**Phase F: Technical Prototype**

**Group Report**

**GROUP: W**

***Grading Rubric***

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **Possible Points** | **Comments** | **Score** |
| Prototype | 5 | Code not in proto\tech.  Late delivery | 6 |
| Report | 8 | “Java environment” isn’t specific. Netbeans?  Missing screenshots on second prototype.  Good discussion. | 10 |
| Group work: time logging, meetings | 2 | Late in getting writeup submitted, but on time for everything else. | 1.5 |
| Total | 15 |  | 17.5 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Meeting** | **Date** | | **Present & Contributing** |
| Initial discussion | 4/3/2012 | | Chris Wald, Dillon Hiatt |
| Final Review | 4/11/2012 | | Chris Wald, Dillon Hiatt |
| **Team Member** | **Hours** | **Explanation, if necessary** | |
| Dillon Hiatt | 3 |  | |
| Chris Wald | 2:59 |  | |
| Zach Gerner | 3.50 |  | |
|  |  |  | |

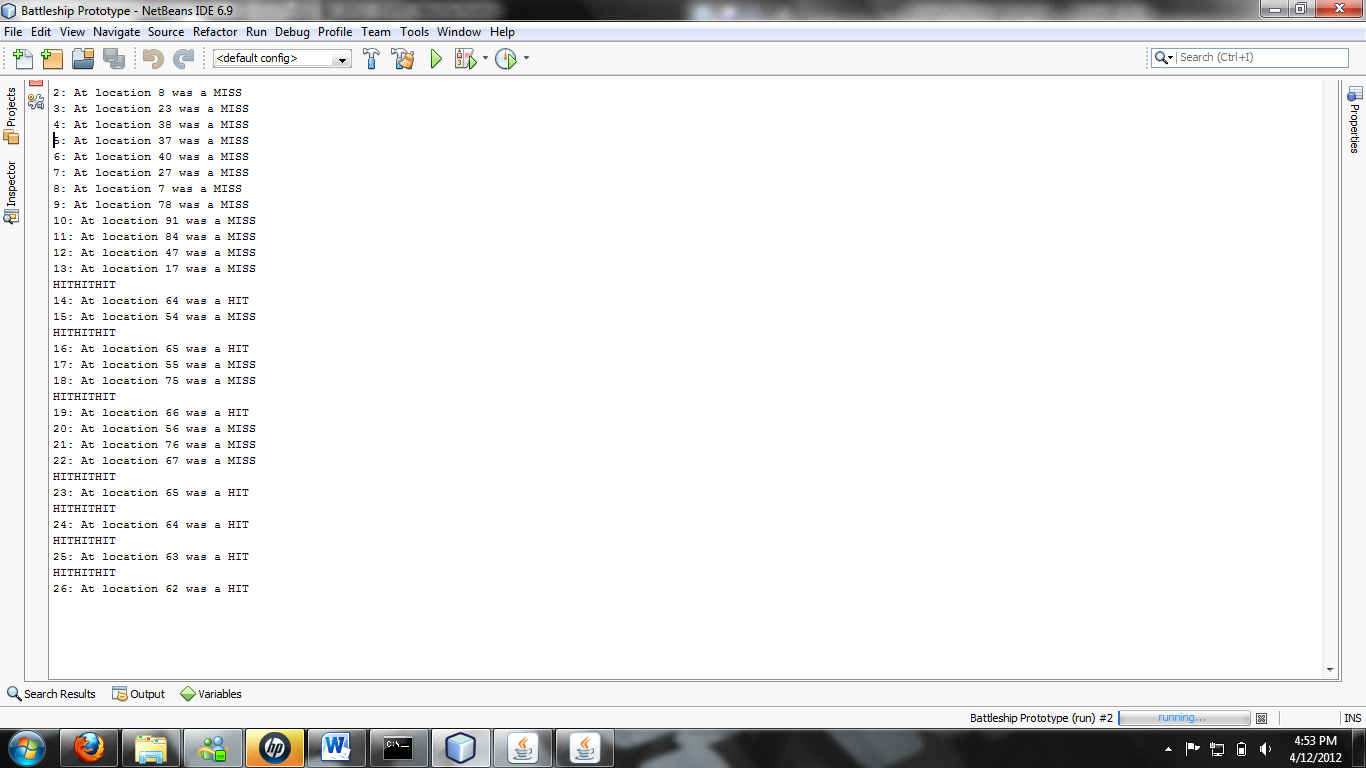
***Prototype Report:***

**Computer Player**

The function of the game that was being prototyped was computer player. The computer will choose a location on the grid by using a random generator. If the building has been hit, then it will guess the locations around the hit building. The computer player will pick a point that’s in the range of the grid. When all of the buildings are hit, then the computer player will stop guessing. The goal of this function is to have the computer player play against the user.

The project is built in a Java environment. One need only load up the project file in a Java IDE and it is ready to be tested. Once the user runs Prototyper.java the prototype is launched and the user is ready to be tested. There is no input that is needed to test this program.

From implementing this computer player, we have learned how to effectively guess locations that are in the range of our 10x10 grid and having the computer player guessing the location intelligently when hitting a building. The implementation for guessing in range was not difficult to implement, but the difficulty came when the computer player guesses where one of the buildings is located and then guesses that same location again.



**Fit to grid**

The function being prototyped was the ability for the system to take objects of varying lengths and snap them to a particular place on the 10x10 grid. The goal of this function is to make it easy for the player to select a building and place it onto the grid.

The project is built in a Java environment. One need only load up the project file in a Java IDE and it is ready to be tested. Once the user runs SnapProto.java the GUI is launched and the user is ready to test the prototype. The only input the prototype accepts is mouse clicks. Simply select the radio button of the building you wish to place and then select a grid tile where you would like to place the building.

[screenshots]

\*\*\*\*\*\*\*\*\*\*

[/screenshots]

From implementing this snap-to-grid functionality we learned how to efficiently draw our buildings onto the grid and how to neatly place the object onto an appropriate set of grid tiles. We believe this method is the easiest and most intuitive way for the user to place their buildings. The snap-to-grid functionality in itself was not very hard to learn or implement, but rather the difficulty in our prototype came from implementing restrictions to the snap-to-grid function such as disallowing users to place a building on top of another building or placing a building off of the grid.