GIT Department of Computer Engineering
CSE 222/505 - Spring 2022
Homework 1 Report

Emircan Demirel
1901042674

1. SYSTEM REQUIREMENTS

1.1 Functional Requirements

This program creates and designs streets to use it’s functions, a Street must be created and initialized.

A Street can be initialized by giving it’s length property. Length must be positive integer to create Street.

After initializing parting is done, software is ready to use. User can both access editing and viewing modes of program via command menu:

Editing Mode:

- User can add an instance of Buildings by giving it’s necessary properties.
- User also delete a Building from Street by entering it’s location position and side.

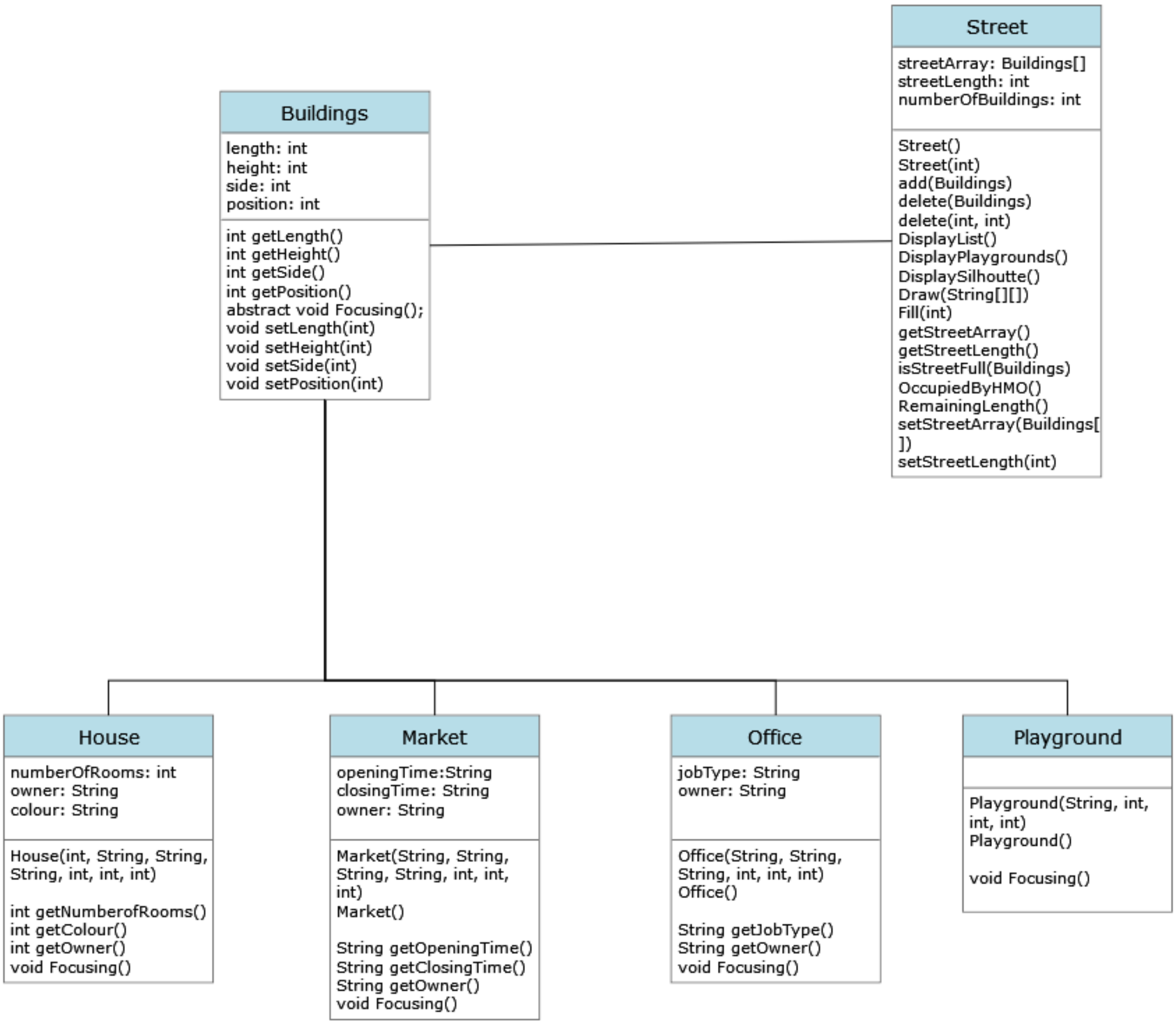
Viewing Mode:

- User can access RemainingLength() method of Street Class to display the total remaining length of lands on the street
- User can access DisplayList() method of Street Class to display the list of buildings on the street.
- With the help of DisplayPlayground() method of Street Class, the user could display the number and ratio of length of playgrounds in the street.
- OccupiedByHMO() method calculates the total length of street occupied by the markets, houses or offices.
- DisplaySilhoutte() method display the skyline silhouette of the street.

