Family Name:..... **Other Names:** .....

Student ID:..... **Signature**.....

COMP 102: Test 4

2022, 16 December ** WITH SOLUTIONS *

Instructions

- Time allowed: **45 minutes**
- Attempt **all** the questions. There are 30 marks in total.
- Write your answers in this test paper and hand in all sheets.
- If you think a question is unclear, ask for clarification.
- Brief Java documentation is provided with the test.
- This test contributes 10% of your final grade.
- You may use dictionaries and calculators.
- You may write notes and working on this paper, but make sure your answers are clear.
- You may assume all the programs import the ecs100 library and other standard libraries.

Questions	**Marks**	
1. Defining Objects	[12]	
2. Event-driven input	[12]	
3. Using ArrayLists	[6]	
	TOTAL:	

Question 1. Defining Objects **[12 marks]**

A program for a simple car driving game for children involves a top-down view of cars driving around a grid of roads. The program needs a class to describe the Car objects.

Each Car object should store information including

- the current position of the car (x and y coordinates)
- the current direction of the car ("north", "south", "east", or "west").
- the current speed of the car
- whether the car is currently working.
- the colour of the car.

When created, all Cars start at the position in the middle of the grid (GRID_SIZE/2, GRID_SIZE/2), are facing "north", have a speed of 0.0, and are working. Each car will have a different colour, and the colour is the only parameter of the constructor.
Note: the constructor should **not** draw the Car.

Cars have lots of methods. You are to define just the following four methods for the class:

- changeSpeed(double amt): if the car is working, increase the current speed by amt (or decrease if amt is negative). But ensure that the speed does not go negative and does not go above the maximum speed of 50.0.
  If the car is not working, do not change the speed.
- crash(): change the current speed to 0 and record that the car is not working.
- isWorking(): returns true if the car is currently working and false if it is not working.
- turnRight(): change the current direction of the car to the right.
  i.e. change "north" to "east", "east" to "south", "south" to "west", and "west" to "north".

Complete the definition of the Car class below and on the next page with fields, constructor and the four methods.

```
public class Car{
    public static final double GRID_SIZE = 500;
// fields
    private double x = GRID_SIZE/2;
    private double y = GRID_SIZE/2;
    private String direction = "north";
    private double speed = 0.0;
    private boolean working = true;
    private Color color;









// constructor
    public Car(Color col){
        this.color = col;
    }

                                        //Car class continued on next page
```

(Question 1 continued on next page)

(Question 1 continued)

```
//methods                                                    // Car class continued
    public void changeSpeed(double amt){
        if (this.working){
            this.speed = this.speed + amt;
            if (this.speed < 0) { this.speed = 0; }
            else if (this.speed > 50) { this.speed = 50; }
        }









    }
    public void crash(){
        this.speed = 0;
        this.working = false;




    }
    public boolean isWorking(){
        return this.working;



    }
    public void turnRight(){
        switch (this. direction ){
            case "north" -> {this.direction = "east";}
            case "east" -> {this.direction = "south";}
            case "south" -> {this.direction = "west";}
            case "west" -> {this.direction = "north";}
        }









    }
}
```

Question 2. Event-Driven Input **[12 marks]**

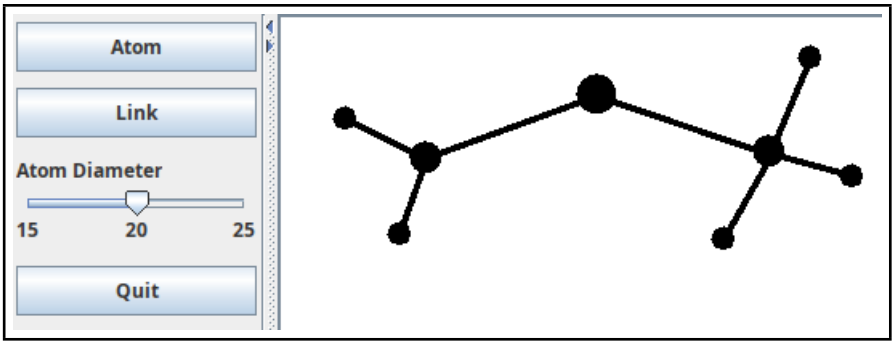
Complete the MoleculeDrawer class below and on the next page so that it allows the user to draw simple diagrams of molecules made of circles for the atoms, and lines for the links.

