Krossaspurningar

Question	Question title	Marks	Question type
ℹ	Forsíða Viðmótsforritun Hlutapróf 2025		Information or resources
ℹ	2025 Inngangur að hluta 1		Information or resources
1	Skrá fyrir notendaviðmót	5	Multiple Choice
2	Java Module System	5	Multiple Choice
3	FXML fx:id Spurning	5	Multiple Choice
4	JavaFX Stage, Scene og Node	5	Multiple Choice
5	JavaFX Layout Question	5	Multiple Choice
6	include	5	Multiple Choice
7	fx:define	5	Multiple Choice
8	createString binding og vaktbreyta	5	Multiple Choice
9	ListView toString	5	Multiple Choice
10	Dialog - fxml	5	Multiple Choice
11	Sérhæfður klasi notaður í fxml	5	Multiple Choice
12	Dialog resultConverter	5	Multiple Choice
13	JavaFX CSS og FXML	5	Multiple Choice
14	MediaPlayer MediaView	5	Multiple Choice

Opnar spurningar

Question	Question title	Marks	Question type

ℹ	2025 Inngangur að hluta 2		Information or resources

HlaupGanga

Question	Question title	Marks	Question type
15	HlaupGanga forritun	15	Programming

Binding Listener

Question	Question title	Marks	Question type
16	BindingListener	15	Programming

ℹ️

Forsíða Viðmótsforritun Hlutapróf 2025

Midterm Exam HBV201G Viðmótsforritun

Teacher: Ebba Þóra Hvannberg (fyrirlestrar)

Faculty: Industrial engineering, Mechanical Engineering and Computer Science

Course: HBV201G Viðmótsforritun

ETCS: 8

Weight of exam: 10% of final grade or more if projects have lower grades than the final exam

When: [08:20-9:45]

- **Exam rules:** The exam is taken in the Inspera testing system on students' computers that are configured to be locked with Safe Exam Browser (SEB).
- **Supporting material:** One A4 handwritten page on both siddes
- **Resources:** Pay special attention to the resources that come with the exam.
- **Structure of the exam** includes multiple choice questions. There are several answer options listed and the student chooses the "correct or most correct" answer option each time. If a student is to select more than one option it is stated. There is no deduction for incorrect answers. In fill-in-the-blank questions, it is important to enter an accurate answer because the examples are reviewed automatically.
- **Taking the exam:** It is important to read the question carefully before selecting a final answer. After answering a question, press the "Next" button or select the next question on the numbered flow line at the bottom of the screen. This action automatically saves the answer, but the student's work is also automatically saved every 15-20 seconds. It is possible to go back and forth in the test and mark questions that remain to be answered.
- **Assessment:** The exam is graded is automatically calculated based on multiple choice and fill-in-the-blank questions. Java questions in an open editor are graded according to a rubric.

Good luck

Ebba Þóra Hvannberg

ℹ # 2025 Inngangur að hluta 1

Part I. Multiple-choice questions. Total 70%. Open-ended questions in Part II count 30%.
In the multiple-choice questions, only one answer is possible. There is no deduction for an incorrect answer.

If you cannot see part of the exam, e.g. pictures or supporting documents, please contact the invigilator.

If a defect in the exam is discovered, it will be reviewed during the exam. If a student is unsure whether his understanding of a question is correct, he can indicate what that understanding is in the answer box for open-ended questions and/or send an email to the teacher after the exam.

The following supporting documents are included

1 https://prof.snara.is/
2 https://www.bing.com/translator
3 https://openjfx.io/javadoc/21/
4 https://openjfx.io/javadoc/21/javafx.graphics/javafx/scene/doc-files/cssref.html#typenumber
5 vocabulary - PDF skjal
6 Java cheatsheet
7 HlaupGanga.java - for exercise 15

1 Skrá fyrir notendaviðmót

In which file is the user interface stored?
Select one alternative:

- ○ MainApplication.java
- ○ UI.xml
- ○ Layout.class
- ○ layout-view.fxml ✓
- ○ styles.css

Maximum marks: 5

2 Java Module System

```
module hi.verkefni.vidmot{
    requires javafx.controls;
    requires javafx.fxml;
    opens hi.verkefni.vidmot to javafx.fxml;
    exports hi.verkefni.vidmot to javafx.graphics;
```

What does

requires javafx.controls;

in the module-info.java file?

