of 4.00

Java and Python are distinctly different programming languages. Fill in the blanks to explain the differences.

There are two major reasons why they are different. Firstly, Java is a **compiled** language unlike Python which is **interpreted**. Secondly, Java variables are **statically** typed while Python variables are **dynamically** typed.

Question **17**

Not complete

Marked out of 4.00

Write a java ***method***, *greet* which takes in a string parameter *name*, and prints "*Hello, <name>!*".

The method signature should be ***public void greet(String name)***.

For example:

Test	Result
`greet("Mufasa");`	`Hello, Mufasa!`

Answer: (penalty regime: 0, 20, ... %)

```
1

    public void greet(String name) {
    System.out.println("Hello, " + name + "!");
    }
```

Check

Question **18**

Not complete

Marked out of 8.00

Write a java class, that contains a method that finds the time it would take for an object to fall on different planets with differing gravity.
The equation that you will need is ***time = square root(height^2/gravity)***.
In case there is no gravity, the time should be returned as infinity. You can use constant POSITIVE_INFINITY from wrapper class Double to express infinity (i.e., *Double.POSITIVE_INFINITY*). In case gravity is negative, the answer should be returned as a positive number.

The class signature should be **public class FallTime**.
The method signature should be ***public List<Double> calculateTime(double height, List<Double> gravity)***.
Remember to include all import statements you use.

For example:

Test	Result

`FallTime fallTime = new FallTime();`
`List<Double> gravity = Arrays.asList(3.70, 8.87);`
`System.out.println(fallTime.calculateTime(3, gravity));`
	`[1.559625734730109, 1.007301416809612]`

`FallTime fallTime = new FallTime();`
`List<Double> gravity = Arrays.asList(9.81, 0.00);`
`System.out.println(fallTime.calculateTime(10, gravity));`
	`[3.1927542840705043, Infinity]`

`FallTime fallTime = new FallTime();`
`List<Double> gravity = Arrays.asList(3.0, 2.5, -1.2, 0.0, -0.0, 9.92);`
`System.out.println(fallTime.calculateTime(10, gravity));`
	`[5.773502691896258, 6.324555320336759, 9.128709291752768, Infinity, Infinity, 3.1750031750047625]`

Answer: (penalty regime: 0, 20, ... %)

```
1
```

Check

Question **19**

Not complete

Marked out of 12.00

Write a java ***class*** called *Person,* with the following methods:

- *public String getName()*
- *public float getWeight()*
- *public int getAge()*
- *public String getMessage*()
- A constructor: *public Person(String name, float weight, int age)*.

The *getMessage* method should return text in the format *"A person named <name>, who is <age> years old."*

For example:

Test	Result

`Person bob = new Person("Bob", 169.5f, 20);`
`System.out.println(bob.getName());`
`System.out.println(bob.getWeight());`
`System.out.println(bob.getAge());`
	`Bob`
	`169.5`
	`20`

`Person bob = new Person("Bob", 100.0f, 20);` `System.out.println(bob.getMessage());`	`A person named Bob, who is 20 years old.`

Answer: (penalty regime: 0,20,... %)

```
1

class Person {
        private String name;
        private float weight;
        private int age;

public Person(String name, float weight, int age) {
        this.name = name;
        this.weight = weight;
        this.age = age;
}

public String getName() { return name; }
public float getWeight() { return weight; }
public int getAge() { return age; }
public String getMessage() {
return String.format("A person named %1$s, who is %2$d years old.", name, age);
        }
}
```

Check

Question **20**

Not yet answered

Not graded

I declare that this exam represents my own work in accordance with university regulations. I also understand that the exam is considered confidential and I must not discuss the exam materials until the exam is over for all students. I also understand that failure to comply with these requirements may mean that the matter will be referred to the Head of Department, Dean or Proctor as appropriate for disciplinary action.

Select one:

- (X) True
- ( ) False

◄ Lab Quiz S1-Y2016 (hidden) Jump to... Lab Test SENG201-2020 (hidden) ►