The program should have two buttons to say whether the user will draw atoms or links next, and a slider that will control the current atom diameter.

The user draws with the mouse. When the user releases the mouse, the program will either

- if the user is currently drawing atoms: draw a filled circle centered at the point the mouse was released. The diameter of the circle should be the current atom diameter. OR
- if the user is currently drawing links: draw a line between the mouse released position and the position where the user last pressed the mouse,

An example diagram that the user could draw is:



The program will need

- fields to remember information between method calls,
- a setupGUI method to set up the interface, and
- methods to respond to the two buttons, the slider, and the mouse.

```
public class MoleculeDrawer {
    // Fields
    private double pressedX;
    private double pressedY;
    private double diam= 20;
    private String shape = "atom";









    public static void main(String[] arguments){
        new MoleculeDrawer().setupGUI();
    }                                          //MoleculeDrawer class continued on next page
```

(Question 2 continued on next page)

(Question 2 continued)

```
    public void setupGUI(){                                    //MoleculeDrawer class continued
        UI.setMouseListener(this::doMouse);
        UI.addButton("Atom", this::setAtom);
        UI.addButton("Link", this::setLink);
        UI.addSlider("Atom Diameter", 15, 25, 20, this::setDiameter);




        UI.addButton("Quit", UI::quit);
        UI.setLineWidth(4);
    }
    public void setAtom(){
        this.shape = "atom";

    }
    public void setLink(){
        this.shape = "link";

    }
    public void setDiameter( double d ){
        this.diam = d;

    }
    public void doMouse (String action, double x, double y) {
        if (action.equals("pressed")) {
                this.pressedX = x;
                this.pressedY = y;

        }
        else if (action.equals("released")) {

                if (this.shape.equals("atom")){
                    UI. fillOval (x-this.diam/2, y-this.diam/2, this.diam, this.diam);
                }
                else {
                    UI.drawLine(this.pressedX, this.pressedY, x, y);
                }




        }
    }
}
```

Question 3. Using ArrayLists **[6 marks]**

Part of the Car game program (from question 1) records all the active Cars in the game in a field containing an ArrayList of Car objects:

```
private ArrayList<Car> activeCars = new ArrayList<Car>();
```

(a) **[3 marks]** The following keepWorkingCarsOnly() method is supposed to remove any Cars in activeCars that are **not** working.

The method has errors.

```
public void keepWorkingCarsOnly(){
    for (int i = 0; i <this.activeCars.size()-1; i++){
        if ( this.activeCars.get(i).isWorking() ){
            this.activeCars.remove(i);
        }
    }
}
```

Suppose the list contains six cars:

[*redCar*, *blueCar*, *greenCar*, *whiteCar*, *blackCar*, *pinkCar*]

and the underlined cars (*redCar*, *blueCar* and *blackCar*) are **not** working.

What will be in the activeCars list after calling keepWorkingCarsOnly?

```
[redCar, blueCar, whiteCar, blackCar, pinkCar]
```

(Question 3 continued on next page)

(Question 3 continued)

(b) **[3 marks]** Write a correct version of keepWorkingCarsOnly() that removes from activeCars all the Cars that are **not** working.

```
public void keepWorkingCarsOnly(){
    for (int i = 0; i <this.activeCars.size(); i++){
        if ( ! this.activeCars.get(i).isWorking() ){
            this.activeCars.remove(i);
            i--;
        }
    }
// OR
    for (int i = this.activeCars.size()-1; i>=0; i--){
        if ( ! this.activeCars.get(i).isWorking() ){
            this.activeCars.remove(i);
        }
    }
// OR
    ArrayList<Car> toRemove = new ArrayList<Car>();
    for (Car car : this.activeCars){
        if ( ! car.isWorking() ){
            toRemove.add(car);
        }
    }
    for (Car car : toRemove){
        this.activeCars.remove(car); }
    }
}
```

* * * * * * * * * * * * * * *

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.