Select one alternative:

- ○ It grants permission for javafx.controls to use hi.project.vidmot.
- ○ It limits the use of javafx.controls to only certain packages within the module.
- ○ It opens the package hi.project.vidmot for javafx.controls.
- ○ This indicates that this module requires javafx.controls to run. ✓
- ○ It allows other modules to use javafx.controls from this module.

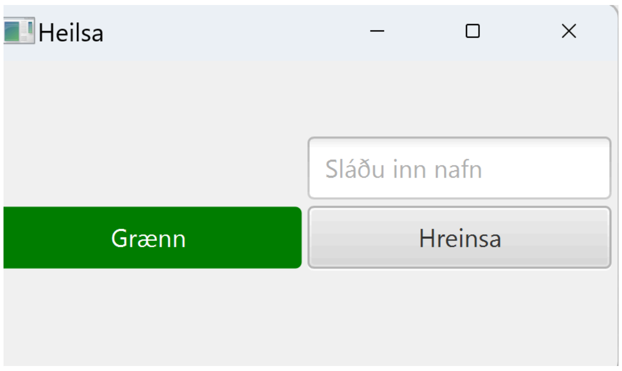
Maximum marks: 5

3 FXML fx:id Spurning

```
<Button  fx:id="jLitur" onAction="#skiptaUmLitHandler" prefHeight="29.0"
prefWidth="167.0" style="-fx-background-color: green;" text="Grænn"
textFill="WHITE" GridPane.rowIndex="2" />
<Label fx:id="jHalloNafn" prefHeight="48.0" prefWidth="154.0"
GridPane.columnIndex="1" />
<TextField fx:id="jNafn" onAction="#heilsaHandler" prefHeight="30.0"
prefWidth="156.0" promptText="Sláðu inn nafn" GridPane.columnIndex="1"
GridPane.rowIndex="1">
    <opaqueInsets>
        <Insets />
    </opaqueInsets>
</TextField>
<Button mnemonicParsing="false" onAction="#hreinsaHandler" prefHeight="30.0"
prefWidth="180.0" text="Hreinsa" GridPane.columnIndex="1"
GridPane.rowIndex="2" />
```

Above is a portion of the FXML description of the user interface for the window.

For your information, here is a running program



In the Controller class, there are instance variables for three interface objects: **jColor, jHelloName, and jName**. However, the fourth interface object (**Hreinsa** button) does not have an fx:id, and there is no instance variable for it in the Controller class. What could be the reason behind this?

Select one alternative:

- ○ The interface object is only used in the onAction handler (hreinsaHandler), so there is no need for an instance variable for it in the Controller class. ✓
- ○ It does not have an fx:id, so the interface object will not be part of the interface tree.
- ○ Instance variables are only required for interface objects placed in a VBox or GridPane.
- ○ The interface object is not visible in the application, so an instance variable is not needed for it.
- ○ FXML does not support defining buttons without fx:id.

Maximum marks: 5

4 JavaFX Stage, Scene og Node

What best describes the roles of Stage, Scene, and Node in JavaFX?

Select one alternative:

- ○ Stage is the application window, Scene contains graphical objects, and Node is a ba ✓ building block that is the root of the inheritance tree of graphical classes in JavaFX.
- ○ Stage is a graphical object, Scene is the application window, and Node is a collection of all the content in the interface.
- ○ Stage controls all interface elements, Scene creates new windows, and Node is only used for text.
- ○ Stage is the root of all interface components, Scene is used for image processing, and Node represents sound effects in the application.
- ○ Stage, Scene, and Node are only used in JavaFX to work with database connections.

Maximum marks: 5

5 JavaFX Layout Question

Which of the following is **not** an example of a layout (layout manager) in JavaFX?
Select one alternative:

- ○ VBox
- ○ GridPane
- ○ BorderPane
- ○ FlowManager ✓
- ○ StackPane

Maximum marks: 5

6 include

```
<VBox xmlns="http://javafx.com/javafx/22"
      xmlns:fx="http://javafx.com/fxml/1">
    <fx:include source="Valmyndir.fxml"/>
</VBox>
```

Look at the clip above. Which of the following statements is correct?

