HOMEWORK 5 - SPRING 2020

HOMEWORK 5 - due Tuesday, April 14th no later than 6:00PM

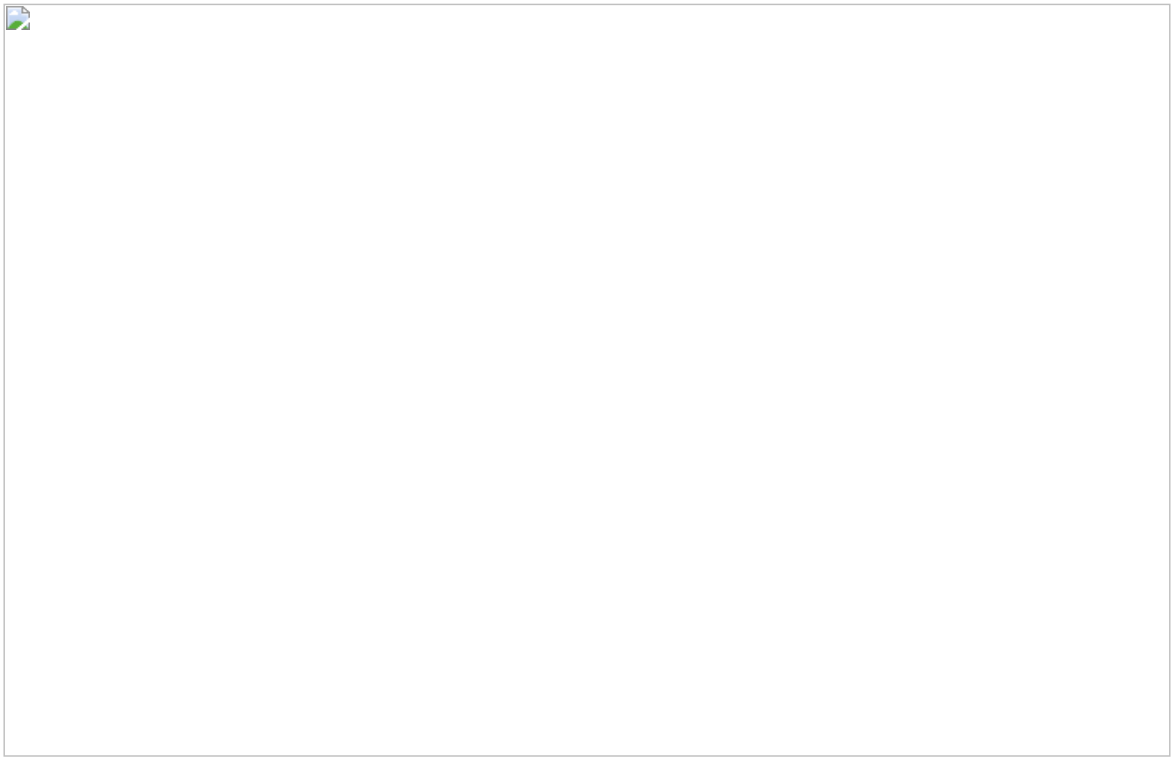
REMINDERS:

- **Be sure your code follows the coding style for CSE214.**
- **Make sure you read the warnings about academic dishonesty. *Remember, all work you submit for homework or exams MUST be your own work.***
- **Login to your grading account and click "Submit Assignment" to upload and submit your assignment.**
- **You are not allowed to use ArrayList, LinkedList, Vector or any other Java API Data Structure classes to implement this assignment.**
- **You may use Scanner, InputStreamReader, or any other class that you wish for keyboard input.**

In this assignment, you will be implementing a Choose-Your-Own Adventure type game. Using a tree structure, you can link together various scenes to tell a story. The program should be able to allow a user to play the game, as well.

