CS 246

Final Group Project

Constructor Demo

Group members:
Chengxu Han (Infinite, c38han)
Yangzhou Han (Bernadette, y236han)
Zijian Feng (Alfred, z63feng)

Dec. 12th, 2020

Command-line Interface Implementation

（1） ./constructor

The constructor will search for the layout.txt file in the current directory and check the existence and the readability of the file. We provided a layout.txt file, the program will generate a board according to the data record in the file. If the file does not exist, the program will display an error message.

（2） ./constructor -seed 666

The constructor will generate a board by the seed number 666. If the seed number is not provided, the program will always generate the same board when you run the program.

（3）./constructor -load loadfile.txt

We provide the loadfile.txt which contains the information of current turn, the data of four builders, the board layout and the location of the geese.

（4） ./constructor -board board.txt

We provide the board.txt which contains the information of the resource and the value for each tile.

（5） ./constructor -random-board

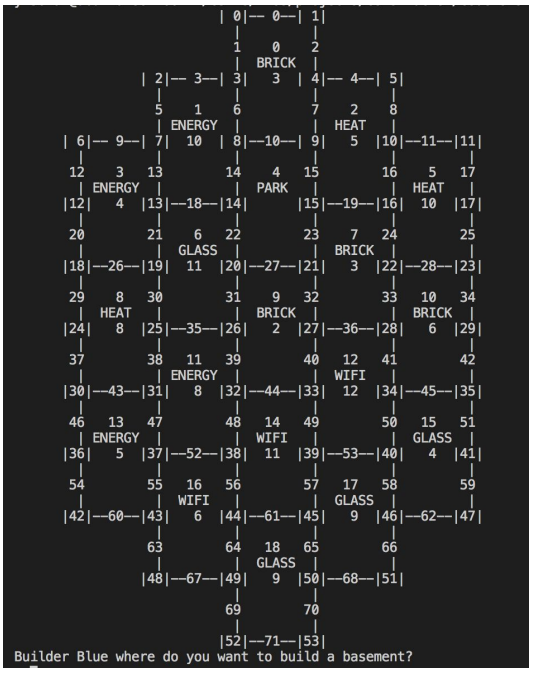
The program will randomly generate a board.

（6） ./constructor -random-board -seed 555

Gameplay 1 (Start a New Game/Beginning of Game)

./constructor

The beginning phrase starts, displays the board which is generated according to layout.txt and asks the first builder Blue to build a basement.



> 20

The player Blue successfully built a basement at 20. And the game will ask the next play Red to build a basement.

> 20

Since player Blue already built at 20, the game would display the message “you cannot build here. Basements already exist as locations: 20 ” and ask builder Red where to build the basement again.

> 21

Since player Blue already built a basement at 20, and 21 is adjacent to 20. Red could not build at 21 according to the rule. The game will display the same message again to warn player Red cannot build and ask where to build again.

