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

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

COMP 103 : Test

2023, 20 January

Instructions

- Time allowed: **45 minutes**
- Attempt **all** the questions. There are 30 marks in total.
- **In-person**: Write your answers in this test paper and hand in all sheets.
  Remote: Type your answers in the template file and submit to "Test 1 Remote" on the COMP 103 submission system.
- If you think a question is unclear, ask for clarification.
- Brief Documentation of Collections and basic Java (Comp102) 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. Sets	[8]	
2. Using Collections	[14]	
3. Undo with Stacks	[8]	
	TOTAL:	

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.

Question 1. Sets **[8 marks]**

Consider the following ways of declaring and intialising a variable to hold a Set of Courses

(A) *Set*<*Course*> myCourses = **new** *Set*<*Course*>();
(B) *Set*<*Course*> myCourses = **new** *HashSet*<*Course*>();
(C) *HashSet*<*Course*> myCourses = **new** *HashSet*<*Course*>();

(a) **[3 marks]** Which of the three options will not work, and why not.

(b) **[3 marks]** Which of the remaining two options is generally the better design, and why.

(c) **[2 marks]** What property does a TreeSet provide that a HashSet does not?

Question 2. Using Collections **[14 marks]**

This question is about a program to keep track of the athletes in a sports competition.

The program has a relayTeams field containing a List of teams for the relay races. Each team is a List of Athletes, listed in the order that they will be running:

```
private List< List<Athlete> > relayTeams;
```

(a) **[4 marks]** In between races, the officials need to change the order of the athletes in each relay team. Complete the following rotateTeams() method which will move the last athlete in every team to the front position of their team.

```
public void rotateTeams(){




}
```

(b) **[5 marks]** The program also has a marathonRunners field containing a Set of Athletes running in the quarter marathon.

```
private Set<Athlete> marathonRunners;
```

Complete the following findMultiAthletes() method that will return a Set of all the Athletes who are in a relay team and are also running in the marathon.

Note: For full marks, your method should be efficient, even if there are a very large number of athletes in the competition.

Assume that the relayTeams and marathonRunners fields have been initialised and filled correctly.

```
public Set<Athlete> findMultiAthletes(){









}
```

(Question 2 continued on next page)

(Question 2 continued)

(c) **[5 marks]** The officials need to create a starting order for the quarter marathon, with the order based on age.

Complete the following startingOrder method that will return a List of the Athletes in the marathonRunners Set, sorted by age, with the oldest athletes first.

Assume the Athlete class contains a getAge method that returns the age as an int. The Athlete class is not Comparable.

Hint: note that marathonRunners is a Set, not a List.

```
public List<Athlete> startingOrder(){




}
```

Question 3. Undo with Stacks **[8 marks]**

The following BoxPlacer program lets users place boxes (black squares) on the window using the mouse. Releasing the mouse on empty space puts a new Box at the location of the mouse; releasing the mouse on top of a Box removes the Box.

There is an "Undo" button, but it doesn't do anything yet. Make the undo work so that the user can undo their recent adds and deletes of boxes. There is an Action class that you can use to record the actions in order to undo them later.

You will need to complete the undo() method, and add other bits of code to the program to make it work. (You do not need to write the Box class.)

```
public class BoxPlacer{
    private List<Box> boxes = new ArrayList<Box>();


    public void setupGUI(){
        UI.setMouseListener(this :: doMouse);
        UI.addButton("Undo", this::undo);
    }

    public void doMouse(String action, double x, double y) {
        if (action.equals("released")){
            for (int i=0; i<boxes.size(); i++){

                Box box = boxes.get(i);

                if (box.on(x, y)){

                    boxes.remove(i);

                    redisplay();
                    return;
                }
            }

            Box box = new Box(x, y);

            boxes.add(box);

            redisplay();
        }
    }

    public void redisplay(){     // redisplay all the boxes in the boxes field
        UI.clearGraphics();
        for (Box box : boxes){ box.draw(); }
    }
// code continued on next page
```

(Question 3 continued on next page)

(Question 3 continued)

```
// code continued from previous page

    public void undo(){









    }

    public static void main(String[] arguments){
        new BoxPlacer().setupGUI();
    }
}
```

```
public class Action {
    private String type;
    private Box box;
    public Action(String t, Box b){
        type = t;
        box = b;
    }
    public Box getBox(){return box;}
    public String getType(){return type;}
}
```

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

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.