COSC 3P91

Lab 9

In this and the following lab you are going to implement parts of a game called **RoboRace**. In this game players have to maneuver a robot by producing small programs for the robot. The individual instructions of the programs are represented by program cards. There are 5 different cards: Move forward, Move backwards, Turn 90 degrees clockwise, Turn 90 degrees counterclockwise, and Turn 180 degrees. In each round of the game the players get 7 cards dealt. From these cards the player selects 5 in the order they are supposed to be executed. After confirming the choice, the robot will execute the program. The images needed for this lab can be found on the course's webpage. In Lab 9 you are going to implement the card selection process. For this, do the following:

1. Implement an enumeration type `CardType` with 5 elements representing the type of a programming card.
2. Implement a class `Card` for programming cards. This class should have a static array of images for display. In the static initialization section load the 5 images Move.png, Back.png, Turn90Clockwise.png, Turn90CounterClockwise.png, Turn180.png. A card itself does have a type (`CardType`), an `getImage()` method returning the image of the card for display, and an appropriate implementation of `toString()`.
3. Implement a class `CardPanel` that is a `JPanel` and implements the `ActionListener` and `MouseListener` interfaces. Add the following constants to the class for convenience:

```
private static final int CARD_WIDTH = 92;
private static final int CARD_HEIGHT = 128;
private static final int MARKER_WIDTH = 32;
private static final int MARKER_HEIGHT = 32;
```

   In addition, implement the following:
   a. Like the class `Card` this class should have a static array of images to indicate that a card has been selected. In the static initialization section load the 5 images Select1.png – Select5.png. In addition, the class must store the image NoCard.png in a static variable.
   b. Add the following variables to the class:

```
private final JButton stopButton;
private Card[] cards;
private final List<Integer> selected;
private boolean selecting;
```

The constructor of `CardPanel` takes the `stopButton` as a parameter. In the constructor you initialize the variables and register the `CardPanel` as its own `MouseListener` and as the `ActionListener` of the `stopButton`. In addition, you set the size of the panel so that it can display exactly 7 cards, i.e., use `7*CARD_WIDTH` as the width and `CARD_HEIGHT` as the height of the panel.

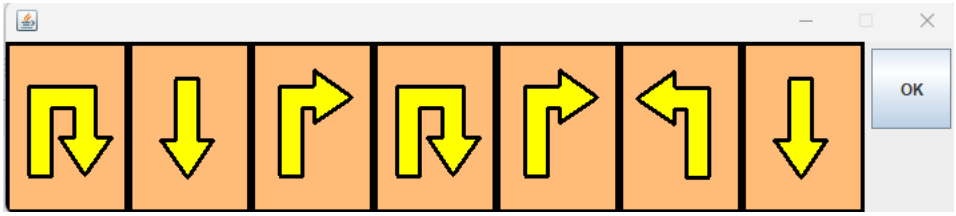
c. Implement the method `paintComponent(Graphics graphics)`. Here you display the 7 cards stored in the variable `cards`. If a card is selected (it is in the list `selected`), then place an Select*i*.png image where *i* is when the card was selected, in the center on top of the card. The order in selected provides information about when the card was selected, i.e., the first card in selected was selected first, the second card in selected was selected second, etc. Should cards be `null`, then you display 7 times the NoCard.png image.

d. In the `actionPerformed(ActionEvent e)` method you check whether there are currently 5 cards selected. If so, you stop the selection process (`selecting = false`), disable the `stopButton`, and notify other threads about this event.

e. In the method `mousePressed(MouseEvent e)` you select or deselect the card on which the mouse event happened, depending on whether the card was already selected or not. This should only be done if `selecting == true`.

f. Implement a method `Card[] selectCards(Card[] cards)`. In this method you enable the `stopButton`, store the `cards` in `this.cards`, clear any previous selections, and set `selecting` to `true`. Then the thread has to wait until the selection process is over. After that you return an array of the cards in the order they have been selected (these are 5 cards!).

4. Implement a main program that does the following:
   a. Create a `stopButton` (`JButton`) of size (60,60).
   b. Create a `cardPanel` (class `CardPanel`) using the button from a.
   c. Create and display a `JFrame` that contains the components above in the layout below. Use appropriate layout managers and containers to achieve this.



Set `JFrame.EXIT_ON_CLOSE` as the default closing operation the `JFrame`.

d. Finally, implement an infinite loop that creates 7 cards randomly, calls `selectCards` of the `cardPanel`, and then prints the 5 selected cards.