> 28

> 52

> 48

> 2

> 8

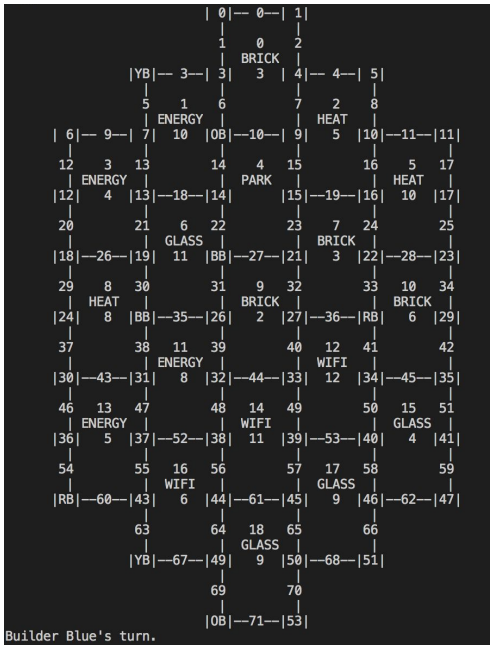
> 66

> 42

>25

The basements will be chosen by builders in order Blue, Red, Orange, Yellow, Yellow, Orange, Red, Blue. When Red input 66 which is greater than the total number of vertice, the game will display a message “You cannot build here.” Also, display all the basement current build as “Basements already exist as locations: 20 28 52 48 2 8 ”. The game asked Red again. Red built a basement at 42 and Blue built a basement at 25. All the builders placed two basements. The updated board displayed and the game started.

```
> 20
Builder Red where do you want to build a basement?
> 20
You cannot build here.
Basements already exist as locations: 20
Builder Red where do you want to build a basement?
> 21
You cannot build here.
Basements already exist as locations: 20
Builder Red where do you want to build a basement?
> 28
Builder Orange where do you want to build a basement?
> 52
Builder Yellow where do you want to build a basement?
> 48
Builder Yellow where do you want to build a basement?
> 2
Builder Orange where do you want to build a basement?
> 8
Builder Red where do you want to build a basement?
> 66
You cannot build here.
Basements already exist as locations: 20 28 52 48 2 8
Builder Red where do you want to build a basement?
> 42
Builder Blue where do you want to build a basement?
> 25
```



> help

The game prompts it is Blue's turn. And Blue input help to see all the available commands. The commands are:

> load

The builder Blue chose a load dice. The game displayed the message "Builder Blue now has Load Dice."

> roll

> 11

The builder Blue rolled a load dice, the game will prompt "Input a roll between 2 and 12:". Blue roll a 11. Since Build had residence in tile 11, the game will notify Blue that the builder gains 2 Glass.

> status

Print out all builder's current status. We can see all builders have two building points, and Blue gained 2 Glasses from the previous step.

> help

The builder Blue entered help to see all the current available command,

> residences

Print out all the residences built by builder Blue in increasing order.

```
> help
Valid commands:
~ load : changes current builder's dice type to 'loaded'
~ fair : changes current builder's dice type to 'fair'
~ roll : rolls the dice and distributes resources.
~ status : prints the current status of all builders in order from builder 0 to 3.
~ help : prints out the list of commands.
> load
Builder Blue now has loaded Dice.
> roll
Input a roll between 2 and 12:
> 11
Builder Blue gained:
2 GLASS
Enter a command:
> status
Builder Blue    has 2 building points, 0 BRICK, 0 ENERGY, 2 GLASS, 0 HEAT, 0 WIFI.
Builder Red     has 2 building points, 0 BRICK, 0 ENERGY, 0 GLASS, 0 HEAT, 0 WIFI.
Builder Orange  has 2 building points, 0 BRICK, 0 ENERGY, 0 GLASS, 0 HEAT, 0 WIFI.
Builder Yellow  has 2 building points, 0 BRICK, 0 ENERGY, 0 GLASS, 0 HEAT, 0 WIFI.
Enter a command:
> help
Valid commands :
~ board : prints the current board.
~ status : prints the current status of all builders in order from builder 0 to 3.
~ residences : prints the residences the current builder has currently completed.
~ build - road <road#> : attempts to builds the road at <road#>.
~ build - res <housing#> : attempts to builds a basement at <housing#>.
~ improve <housing#> : attempts to improve the residence at <housing#>.
~ trade <colour> <give> <take> : attempts to trade with builder <colour>, giving one resource of type <give> and receiving one resource of type
<take>.
~ next : passes control onto the next builder in the game.
~ save <file> : saves the current game state to <file>.
~ help : prints out the list of commands.
Enter a command:
> residences
Blue has built:
20 B
25 B
Enter a command:
```

> next

Print out the current board. Builder Red is prompted.