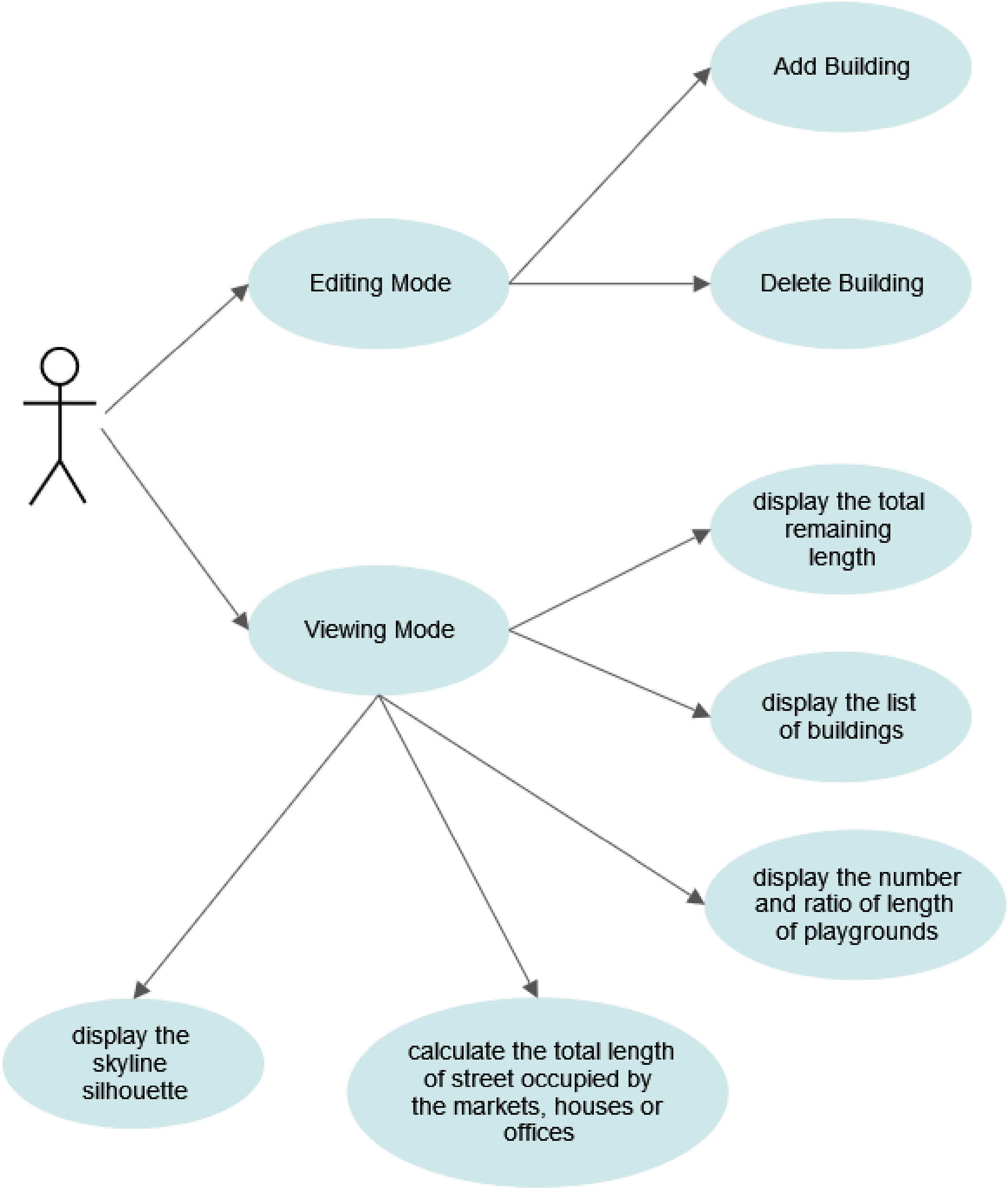
1.2 Non-functional Requirements

- Add method blocks to add a building if given position or street is full.
- Delete method blocks to delete if given position is already empty.
- numberOfBuildings variable keeps to total number of buildings for a street.
- streetLength variable must be a positive integer to be able to create streetArray.
- User inputs shouldn’t throw exceptions which unhandled.

2. CLASS DIAGRAMS



3. USE CASE DIAGRAM


Add Building
Editing Mode
Delete Building
display the total remaining length
Viewing Mode
display the list of buildings
display the number and ratio of length of playgrounds
display the skyline silhouette
calculate the total length of street occupied by the markets, houses or offices

4. PROBLEM SOLUTION APPROACH

In this homework data always and only stored in an array. When user wants to add/delete different types of buildings he/she must be change array dynamically. To determine size of array, I used numberOfBuildings variable. In every work of add method numberOfBuildings has increased (the opposite way of delete method). Then I create an array with size of numberOfBuildings and initialize every element of old array to this new array. At the end the new array set as streetArray.

Second problem is Focusing method which must be implemented by every type of Building. Instead of early binding, every method has different implementation on it. To obtained it I used polymorphism and inheritance. With the help of "abstract" and "extends" keyword, every implementation has done separately in subclasses.

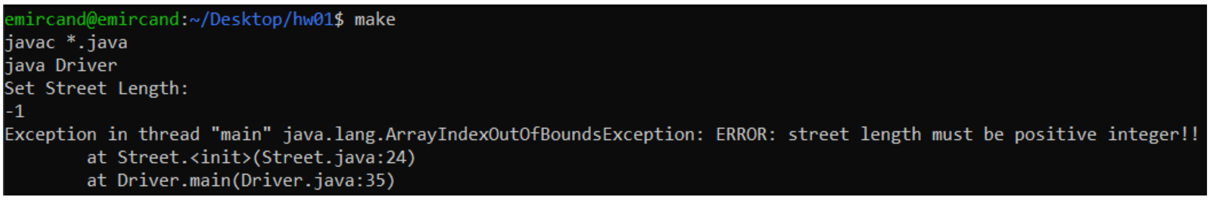
The last problem is printing skyline silhouette to the console. streetArray keeps position in unsorted way and some of positions also doesn't contain anything. To print in silhouette view, firstly I write an algorithm which finds a maximum height value in Street. Then, I create 2D array in size of (max height * street length). This array filled by upper corner points of buildings. After that, upper corners of buildings are combined by "*" strings. I completed the street view with lines descending from the upper corners to the floor. At the end I create conditional statements to get silhouette view.

5. TEST CASES

1. Compile -> Set Street Size -> Invalid Input
2. Compile -> Menu -> Valid Input
3. Compile -> Menu -> Invalid Input
4. Compile -> Menu -> Editing Mode -> Add -> New Location -> Valid Input
5. Compile -> Menu -> Editing Mode -> Add -> New Location -> Invalid Input
6. Compile -> Menu -> Editing Mode -> Add -> Try Used Location
7. Compile -> Menu -> Editing Mode -> Delete -> Valid Input
8. Compile -> Menu -> Editing Mode -> Delete -> Invalid Input
9. Compile -> Menu -> Editing Mode -> Delete -> Try Empty Position
10. Compile -> Menu -> Viewing Mode -> Display the total remaining length of lands
11. Compile -> Menu -> Viewing Mode -> Display the list of buildings
12. Compile -> Menu -> Viewing Mode -> Display the number and ratio of length of playgrounds
13. Compile -> Menu -> Viewing Mode -> Calculate the total length of street occupied by the markets, houses or offices
14. Compile -> Menu -> Viewing Mode -> Display the skyline silhouette
15. Compile -> Menu -> Viewing Mode -> Focusing Methods