Select one alternative:

- ○ `<fx:include>` inserts the contents of another `.fxml` file at the same location as `<fx:include>`. ✓
- ○ `<fx:include>` is only used for CSS files and has no effect on `.fxml` files.
- ○ `<fx:include>` works like `<fx:controller>`, but allows multiple `.fxml` files to be associated with the same controller.
- ○ `<fx:include>` is unnecessary if the `.fxml` file contains `fx:root`.
- ○ `<fx:include>` automatically creates a new `Stage` when the file is loaded.

Maximum marks: 5

7 fx:define

```
<VBox xmlns="http://javafx.com/javafx/22"
      xmlns:fx="http://javafx.com/fxml/1">
    <fx:define>
        <Label fx:id="myLabel" text="Halló heimur!" />
    </fx:define>
</VBox>
```

Check outcode above

Which of the following is correct

Select one alternative:

- [✓ `<fx:define>` is used to define objects that are not automatically exposed in the JavaFX interface tree.]
- `<fx:define>` is used to set the controller for a `.fxml` file.
- `<fx:define>` automatically adds all defined objects to the JavaFX scene (`Scene`).
- `<fx:define>` is only used to define CSS styles for a `.fxml` file.
- `<fx:define>` must always be the first tag in a `.fxml` file for it to work correctly.

Maximum marks: 5

8 createString binding og vaktbreyta

```
<VBox spacing="10" xmlns="http://javafx.com/javafx/22"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="hi.properties.HelloController">
    <Label fx:id="fxTimi"/>
    <Button fx:id="fxHnappur" text="Uppfæra tíma" onAction="#uppfaeraTima"/>
</VBox>

@FXML private Label fxTimi;
@FXML private Button fxHnappur;
private final ObjectProperty<LocalTime> valinnTimi =
          new SimpleObjectProperty<>(LocalTime.now());
private final DateTimeFormatter formatter =
           DateTimeFormatter.ofPattern("HH:mm:ss"):
@FXML
public void initialize() {
    fxTimi.textProperty().bind(Bindings.createStringBinding(
           () -> valinnTimi.get().format(formatter),
           valinnTimi
    ));
}
@FXML
private void uppfaeraTima(ActionEvent actionEvent) {
    valinnTimi.set(LocalTime.now());
}
```

Above is an FXML file and Java code that shows how binding and lambda functions are used in JavaFX. What **observable** is used in the binding in fxTimi.textProperty().bind(...)?

Select one alternative:

- ○ valinnTimi ✓
- ○ formatter
- ○ valinnTimi.get().format(formatter)
- ○ fxHnappur.setOnAction(e -> valinnTimi.set(LocalTime.now()))
- ○ fxTimi.textProperty

Maximum marks: 5

9 ListView toString

```
@FXML
private ListView<Vedurstod> vedurstodvarView;
```

Review the above definition:

For the name of the weather station to appear in the list in the interface, the following would need to be programmed:

Select one of the following

- ○ Call vedurstodvarView.toString() every time the list contents are updated
- ○ Nothing additional
- ○ Implement the method getVedurstod() in Vedurstod
- ○ [✓ Implement the method @Override public String toString() { return getName()} in the class Vedurstod]
- ○ Call vedurstodvarView.toString() in initialize() in the controller

Maximum marks: 5

10 Dialog - fxml

A programmer designs a user interface for a dialog box that contains information from the **Vidskiptavinur** processing (backend) class. The programmer describes the dialog in an .fxml file. Which class is at the root of the .fxml file?

Select one of the following

- ○ Scene
- ○ DialogPane
- ○ It inherits from the Vidskiptavinur
- ○ [✓ DialogPane or a class that inherits from DialogPane]
- ○ Dialog<Vidskiptavinur> containing a DialogPane

Maximum marks: 5

11 Sérhæfður klasi notaður í fxml

When placing an object of a specialized class (Custom component) **A** in an .fxml file, it is done as follows:

Select one alternative:

- ○ <fx:component >
- ○ <fx:include class=A >
- ○ <A /> ✓
- ○ <fx:object A >
- ○ <fx:root type=A >

Maximum marks: 5

12 Dialog resultConverter

If Dialog **d** is to return data from a dialog for data from a game where the class **Game** contains the data, then a statement that opens a modal window would be like this:

Select one alternative:

- ○ Game game = d.getResult();
- ○ Dialog<Game> d = new GameDialog(); d.showModal();
- ○ d.open(Game.class);
- ○ Game game = d.show();
- ○ Optional<Game> result = d.showAndWait(); ✓

Maximum marks: 5

13 JavaFX CSS og FXML

```
protected void onRautt() {
    fxTexti.getStyleClass().removeAll("texti-green");
    fxTexti.getStyleClass().add("texti-red");
}
```

Consider the above handler in the Controller class:

What is text-green and where is it defined?