> fair

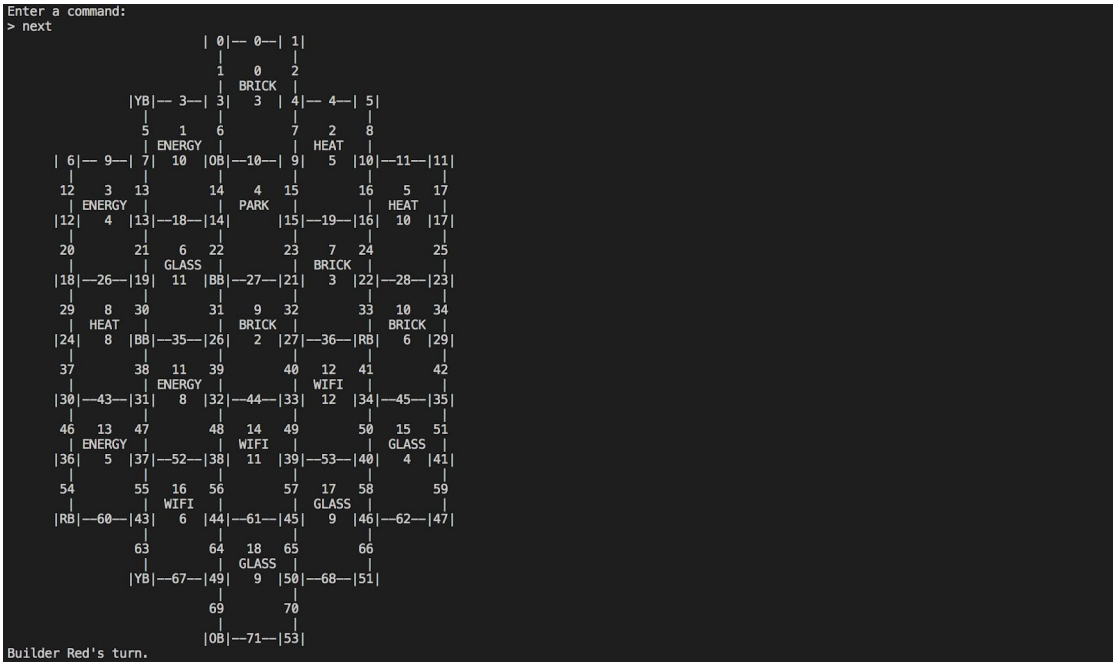
Red choosed to roll a fair dice. Show the message “Builder Red now has fair Dice.”

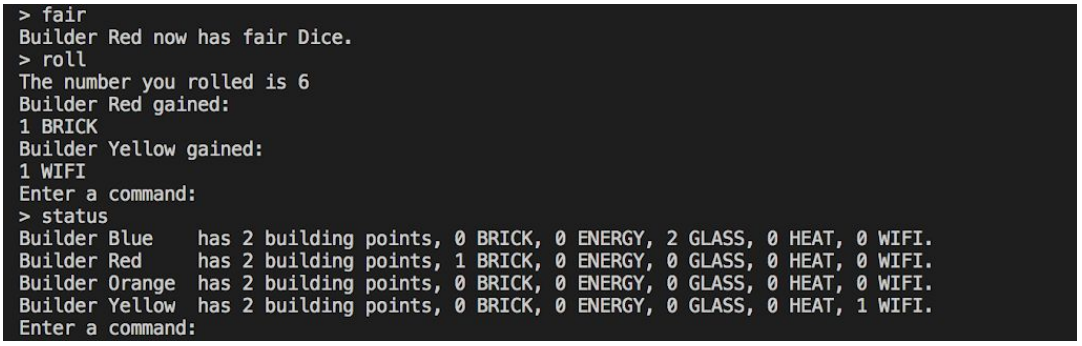
> roll

Red rolled a fair dice and the game displayed a roll number. If the corresponding tile had residences, the game would display a message to show the resource distribution had been made with the builder name and the number of resources that builder gained. If the tile did not have any residence, the game would display a message stating no builder gained resources.

> status

Print out all builder’s current status.