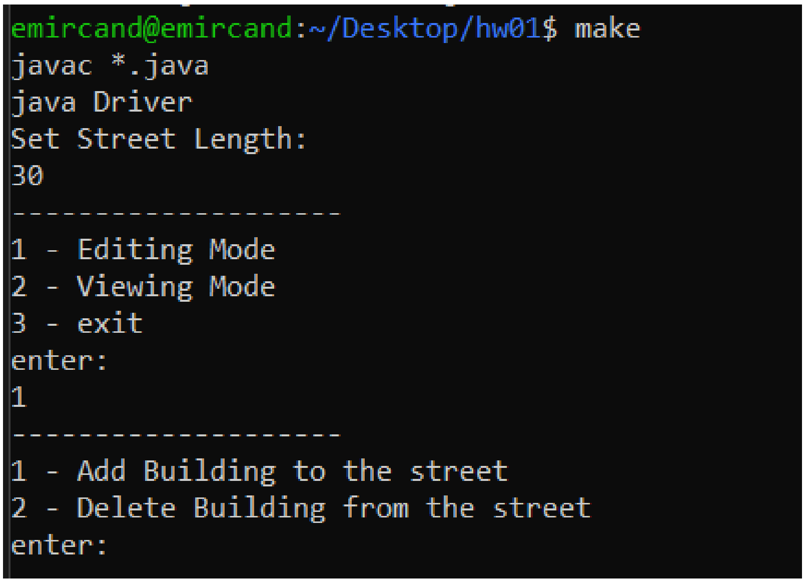
6. RUNNING AND RESULTS

1.



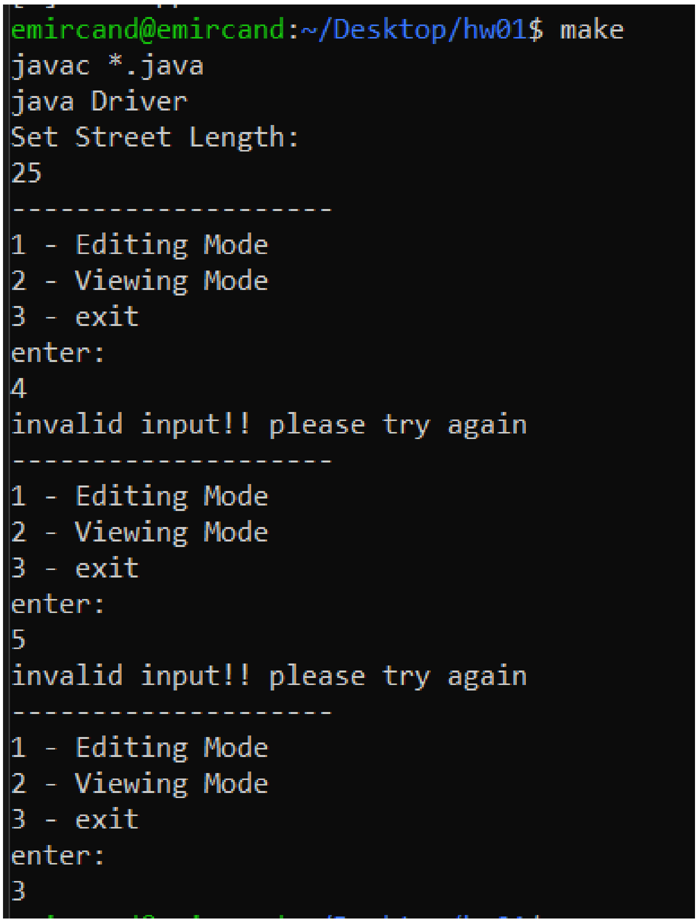
```
emircand@emircand:~/Desktop/hw01$ make
javac *.java
java Driver
Set Street Length:
-1
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: ERROR: street length must be positive integer!!
        at Street.<init>(Street.java:24)
        at Driver.main(Driver.java:35)
```

2.



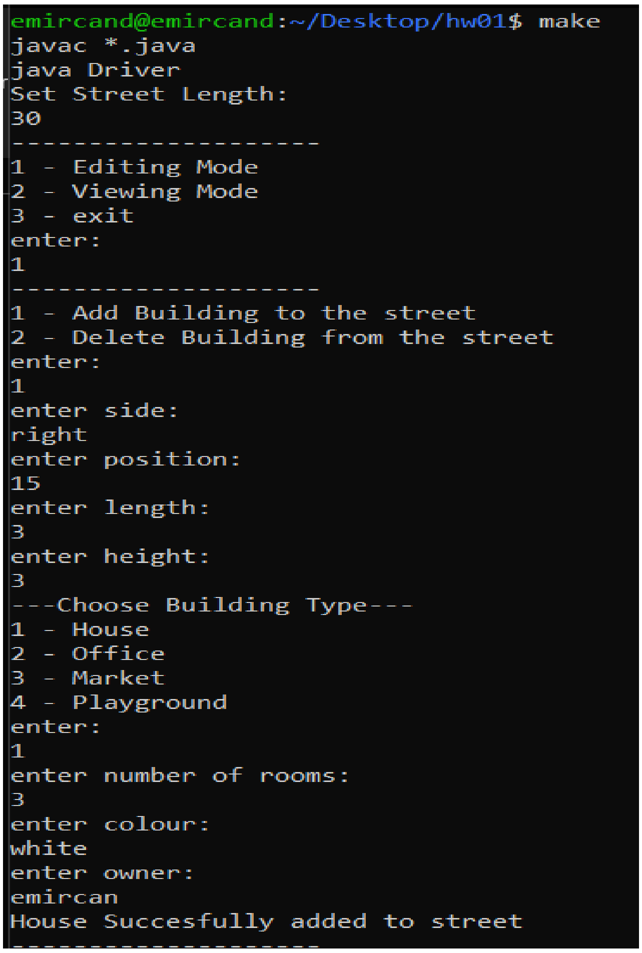
```
emircand@emircand:~/Desktop/hw01$ make
javac *.java
java Driver
Set Street Length:
30
--------------------
1 - Editing Mode
2 - Viewing Mode
3 - exit
enter:
1
--------------------
1 - Add Building to the street
2 - Delete Building from the street
enter:
```

3.