Select one alternative:

- ○ [✓ text-green is a CSS class defined in a .css file and linked to the JavaFX application using the stylesheets attribute in FXML.]
- ○ text-green is defined in the FXML file and controls the appearance of the text there.
- ○ text-green is a method in the Controller class that changes the color of the text to green.
- ○ text-green is a setting in a .css file that is set directly to the style attribute in FXML.
- ○ text-green is a constant in the Controller class that stores a string with the color code for green text.

Maximum marks: 5

14 MediaPlayer MediaView

```
private void newMediaPlayer(String mediaURL) {
    mediaPlayer = new MediaPlayer(new Media(mediaURL));
    mediaPlayer.setAutoPlay(true);
    fxMediaView.setMediaPlayer(mediaPlayer);
}
```

Look at the above code snippet and answer
What does the **setMediaPlayer(MediaPlayer mediaPlayer)** method do in the MediaView class in JavaFX?

Select one alternative:

- ○ It connects the MediaPlayer to the Slider to control where in the video it plays.
- ○ It binds a MediaPlayer to a MediaView to display video or audio. ✓
- ○ It changes the volume in MediaPlayer according to MediaView.
- ○ It automatically starts playing the video when the MediaView is shown.
- ○ It allows the user to select a file from the computer to play in MediaPlayer.

Maximum marks: 5

i 2025 Inngangur að hluta 2

Part II
This part contains 2 questions worth a total of 30%.

A Java editor is provided for programming in. The editor has statement numbers and displays Java highlights in color. The editor helps you with correct line indentation if you use curly braces. The editor does not check whether the program is grammatically correct and it does not translate or run the program.

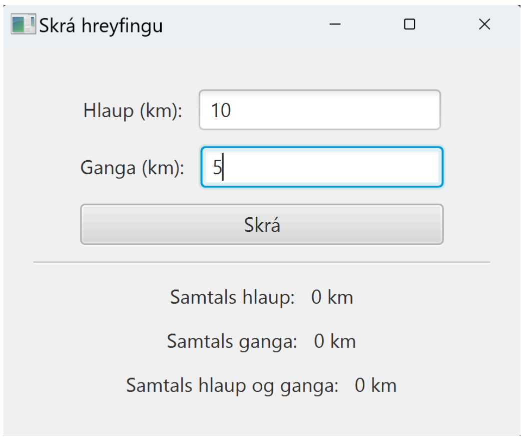
Use best practices for programming. Import statements are not required
----
If you cannot see part of the exam, e.g. images or supporting documents, please contact the proctor.
If a defect in the exam is discovered, it will be reviewed during the exam review. If a student is unsure whether his understanding of a question is correct, he can indicate what that understanding is in the answer box for open-ended questions and/or send an email to the teacher after the exam.

You are to program an application for recording exercise, running, and walking. The user enters the number of km run and the number of km walk into the text fields (**TextField**). The user does not enter <enter> at the end of the text fields. When the user presses the button **Skrá**, the text field displays the total number of km run, the total number of km walked since the program was started, and the total number of running and walking. Use the processing class (back end) **HlaupGanga.java**, which is given in resources #7.

The following table shows the status in the user interface after two records.

Skráning	**Hlaup**	**Ganga**	**Samtals hlaup**	**Samtals ganga**	**Samtals hlaup og ganga**
1	10	5	10	5	15
2	1	2	11	7	18

The following image shows the user interface after the application is launched and running and walking have been entered (Skráning 1 above) but **Skrá** has not been pressed. When you press Skrá, Samtals hlaup is 10 and Samtals ganga is 5.