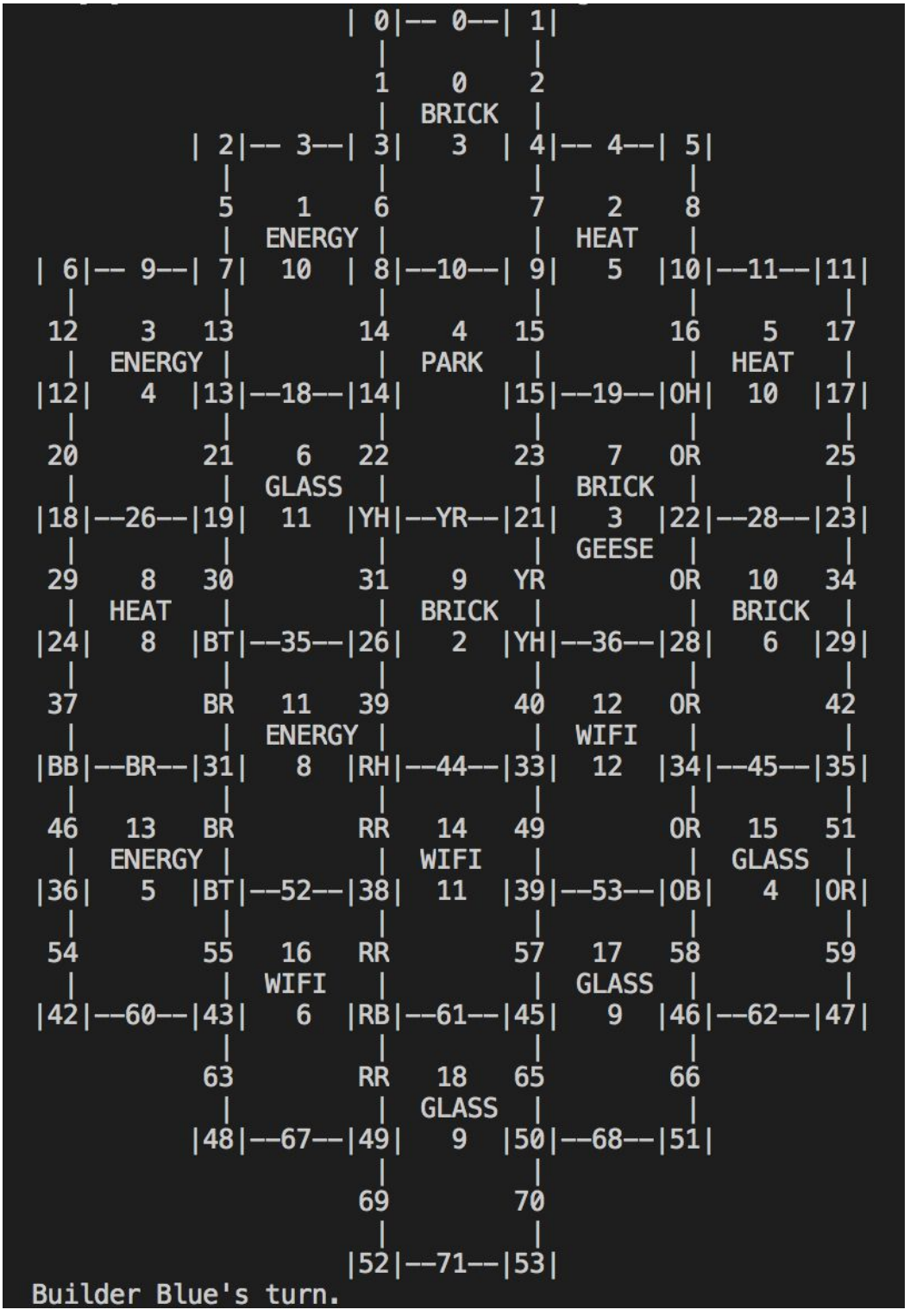




Gameplay 2 (During the Turn && Geese)

./constructor -load loadfile.txt

A board is generated according to the information given in loadfile.txt. Builder blue is prompted with the message “Builder Blue’s turn.”



> help

> load

> roll

> 7

Blue rolled a load dice with a number 7. Since the builder Blue has 50 resources in total and the builder Orange has 11 resources in total, both with more than 10 resources. Thus, Blue and Orange lost half resources which are chosen at random. The board will display the message showing what resource they lost. And then ask Blue where to place the geese.