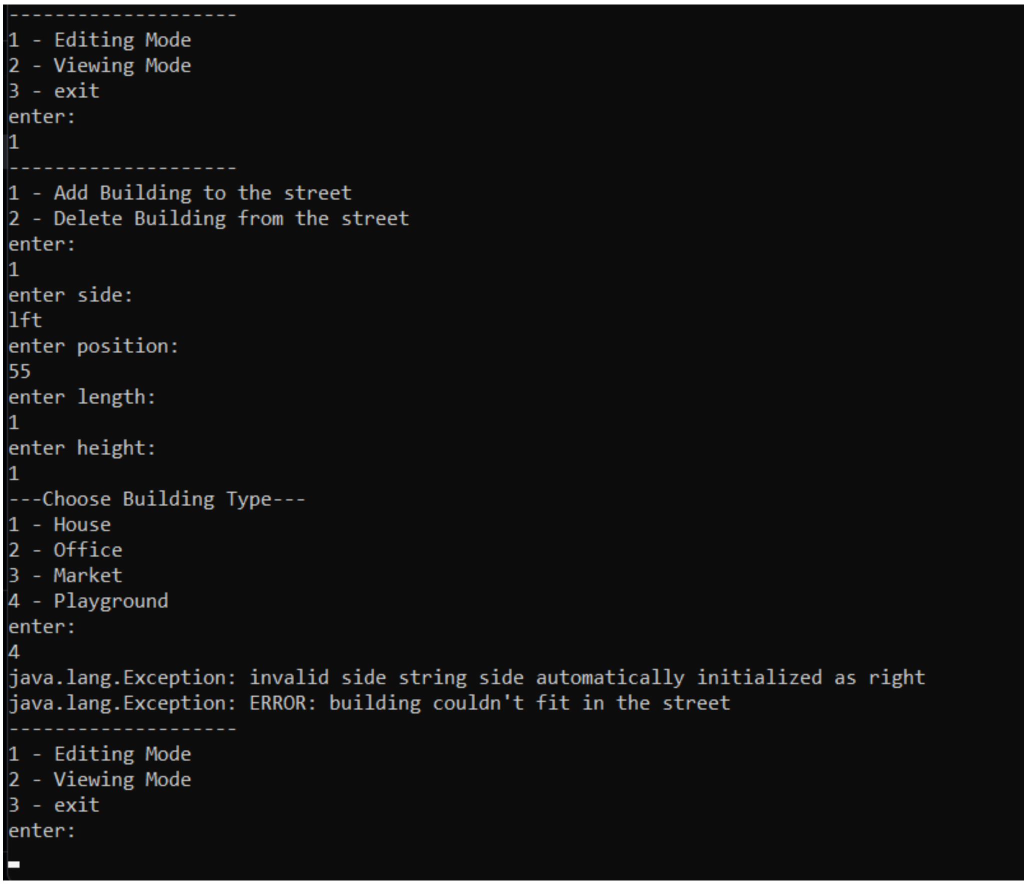
```
emircand@emircand:~/Desktop/hw01$ make
javac *.java
java Driver
Set Street Length:
25
--------------------
1 - Editing Mode
2 - Viewing Mode
3 - exit
enter:
4
invalid input!! please try again
--------------------
1 - Editing Mode
2 - Viewing Mode
3 - exit
enter:
5
invalid input!! please try again
--------------------
1 - Editing Mode
2 - Viewing Mode
3 - exit
enter:
3
```

4.

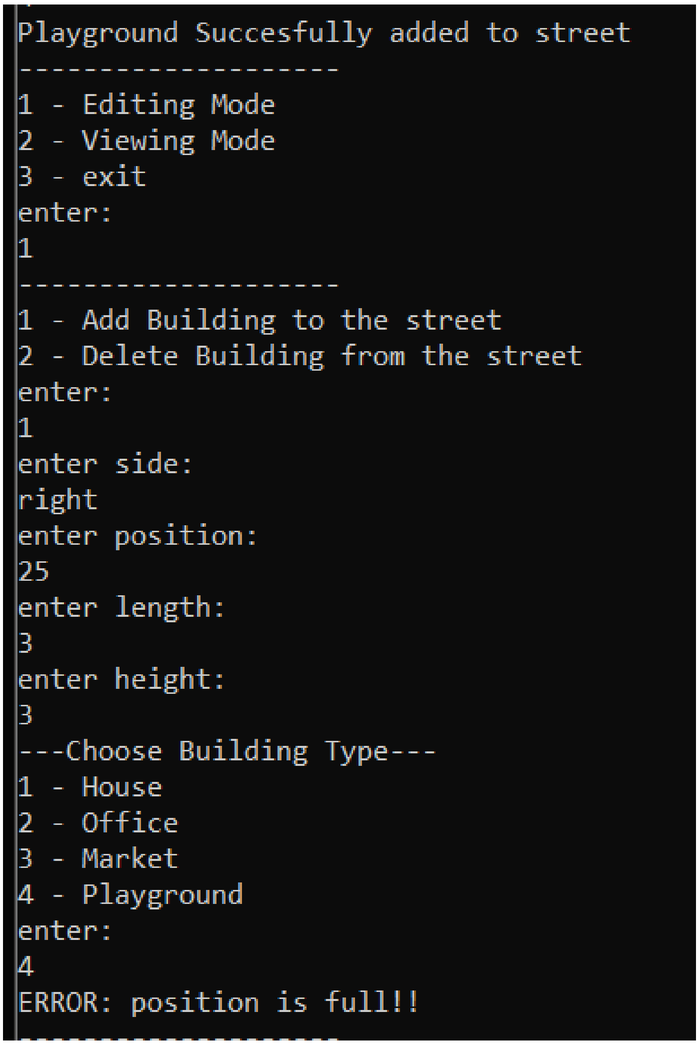


```
emircand@emircand:~/Desktop/hw01$ make
javac *.java
java Driver
Set Street Length:
30
--------------------
1 - Editing Mode
2 - Viewing Mode
3 - exit
enter:
1
--------------------
1 - Add Building to the street
2 - Delete Building from the street
enter:
1
enter side:
right
enter position:
15
enter length:
3
enter height:
3
---Choose Building Type---
1 - House
2 - Office
3 - Market
4 - Playground
enter:
1
enter number of rooms:
3
enter colour:
white
enter owner:
emircan
House Succesfully added to street
--------------------
```

