THE UNIVERSITY OF SYDNEY

INFO1113 Week 12 Tutorial

Revision

Revision

Tell us what you think

Please spend some time to complete the INFO1113 Unit of Study Survey (USS) survey found here: Unit of Study Survey Any feedback you can give on the course is helpful.

Please be specific about anything you think was good or bad about the course, and if you want to mention your tutor, remember to mention them by name. Your responses will be anonymous, and any feedback you can give (positive or negative) will help us in making the course better for future students.

Question 1: Assessed Task - Assignment 2

Remember you are required to complete Assignment 2 within the due date. Go to EdStem for this unit and click on Assessment to find out the task and the due date. This is a marked task.

Question 2: Guessing game

Write a simple guessing game where number to be guessed is provided as a command line argument to the program. It will request input from the user until the user guesses the correct number.

```
java GuessingGame 42
Guess what number it is: 21
Incorrect guess, try again: 67
Incorrect guess, try again: 42
Correct!
<Program Terminates>
```

Question 3: Monster

Consider the following interface:

```java
public interface Food {
    // returns whether or not the food is tasty
    public boolean isTasty ();
    // weight of the food in kilograms
    public int getWeight ();
    // when a food is consumed , its weight becomes zero
    // and it is no longer tasty .
    public void eat ();
}
```

The following class represents your pet monster. The monster needs to be fed regularly, otherwise it will get angry and eat you.

```java
public class Monster {
    private int weight; // weight in kilograms
    private boolean hungry;
    private boolean angry;
    // implement the constructor

    public Monster ( int weight ) { }
    // implement this method

    public void feed ( Food food ) { }

    public void pet () {
        if ( hungry ) {
            if ( angry )
            System.out.println ("Monster has eaten you");
            else {
                System.out.println("Monster has bitten you");
            }
        else {
            if ( angry )
                ystem.out.println("Monster is content");
            else {
                System.out.println("Monster loves you");
            }
        }
    }
}
```

- Implement the Monster constructor. A monster will initially be not hungry or angry, and will have the given weight.
- Implement the feed method. When given an item of food, the following things happen in this order:
  - if the food is tasty, the monster is not angry
  - if the food weighs less than 10% of the weight of the monster, it is hungry
  - the food is eaten (.eat() called)
  - if the monster is hungry after feeding it becomes angry

Question 4: Convert the following code to UML

Take the current java source code and convert it to a UML diagram.

```
public class Song extends Work {
    private String title;
    private double duration;
    public Song(Author author, String title, double duration) {
        super(author);
        this.title = title;
        this.duration = duration;
    }
    public String getTitle() {
        return title;
    }
    public double getDuration() {
        return duration;
    }
}
```

```
public class Painting extends Work {
    private String title;
    private String authorComment;
    public Painting(Author author, String title, String authorComment) {
        super(author);
        this.title = title;
        this.authorComment = authorComment;
    }
    public String getTitle() {
        return title;
    }

    public String getComment() {
        return authorComment;
    }
}

public abstract class Work {
    protected Author author;
    public Work(Author author) {
        this.author = author;
    }
    public String getAuthor() {
        return author;
    }
    public void setAuthor(Author a) {
        this.author = a;
    }
}

public class Author {
    private String name;
    List<Work> works;
    public Author(String name) {
        this.author = author;
        works = new ArrayList<Work>();
    }
    public String getName() {
        return name;
    }
    public void addWork(Work w) {
        works.add(w);
        w.setAuthor(this);
    }
}
```

Use the whiteboard to draw the UML diagram. Ensure you outline the different relationship types and clearly display all properties and types.

Sample Exam Questions

You can access the sample exam on Canvas, feel free to discuss approaches and solutions to the sample exam questions.