> 10

Blue places the geese at 10. Since there are no residences built in tile 10. The message "Builder Blue has no builder to steal from ." is prompted and also asks Blue to enter a command.

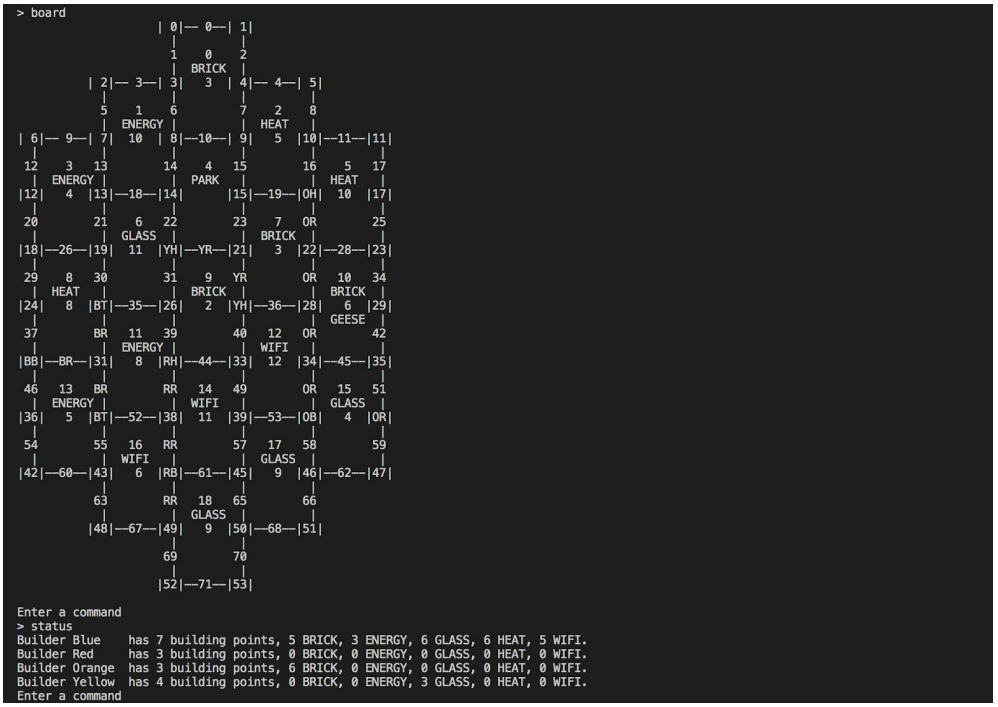
```
> help
Valid commands:
~ load : changes current builder's dice type to 'loaded'
~ fair : changes current builder's dice type to 'fair'
~ roll : rolls the dice and distributes resources.
~ status : prints the current status of all builders in order from builder 0 to 3.
~ help : prints out the list of commands.
> load
Builder Blue now has loaded Dice.
> roll
Input a roll between 2 and 12:
> 7
Now you have : 7
Builder Blue loses 25 resources to the geese. They lose:
5 BRICK
7 ENERGY
4 GLASS
4 HEAT
5 WIFI
Builder Orange loses 5 resources to the geese. They lose:
5 BRICK
Choose where to place the GEESE.
> 10
Builder Blue has no builders to steal from.
Enter a command
```

> board

The current board is displayed. We can see the geese remove from tile 7 to tile 10.

> status

Prints the current status of all builders. Blue originally has 10 resources of each type. Assume 5 brick, 5 energy, 4 glass, 5 heat and 6 wifi was lost when Blue rolled a 7. The status of Blue will show builder Blue has 5 bricks, 5 energy, 6 glass, 5 heat, 4 wifi now.