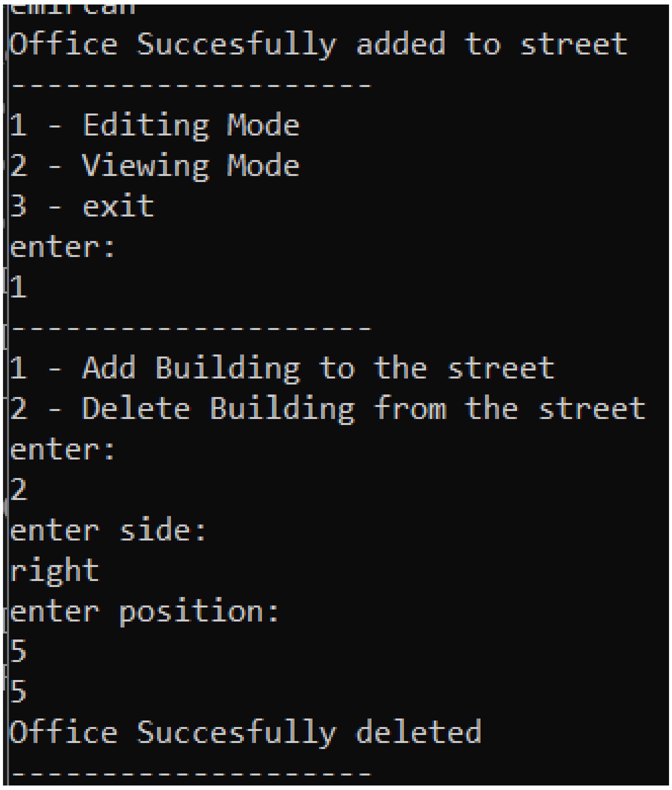
5.



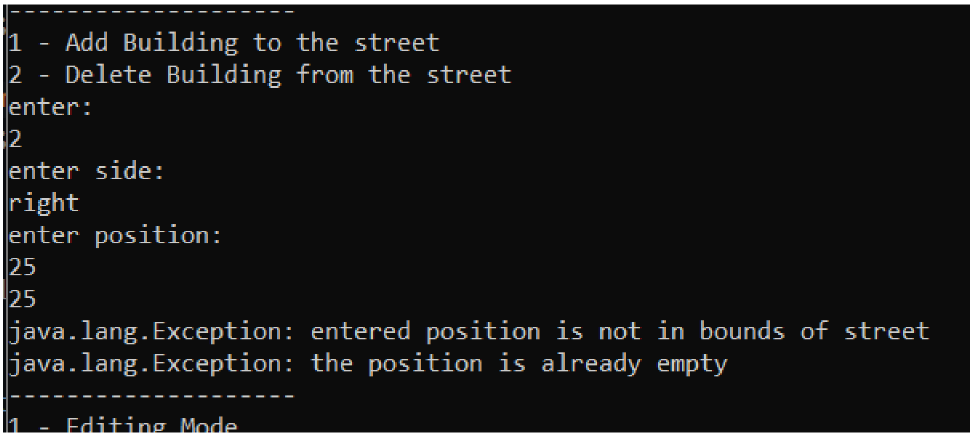
```
--------------------
1 - Editing Mode
2 - Viewing Mode
3 - exit
enter:
1
--------------------
1 - Add Building to the street
2 - Delete Building from the street
enter:
1
enter side:
lft
enter position:
55
enter length:
1
enter height:
1
---Choose Building Type---
1 - House
2 - Office
3 - Market
4 - Playground
enter:
4
java.lang.Exception: invalid side string side automatically initialized as right
java.lang.Exception: ERROR: building couldn't fit in the street
--------------------
1 - Editing Mode
2 - Viewing Mode
3 - exit
enter:
```

6. 

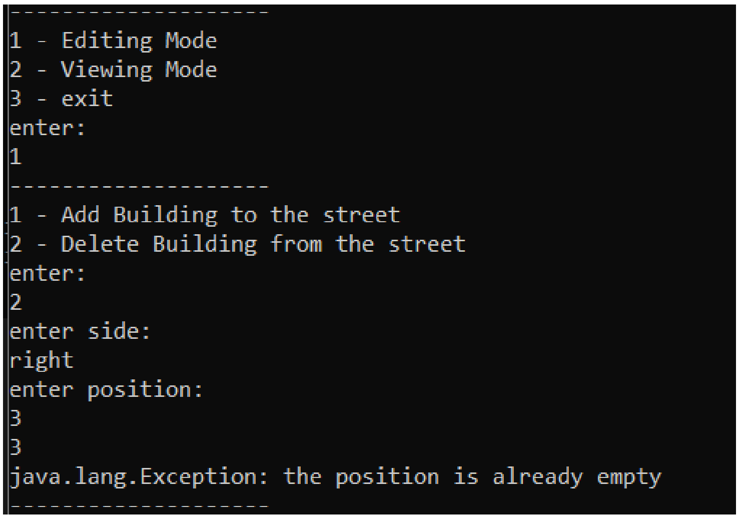
```
Playground Succesfully added to street
--------------------
1 - Editing Mode
2 - Viewing Mode
3 - exit
enter:
1
--------------------
1 - Add Building to the street
2 - Delete Building from the street
enter:
1
enter side:
right
enter position:
25
enter length:
3
enter height:
3
---Choose Building Type---
1 - House
2 - Office
3 - Market
4 - Playground
enter:
4
ERROR: position is full!!
--------------------
```

7. 