UML

The UML Diagram for all the classes as specified below is as follows:



SPECIFICATIONS

1. SceneNode

This class represents a specific scene. A scene has a `title` (String), a `sceneDescription` (String), a `sceneID` (int), and up to three possible child `SceneNode` references (called `left`, `middle`, and `right`). In addition, this class should have the following methods:

- **`public SceneNode()`**
  - **Constructor for `SceneNode`. You may include additional constructors with parameters.**
- **`public void addSceneNode(SceneNode scene) throws FullSceneException`**

    - Sets the scene to the first available slot in the child nodes. Children should be added in the left-most node.
    - Throws:
        - `FullSceneExpection` if the current node does not have any empty child nodes.
- `public boolean isEnding()`
    - Determines if this scene is an ending for a storyline. If this node has no children (a leaf), then the user is deemed to have reached the end of that particular story path.
    - Returns:
        - `true` if this scene is the ending of a storyline, `false` otherwise.
- `public void displayScene()`
    - Outputs the scene information, as would be shown during gameplay (option G).
- `public void displayFullScene()`
    - Outputs all information about a scene, as would be shown in creation mode (option S)
- `public String toString()`
    - Returns:
        - A String representing a brief summary of this node, as would be shown in the tree (option P).
- Getters/Setters for all fields as necessary.

In addition, the class should keep a `static numScenes` which should keep a count of how many scenes have been created so far, and can then be used to generate Scene ID numbers from.

2. SceneTree

This class represents a collection of `SceneNode` objects arranged as a ternary tree. It has references to the `root` of the tree, as well as a `cursor` which can be used to navigate the tree (NOTE: The root of this tree should never be removed!). This class also contains methods which can be used to add and remove `SceneNodes`, move the `cursor`, update `SceneNodes`, and print them. This class should have the following methods:

- `public SceneTree()`
    - Constructor for `SceneTree`. You may include additional constructors with parameters.
- `public void moveCursorBackwards() throws NoSuchNodeException`
    - Moves the `cursor` to the parent node.
    - Throws:
        - `NoSuchNodeException` if the current node does not have a parent.
- `public void moveCursorForward(String option) throws NoSuchNodeException`
    - Move the `cursor` to the appropriate child node.
    - Parameters:
        - `option` - A String ("A", "B", or "C") dictating whether we should select the `left`, `middle`, or `right` child, resepectively.
    - Throws:
        - `NoSuchNodeException` if the current node does not have any such child.
- `public void addNewNode(String title, String sceneDescription) throws FullSceneException`
    - Creates a new `SceneNode` object as a child of the current node with the supplied values. It will be set in the first empty `left`, `middle`, or `right` node.
    - Throws:
        - `FullSceneException` if the current node does not have any available child positions.
- `public void removeScene(String option) throws NoSuchNodeException`
    - Removes the specified child node from the tree.
    - Params:
        - `option` - A String ("A", "B", or "C") dictating whether we should remove the `left`, `middle`, or `right` child, resepectively.
    - Throws:
        - `NoSuchNodeException` if the current node does not have any such child.
- `public void moveScene(int sceneIDToMoveTo) throws NoSuchNodeException, FullSceneException`
    - Moves the node at `cursor` to be a child of the node with the specified ID.
    - Params:
        - `sceneIDToMoveTo` - An integer which specifies which node this `SceneNode` object should be moved under.
    - Throws:
        - `NoSuchNodeException` if there does not exist a `SceneNode` with the supplied Scene ID.

        - `FullSceneException` if the `SceneNode` specified by the supplied Scene ID does not have any available child positions.
- `public String getPathFromRoot()`
    - Constructs a String showing the path from the root of the tree to the currently selected `SceneNode`, illustrating the decisions which would have to be made to arrive at the selected node (option N).
    - Returns:
        A String representing the path between the root of the tree and the currently selected `SceneNode`.
- `public String toString()`
    - Constructs a String representation of the tree, as shown in the Sample I/O below.
    - Returns:
        A String representation of the tree.
- Getters/Setters for all fields as necessary.

3. AdventureDesigner

This class will guide the user through making and possibly playing the game. Upon starting the program, the user should immediately be prompted to create the information of the first scene, which will be stored in the root of the tree (see Sample I/O). This class should also act as the main driver for the program, containing the `main` method and allowing the user to select from the following menu options:

- `A) Add Scene` - Prompt the user for
- `R) Remove Scene`
- `S) Show Current Scene`
- `P) Print Adventure Tree`
- `B) Go Back A Scene`
- `F) Go Forward A Scene`
- `G) Play Game`
- `N) Print Path To Cursor`
- `M) Move Scene`
- `Q) Quit` - Exits the program

In addition, this class should have the following methods:

- `public static void main(String[] args)`
    The main driver for the program, which should initialize a `SceneTree` object and guide users through the menu to help them create their own adventures.
- `public static void playGame()`
    This method start at the root of the tree and displays the scene. It asks the user to select an option and prints the scene for that option. This continues until a leaf node has been displayed, at which point the adventure is concluded and the program returns to the main method. See Sample I/O for example.