Skrá hreyfingu

Hlaup (km): 10

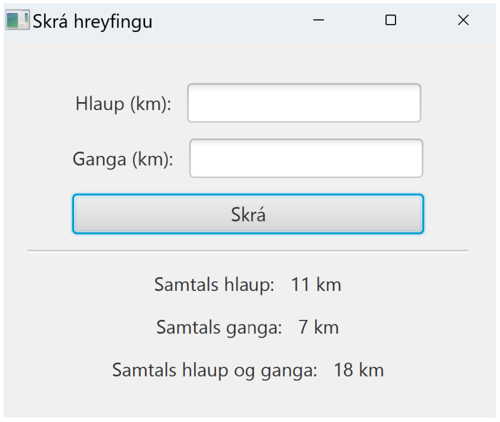
Ganga (km): 5

Samtals hlaup: 0 km

Samtals ganga: 0 km

Samtals hlaup og ganga: 0 km

The following image shows the user interface after recording another movement (Recording 2 above) and pressing **Skrá** movement.


Skrá hreyfingu

Hlaup (km):

Ganga (km):

Samtals hlaup: 11 km

Samtals ganga: 7 km

Samtals hlaup og ganga: 18 km

15 HlaupGanga forritun

```
public class HlaupGangaController {
    // tilviksbreytur
    public TextField fxHlaup;
    public Label fxHlaupSamtals;

    // forritið hér viðbótarbreytur fyrir viðmótshluti - gefið ykkur
breytuheitin

    // vinnsla - forritið

    // aðferðir
    /**
     * Aðferð fyrir Skrá hreyfingu hnappinn
     * @param actionEvent ónotað
     */
    public void skraHreyfingu(ActionEvent actionEvent) {
       // forritið
    }
}
```

Hér er beinagrind af HlaupGangaController.java - afritið í ritilinn

Fill in your answer here

Maximum marks: 15

```
<TextField fx:id="efri" promptText="Sláðu inn texta hér"/>
<Label fx:id="label"/>
<TextField fx:id="nedri" promptText="Sláðu inn texta hér"/>
<Label fx:id="uppercaseLabel" />
```

An application has two TextFields (**efri**, **nedri**) and one Label object (**label**). The application has the following functionality:

1. The **label** object should have the same value as the upper (**efri)** one. Implement with **bind**
2. When **nedri** object changes, change the new value (nyja) to uppercase and put it in **uppercaseLabel**
and print to console **breyttist** + the new value. Implement with addListener

Hint: SimpleStringProperty implements the Property interface which has the **bind** method where T is String

```
 void bind (ObservableValue<? extends T> observable)
```

and could start like this if x is an object with textProperty

```
x.textProperty().bind();
```

Hint: The String class has an instance method **toUpperCase** to convert a string to uppercase

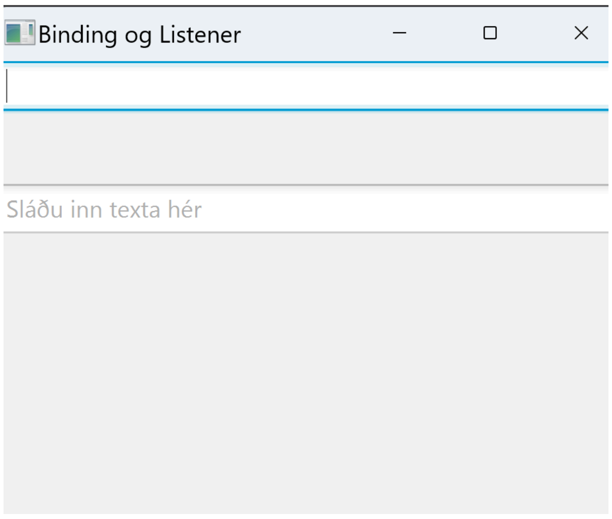
```
public String toUpperCase ()
```

Hint: You can use the following skeleton. **addListener** is a method of the ObservableValue interface that SimpleStringProperty implements, but textProperty() returns it. Types of parameters to the addListener **lambda function** are **ObservableValue<extends String>, String, String**

If x is a TextField object then you could start with and fill in.