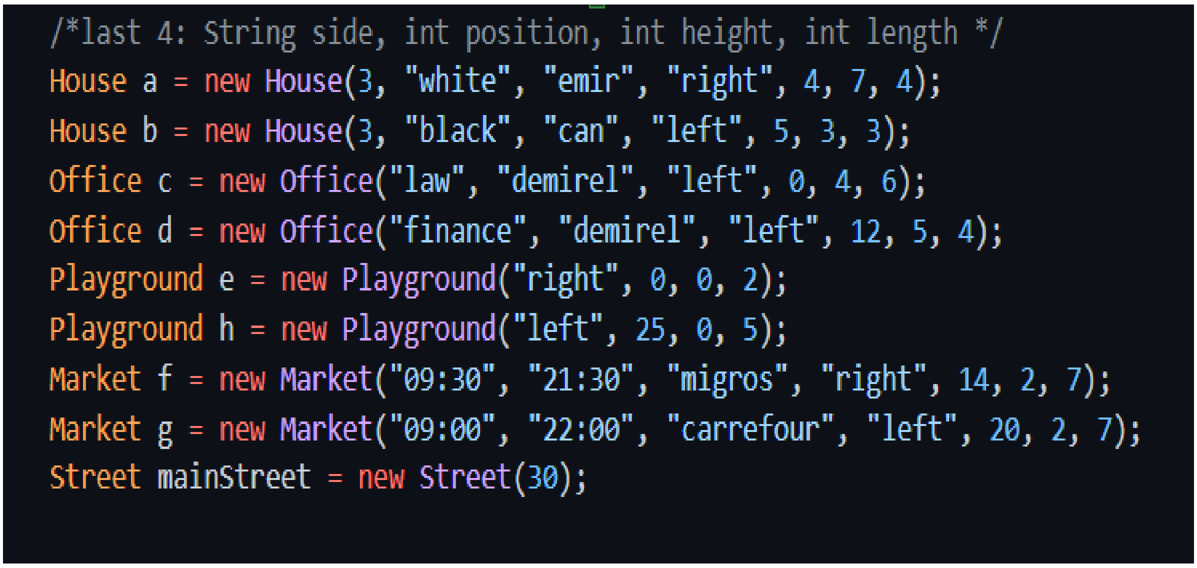
```
emircan
Office Succesfully added to street
--------------------
1 - Editing Mode
2 - Viewing Mode
3 - exit
enter:
1
--------------------
1 - Add Building to the street
2 - Delete Building from the street
enter:
2
enter side:
right
enter position:
5
5
Office Succesfully deleted
--------------------
```

8. 

```
--------------------
1 - Add Building to the street
2 - Delete Building from the street
enter:
2
enter side:
right
enter position:
25
25
java.lang.Exception: entered position is not in bounds of street
java.lang.Exception: the position is already empty
--------------------
1 - Editing Mode
```

9. 

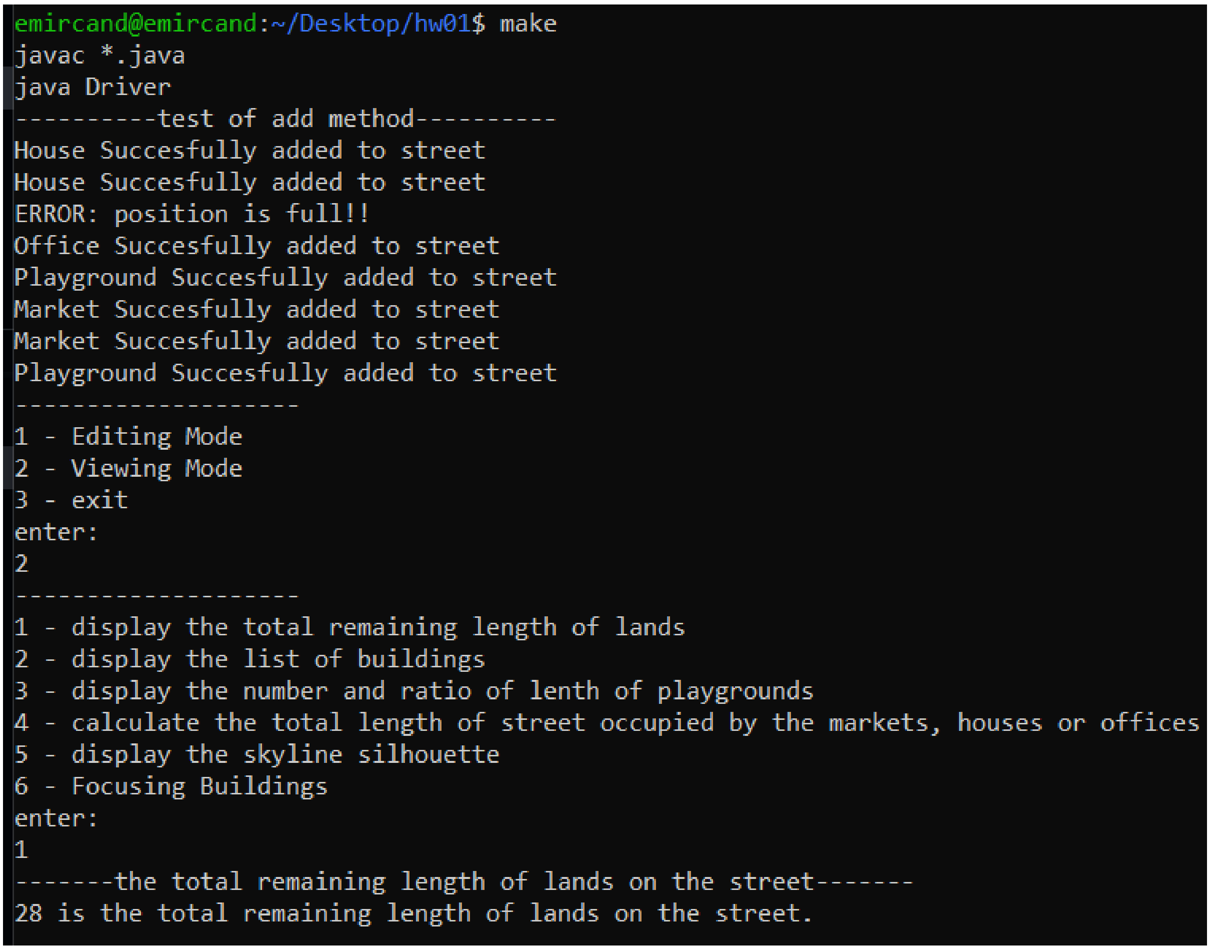
```
--------------------
1 - Editing Mode
2 - Viewing Mode
3 - exit
enter:
1
--------------------
1 - Add Building to the street
2 - Delete Building from the street
enter:
2
enter side:
right
enter position:
3
3
java.lang.Exception: the position is already empty
--------------------
```



```java
/*last 4: String side, int position, int height, int length */
House a = new House(3, "white", "emir", "right", 4, 7, 4);
House b = new House(3, "black", "can", "left", 5, 3, 3);
Office c = new Office("law", "demirel", "left", 0, 4, 6);
Office d = new Office("finance", "demirel", "left", 12, 5, 4);
Playground e = new Playground("right", 0, 0, 2);
Playground h = new Playground("left", 25, 0, 5);
Market f = new Market("09:30", "21:30", "migros", "right", 14, 2, 7);
Market g = new Market("09:00", "22:00", "carrefour", "left", 20, 2, 7);
Street mainStreet = new Street(30);
```