4. You will need classes to handle the exceptions thrown (see class specifications above for exception classes you need).

NOTE: You may include additional methods, variables, or classes as necessary or as you find convenient.

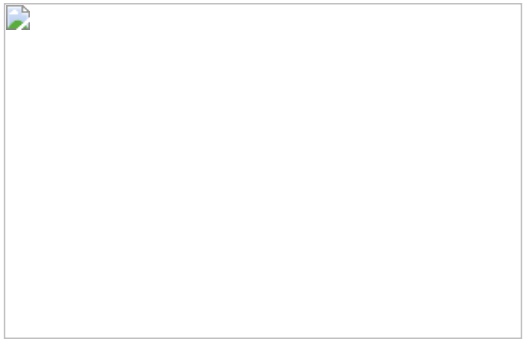
INPUT FORMAT

- Each menu operation is entered on its own line and should be case insensitive (i.e. 'q' and 'Q' are the same).
- Check to make sure that the Scene ID number or reference, if required, is valid. If not, print an error message and return to the main menu.
- For the Add commands, if the input information is valid, construct the object accordingly. Otherwise, print an error message and return to the main menu.

OUTPUT FORMAT

- Each command should output the result (as shown in the sample IO below) after each operation is performed.
- All menu operations must be accompanied by a message indicating what operation was performed and whether or not it was successful.

