Hex

Team Name:

Manager:

Recorder:

Presenter:

Analyst:

This is a Process Oriented Guided Inquiry Learning (POGIL) activity. You and your team will examine a working program. A series of questions will guide you through a cycle of exploration, concept invention, and application. There is strong evidence that this is more effective (and less boring) than a traditional lecture.

By the time you are done with this activity, you and your team should be able to:

- represent a graph.
- perform depth-first search.
- think more critically as a team.

Your team's recorder is responsible for writing your team's answers to the numbered questions on this form.

After you complete this activity, please fill out the short survey at

http://goo.gl/forms/HXjyuUb2ou

to improve this activity for future users.

Playing the game

Hex was invented independently in 1942 by the Danish mathematician Piet Hein and in 1947 by the American mathematician John Nash (of *A Beautiful Mind* fame). It is a two-player game. Divide your team into pairs and play by running Hex.java. After the first game, divide your team differently and play again.

1. Is everyone done playing and ready to pay attention to the team?

You may need to go back and play the game again to answer some of the questions to come, but you should do so *deliberately*, because your team's manager assigned one or more people to find something out, not merely because you got bored with the conversation or thought you could answer a question better on your own.

2. What is the black player trying to accomplish?

3. Is it necessary for each black or white colored hex to be next to an existing black or white hex?

4. What does the program do if you try to make an illegal move?

5. Is a tie possible? If so, give an example. If not, explain why not.

Stop here and wait for the other teams. If your instructor has given you a way to indicate that you have reached this point, use it now. Once all teams are ready, there will be a short discussion involving the whole class. Your team's presenter should be prepared to present any of your team's previous answers to the class. This discussion is also a good time for your team (through your presenter) to ask any questions you have. If your team is done before other teams, discuss the following open-ended question:

6. What strategy advice would you give for this game?

HexNode

Examine HexNode.java.

7. What is the type of the field `neighbors`?

8. Which two methods of the ArrayList class are called in HexNode?

9. If (in another class) you had an instance `h` of HexNode, how would you determine how many neighbors `h` had?

Stop here and wait for the other teams. If your instructor has given you a way to indicate that you have reached this point, use it now. Once all teams are ready, there will be a short discussion involving the whole class. Your team's presenter should be prepared to present any of your team's previous answers to the class. This discussion is also a good time for your team (through your presenter) to ask any questions you have. If your team is done before other teams, discuss the following open-ended question:

10. We could have used an array and a size counter directly in this class, rather than using an ArrayList. What are the advantages of each approach?

HexModel

Examine HexModel.java.

11. If `size` is 3, how many HexNodes does the constructor create?

12. How did your team reach your answer to the previous question?

13. After the constructor finishes, which HexNodes are neighbors of `grid[0][0]`?

14. What does the last for loop in `toString` accomplish?

15. What two things does `playAt` accomplish?

16. Aside from tests, where is `search` called?

17. How could the method that calls search be implemented without the special nodes `north`, `south`, `east`, and `west`? (Special values or objects like these, placed at the boundaries of data structures, are called *sentinels*.)

18. How are the sentinels similar to the other HexNodes? How are they different?

19. What are the base cases in the recursive method `search`?

20. Under what conditions does the for loop in `search` complete all of its passes rather than returning early?

21. How does the game fail if you remove the check `visited.contains(node)`? (It might not fail for several moves.)

22. Explain the purpose of the list `visited`.

23. Why does `findWinner` need to create a new ArrayList for each call to `search`? In other words, what would go wrong if `findWinner` just created one ArrayList and used it for both calls?

24. Explain, in plain English, the algorithm implemented by `search`.

Stop here and wait for the other teams. If your instructor has given you a way to indicate that you have reached this point, use it now. Once all teams are ready, there will be a short discussion involving the whole class. Your team's presenter should be prepared to present any of your team's previous answers to the class. This discussion is also a good time for your team (through your presenter) to ask any questions you have. If your team is done before other teams, discuss the following open-ended question:

25. If the HexNodes are linked to each other via their `neighbors` instance variables, why is it necessary to store them in the two-dimensional array `grid`?

Please fill out the survey at http://goo.gl/forms/HXjyuUb2ou.

This work was supported by the Google Education and Relations Fund's CS Engagement Small Grant Program grant #TFR15-00411.

The principal investigator, Peter Drake, wishes to thank the following for their useful comments: the members of the CS-POGIL project, specifically Clif Kussmaul and Helen Hu; Maggie Dreyer; and various anonymous reviewers.