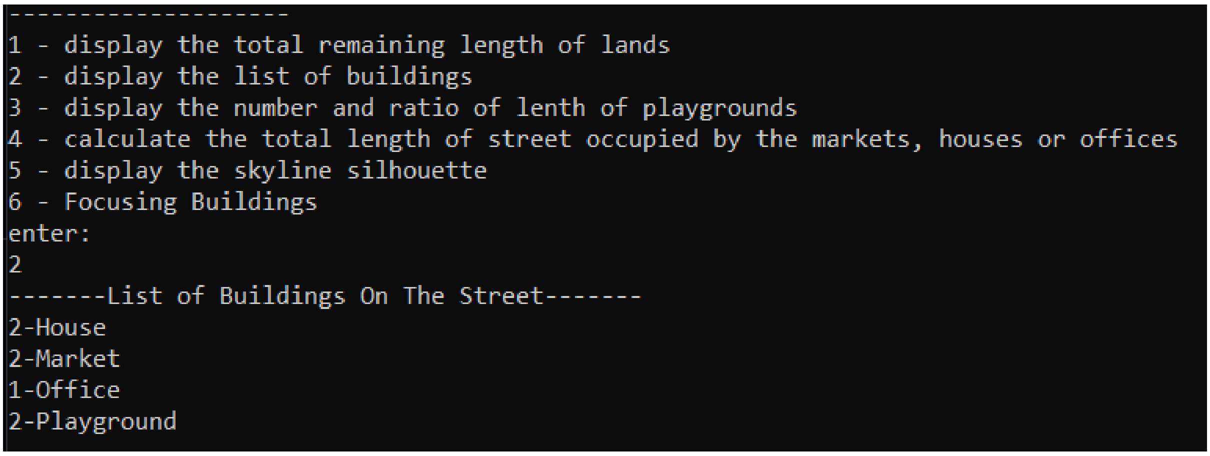
Figure 1 Inputs of next run cases

10.



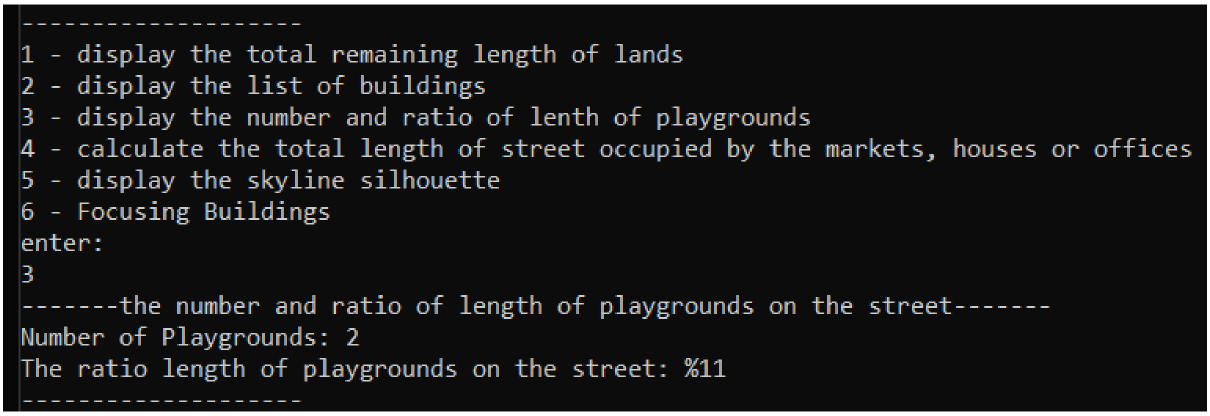
```
emircand@emircand:~/Desktop/hw01$ make
javac *.java
java Driver
----------test of add method----------
House Succesfully added to street
House Succesfully added to street
ERROR: position is full!!
Office Succesfully added to street
Playground Succesfully added to street
Market Succesfully added to street
Market Succesfully added to street
Playground Succesfully added to street
--------------------
1 - Editing Mode
2 - Viewing Mode
3 - exit
enter:
2
--------------------
1 - display the total remaining length of lands
2 - display the list of buildings
3 - display the number and ratio of lenth of playgrounds
4 - calculate the total length of street occupied by the markets, houses or offices
5 - display the skyline silhouette
6 - Focusing Buildings
enter:
1
-------the total remaining length of lands on the street-------
28 is the total remaining length of lands on the street.
```

11.



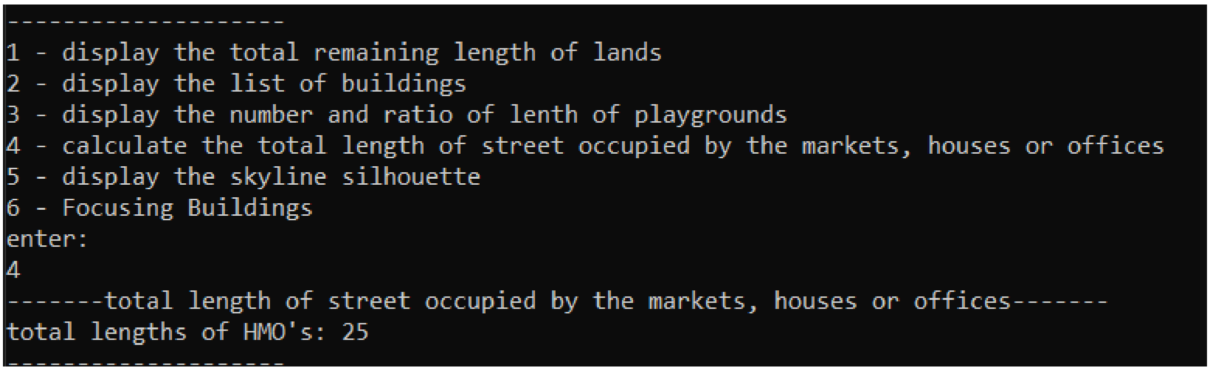
```
--------------------
1 - display the total remaining length of lands
2 - display the list of buildings
3 - display the number and ratio of lenth of playgrounds
4 - calculate the total length of street occupied by the markets, houses or offices
5 - display the skyline silhouette
6 - Focusing Buildings
enter:
2
-------List of Buildings On The Street-------
2-House
2-Market
1-Office
2-Playground
```

12.