SAMPLE ADVENTURE TREE



SAMPLE INPUT/OUTPUT

```
// Comments in green, input in red, output in black.

Creating a story...

Please enter a title: Introduction
Please enter a scene: You enter a forest and see two paths. What will you do?

Scene #1 added.

A) Add Scene
R) Remove Scene
S) Show Current Scene
P) Print Adventure Tree
B) Go Back A Scene
F) Go Forward A Scene
G) Play Game
N) Print Path To Cursor
M) Move scene
Q) Quit

Please enter a selection: A

Please enter a title: First path
```

```
Please enter a scene: Taking the first path, you encounter a monster! What will you do?

Scene #2 added.

// Menu not displayed in sample i/o

Please enter a selection: F

Which option do you wish to go to: B

That option does not exist. // Only one child scene has been added so far.

// Menu not displayed in sample i/o

Please enter a selection: F

Which option do you wish to go to: A

Successfully moved to First path.

// Menu not displayed in sample i/o

Please enter a selection: S

Scene ID #2
Title: First path
Scene: Taking the first path, you encounter a monster! What will you do?
Leads to: NONE

// Menu not displayed in sample i/o

Please enter a selection: A

Please enter a title: Try to Fight
Please enter a scene: You try to fight the monster, but you don't have a sword! The monster kills you.

Scene #3 added.

// Menu not displayed in sample i/o

Please enter a selection: A

Please enter a title: Run away
Please enter a scene: You retreat and see that you are really lost! What should you do?

Scene #4 added.

// Menu not displayed in sample i/o

Please enter a selection: B

Successfully moved back to Introduction.

// Menu not displayed in sample i/o

Please enter a selection: A

Please enter a title: Second path
Please enter a scene: Taking the second path, you encounter a treasure chest! What will you do?

Scene #5 added.
```

```
// Menu not displayed in sample i/o

Please enter a selection: F

Which option do you wish to go to: B

Successfully moved to Second path.

// Menu not displayed in sample i/o

Please enter a selection: A

Please enter a title: Open the chest
Please enter a scene: You open the chest, discovering some gold! After you return back home, you donate it to the poor and become the hero of your town.

Scene #6 added.

// Menu not displayed in sample i/o

Please enter a selection: A

Please enter a title: Don't open the chest
Please enter a scene: You walk away from the chest, worrying about what's inside. But you see a sword lying in a bush and decide to take it. Suddenly a monster appears! What do you do?

Scene #7 added.

// Menu not displayed in sample i/o

Please enter a selection: F

Which option do you wish to go to: B

Successfully moved to Donâ€™t open the chest.

// Menu not displayed in sample i/o

Please enter a selection: A

Please enter a title: Fight the monster
Please enter a scene: With your sword, you manage to defeat the monster and become the hero of your town!

Scene #8 added.

// Menu not displayed in sample i/o

Please enter a selection: A

Please enter a title: Flee
Please enter a scene: You successfully ran away from the monster and make it home safely.

Scene #9 added.

// Menu not displayed in sample i/o

Please enter a selection: B

Successfully moved back to Second path.

// Menu not displayed in sample i/o

Please enter a selection: B
```

```
Successfully moved back to Introduction.

// Menu not displayed in sample i/o

Please enter a selection: A

Please enter a title: Turn around and go home
Please enter a scene: You decide you're just not cut out for this adventure stuff and go back home. Sleep is good.

Scene #10 added.

// Menu not displayed in sample i/o

Please enter a selection: A

Please enter a title: Find another path
Please enter a scene: You hate conformity! So you decide to venture through the trees.

You cannot add another scene!

// Menu not displayed in sample i/o

Please enter a selection: F

Which option do you wish to go to: A

Successfully moved to First path.

// Menu not displayed in sample i/o

Please enter a selection: F

Which option do you wish to go to: B

Successfully moved to Run away.

// Menu not displayed in sample i/o

Please enter a selection: A

Please enter a title: Look for moss
Please enter a scene: Moss always points towards civilization! Using its guidance, you make your way back home safe and sound.

Scene #11 added.

// Menu not displayed in sample i/o

Please enter a selection: A

Please enter a title: Wait for help to arrive
Please enter a scene: You wait for ages and are never found!

Scene #12 added.

// Menu not displayed in sample i/o

Please select an option: N

Introduction, First path, Run away

// Menu not displayed in sample i/o
// At this point, our tree should be representative of the sample adventure image shown above.
```

```
Please select an option: P

Introduction (#1)
    A) First path (#2)
        A) Try to Fight (#3)
        B) Run away (#4) * // The asterisk marks the currently selected scene.
            A) Look for moss (#11)
            B) Wait for help to arrive (#12)
    B) Second path (#5)
        A) Open the chest (#6)
        B) Don't open the chest (#7)
            A) Fight the monster (#8)
            B) Flee (#9)
    C) Turn around and go home (#10)

// Menu not displayed in sample i/o

Please enter a selection: S

Scene ID #4
Title: Run away
Scene: You retreat and see that you are really lost! What should you do?
Leads to: 'Look for moss' (#11), 'Wait for help to arrive' (#12)

// Menu not displayed in sample i/o

Please select an option: G

Now beginning game...

Introduction
You enter a forest and see two paths. What will you do?

A) First path
B) Second path
C) Turn around and go home

Please enter an option: B

Second path
Taking the second path, you encounter a treasure chest! What will you do?

A) Open the chest
B) Donâ€™t open the chest

Please enter an option: A

Open the chest
You open the chest, discovering some gold! After you return back home, you donate it to the poor and become the hero of your town.

The End

Returning back to creation mode...

// Menu not displayed in sample i/o

Please enter a selection: B

Successfully moved back to First path

// Menu not displayed in sample i/o
```

```
Please enter a selection: B

Successfully moved back to Introduction

// Menu not displayed in sample i/o

Please enter a selection: R

Please enter an option: A
First path removed.

// Menu not displayed in sample i/o

Please enter a selection: P

Introduction (#1) *
    A) Second path (#5)
        A) Open the chest (#6)
        B) Don't open the chest (#7)
            A) Fight the monster (#8)
            B) Flee (#9)
    B) Turn around and go home (#10)

// Menu not displayed in sample i/o

Please enter a selection: F

Which option do you wish to go to: B

Successfully moved to Turn around and go home.

// Menu not displayed in sample i/o

Please enter a selection: M

Move currrent scene to: 5 // This is the Scene ID # to make the current node a child of

Successfully moved scene.

// Menu not displayed in sample i/o

Please enter a selection: P

Introduction (#1)
    A) Second path (#5)
        A) Open the chest (#6)
        B) Don't open the chest (#7)
            A) Fight the monster (#8)
            B) Flee (#9)
        C) Turn around and go home * (#10)

// Menu not displayed in sample i/o

Please enter a selection: Q

Program terminating normally...
```