> trade yellow wifi glass

> yes

> status

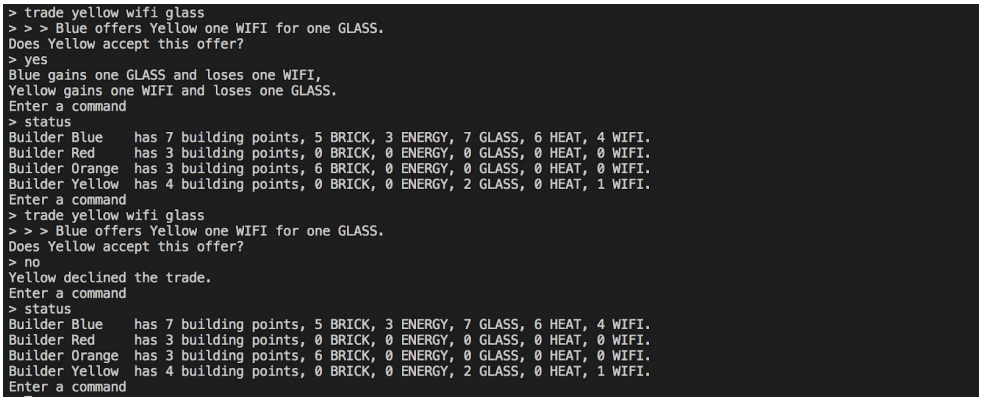
Builder Blue asks builder Yellow to make a trade. Blue wants to exchange one WIFI for one Glass. Builder Yellow accepted, and from status we can see the trade is successfully completed.

> trade yellow wifi glass

> no

> status

Builder Blue asks builder Yellow to make the previous trade again. Blue wants to exchange one WIFI for one Glass. Builder Yellow declined the trade, and from status we can see the builder's data remain the same.



> residences

Print out all the residences in increasing order of vertex numbers built by builder Blue. Blue has built a Tower at 25, a Basement at 30 and a Tower at 37.

> improve 25

There is a tower built at vertex 25, which cannot be improved. The message “You cannot improve that building” is prompted.

> improve 30

> status

Blue successfully improved its basement at vertex 30 to a house. From status, we can see 2 Glass and 3 Heat are deducted

```
> residences
Blue has built:
25 T
30 B
37 T
Enter a command
> improve 25
You can't improve that building.
Enter a command
> improve 30
Builder Blue successfully built a House at 30.
Enter a command
> status
Builder Blue    has 8 building points, 5 BRICK, 3 ENERGY, 5 GLASS, 3 HEAT, 4 WIFI.
Builder Red     has 3 building points, 0 BRICK, 0 ENERGY, 0 GLASS, 0 HEAT, 0 WIFI.
Builder Orange  has 3 building points, 6 BRICK, 0 ENERGY, 0 GLASS, 0 HEAT, 0 WIFI.
Builder Yellow  has 4 building points, 0 BRICK, 0 ENERGY, 2 GLASS, 0 HEAT, 1 WIFI.
Enter a command
```

> board

Print out the current board, we can see the vertex change from “BB” to “BH”.