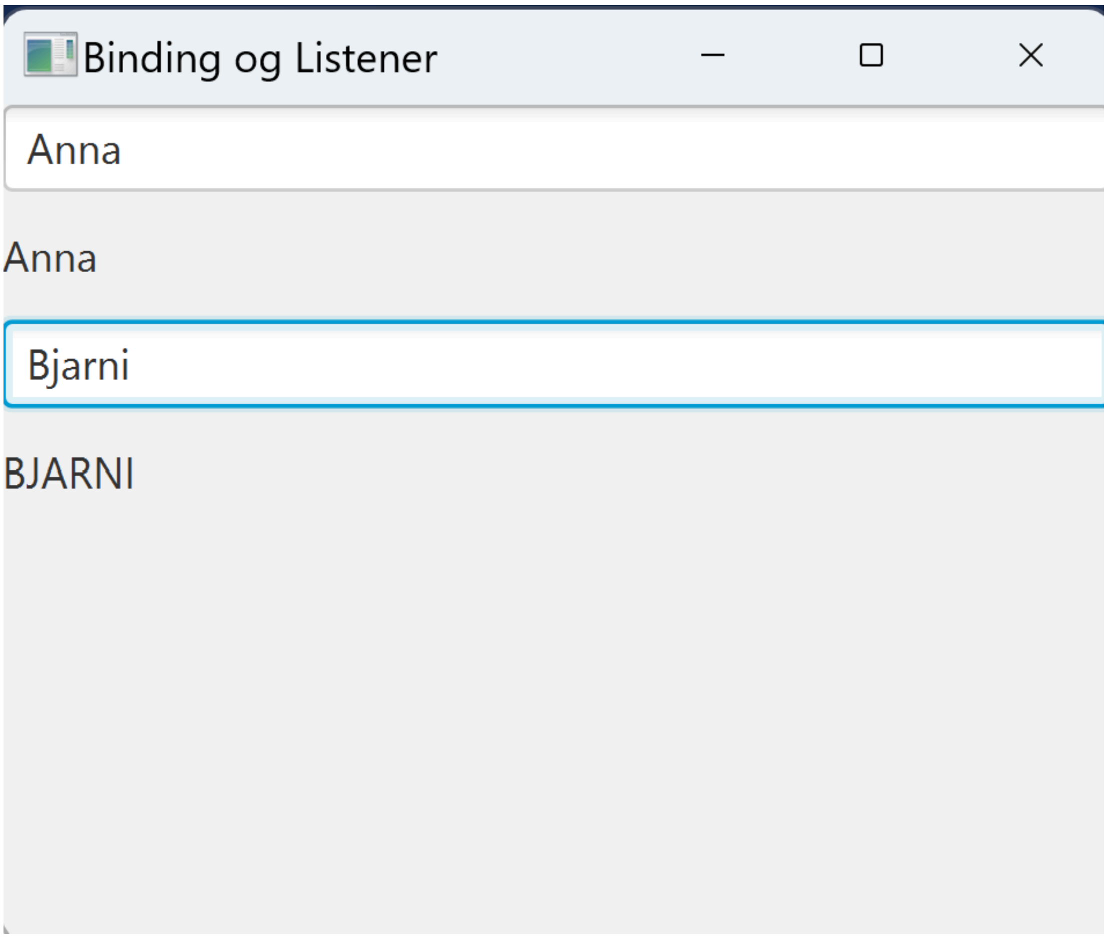
```
x.textProperty().addListener ((vaktad, gamla, nyja) -> {});
```

When the program is launched, the interface looks like this:



Þegar slegið hefur verið inn Anna í efri og Bjarni í neðri lítur viðmótið svona út og eftirfarandi er prentað á console

```
nedri breyttist: B
nedri breyttist: Bj
nedri breyttist: Bja
nedri breyttist: Bjar
nedri breyttist: Bjarn
nedri breyttist: Bjarni
```



Gefin er beinagrind að controller fyrir viðmótið. Ljúkið við að forrita.

16 BindingListener

```
public class BindingListenerController
{
    @FXML
    private TextField efri;
    @FXML
    private TextField nedri;
    @FXML
    private Label label;
    @FXML
    private Label uppercaseLabel;
    @FXML
    public void initialize() {
        // Binding: label hefur sama texta og efri
        // forritið hér - breytið
        x.textProperty().bind();

        // Þegar nedri hluturinn breytist þá er nýi textinn settur í hástafi
og settur í uppercaseLabel
        // og prentað út á console "breyttist" + nýja gildið
        // forritið hér - breytið
        x.textProperty().addListener((vaktad, gamla, nyja) -> { });
    }
}
```

Above is a skeleton of a controller. Copy it to the editor.

Fill in your answer here

```
1
```

Maximum marks: 15