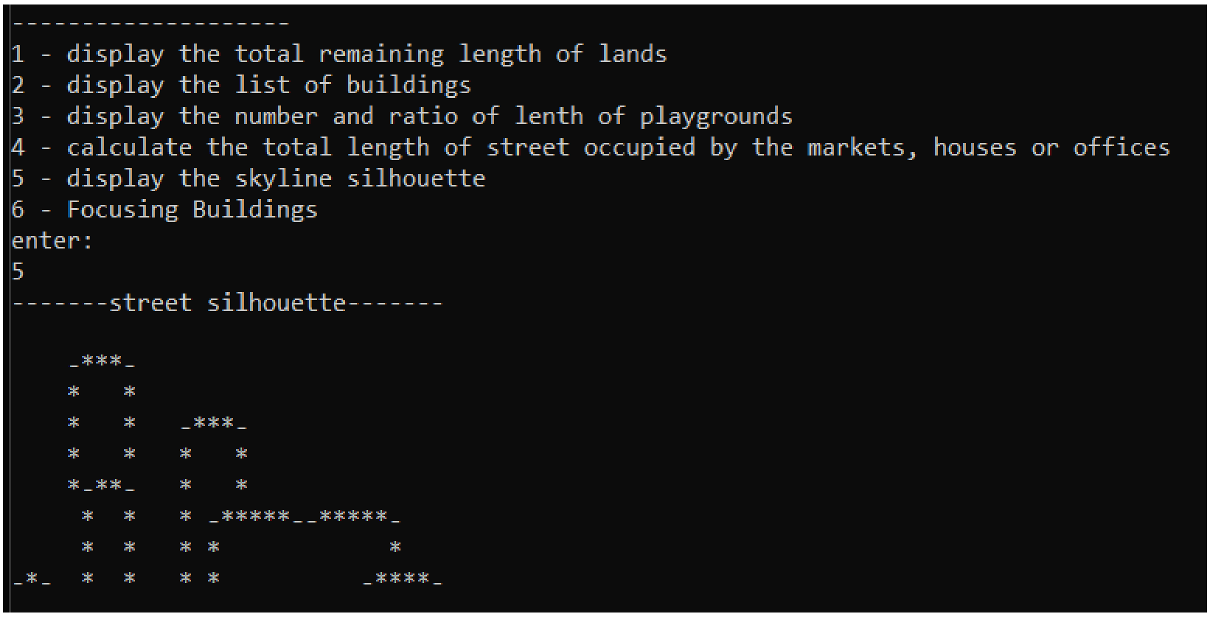
```
--------------------
1 - display the total remaining length of lands
2 - display the list of buildings
3 - display the number and ratio of lenth of playgrounds
4 - calculate the total length of street occupied by the markets, houses or offices
5 - display the skyline silhouette
6 - Focusing Buildings
enter:
3
-------the number and ratio of length of playgrounds on the street-------
Number of Playgrounds: 2
The ratio length of playgrounds on the street: %11
--------------------
```

13.



```
--------------------
1 - display the total remaining length of lands
2 - display the list of buildings
3 - display the number and ratio of lenth of playgrounds
4 - calculate the total length of street occupied by the markets, houses or offices
5 - display the skyline silhouette
6 - Focusing Buildings
enter:
4
-------total length of street occupied by the markets, houses or offices-------
total lengths of HMO's: 25
--------------------
```

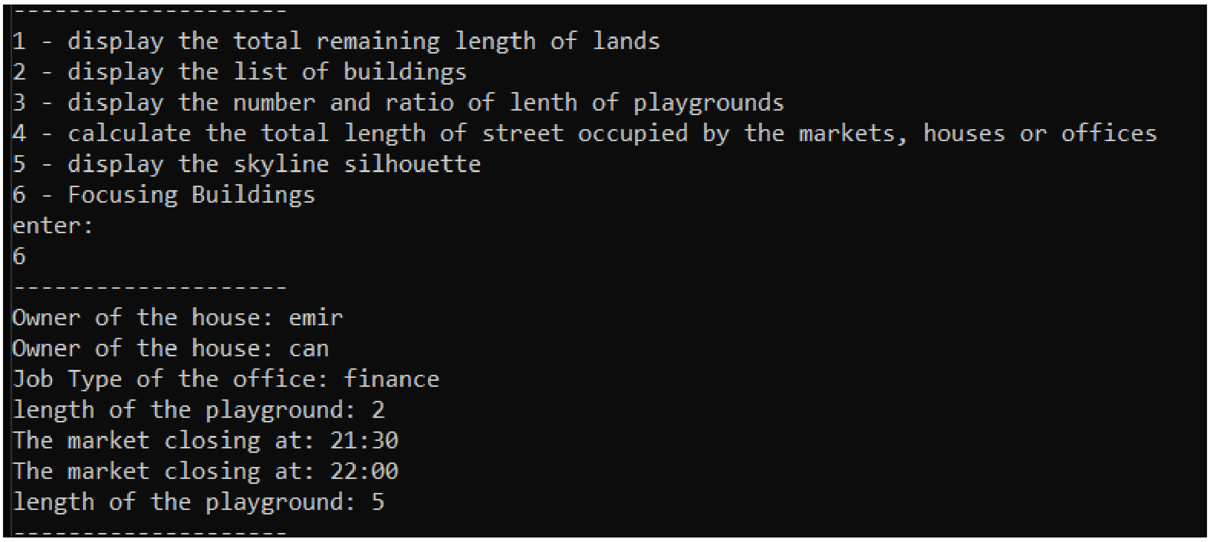
14.



```
--------------------
1 - display the total remaining length of lands
2 - display the list of buildings
3 - display the number and ratio of lenth of playgrounds
4 - calculate the total length of street occupied by the markets, houses or offices
5 - display the skyline silhouette
6 - Focusing Buildings
enter:
5
-------street silhouette-------

    _***_
    *   *
    *   *   _***_
    *   *   *   *
    *_**_   *   *
     *  *   * _*****__*****_
     *  *   * *            *
_*_  *  *   * *          _****_
```

15.



```
--------------------
1 - display the total remaining length of lands
2 - display the list of buildings
3 - display the number and ratio of lenth of playgrounds
4 - calculate the total length of street occupied by the markets, houses or offices
5 - display the skyline silhouette
6 - Focusing Buildings
enter:
6
--------------------
Owner of the house: emir
Owner of the house: can
Job Type of the office: finance
length of the playground: 2
The market closing at: 21:30
The market closing at: 22:00
length of the playground: 5
--------------------
```