> build-road 38

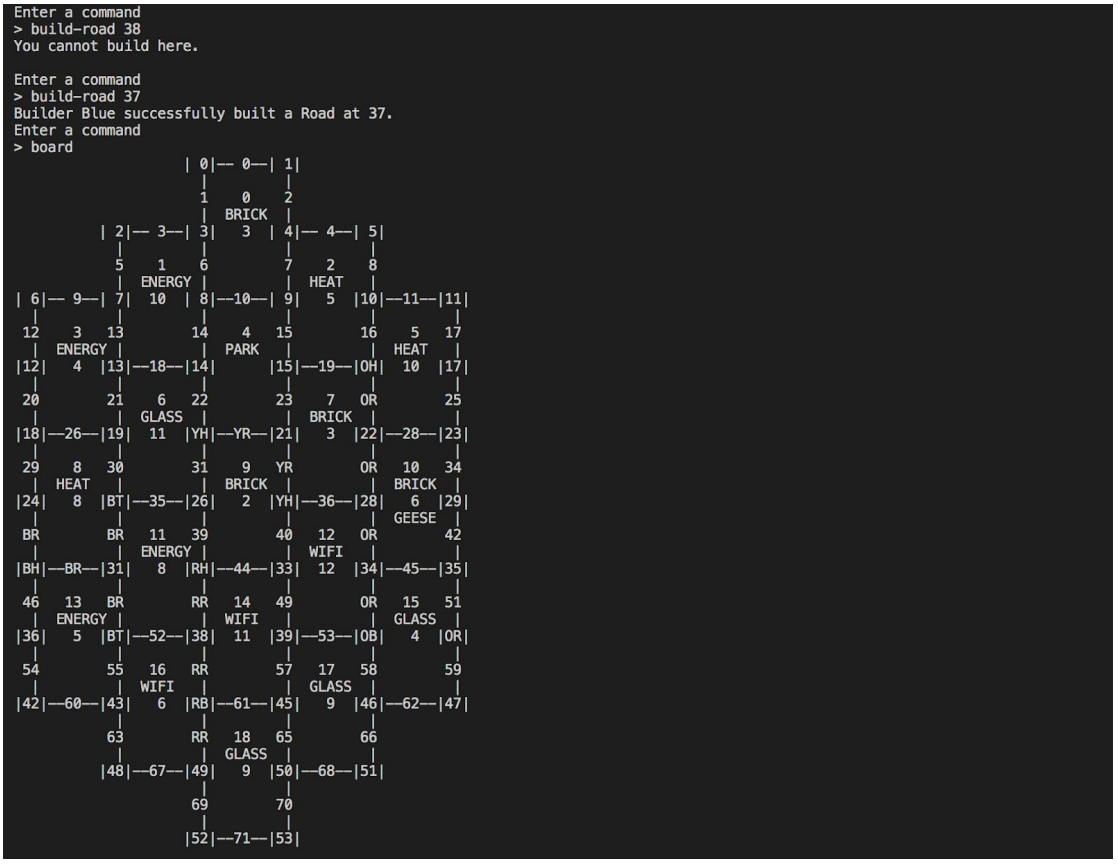
Builder Blue wants to build a road at edge 38, but there already exist roads built by Blue. Thus, the “You cannot build here.” message is prompted.

> build-road 37

Blue tries to build a road at edge 37. Since vertex 37 is adjacent to vertex 30 where a basement built by Blue, Blue can successfully build a road, also the game would prompt a message showing the completion.

> board

The board shows builder Blue built a road on edge 37.



> build-res 38

Blue tries to build a residence at vertex 38. Since vertex 38 is adjacent to the vertex with a residence built by Red, Blue cannot build here according to the game rule. The "You cannot build here." message is prompted.

> build-res 48

Blue tries to build a residence at vertex 48. Since the vertex is not adjacent to any road built by Blue. The "You cannot build here." message is prompted.

> build-road 29

> build-res 18

> board

Blue successfully built a road in edge 29 and a basement at vertex 18. We can see those changes from the board.

Gameplay 3 (End of Game)

> board -load endOfGame.txt

> load

> roll

> 5

> status

> build-res 13

> yes

> load

> roll

> 5

> status

> build-res 13

> no

```
                    |52|--71--|55|
Builder Blue's turn.
> load
Builder Blue now has loaded Dice.
> roll
Input a roll between 2 and 12:
> 5
Now you have : 5
Builder Blue gained:
6 ENERGY
Builder Orange gained:
2 HEAT
Enter a command
> status
Builder Blue    has 9 building points, 10 BRICK, 16 ENERGY, 10 GLASS, 10 HEAT, 10 WIFI.
Builder Red     has 3 building points, 0 BRICK, 0 ENERGY, 0 GLASS, 0 HEAT, 0 WIFI.
Builder Orange  has 3 building points, 11 BRICK, 0 ENERGY, 0 GLASS, 2 HEAT, 0 WIFI.
Builder Yellow  has 4 building points, 0 BRICK, 0 ENERGY, 3 GLASS, 0 HEAT, 0 WIFI.
Enter a command
> build-res 13
Builder Blue successfully built a Basement at 13.

Builder Blue reach 10 BUILDING POINTS!
===========Builder Blue Win!==============
Would you like to play again?
no
Bye Bye!
```