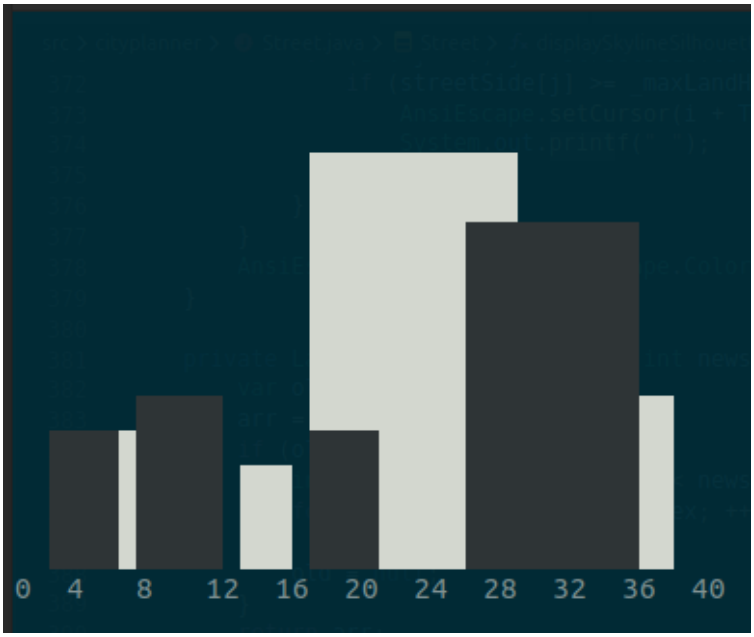
GIT Department of Computer Engineering
CSE 222/505 - Spring 2022
Homework 3 Report



Emirkan Burak Yılmaz
1901042659

1. SYSTEM REQUIREMENTS

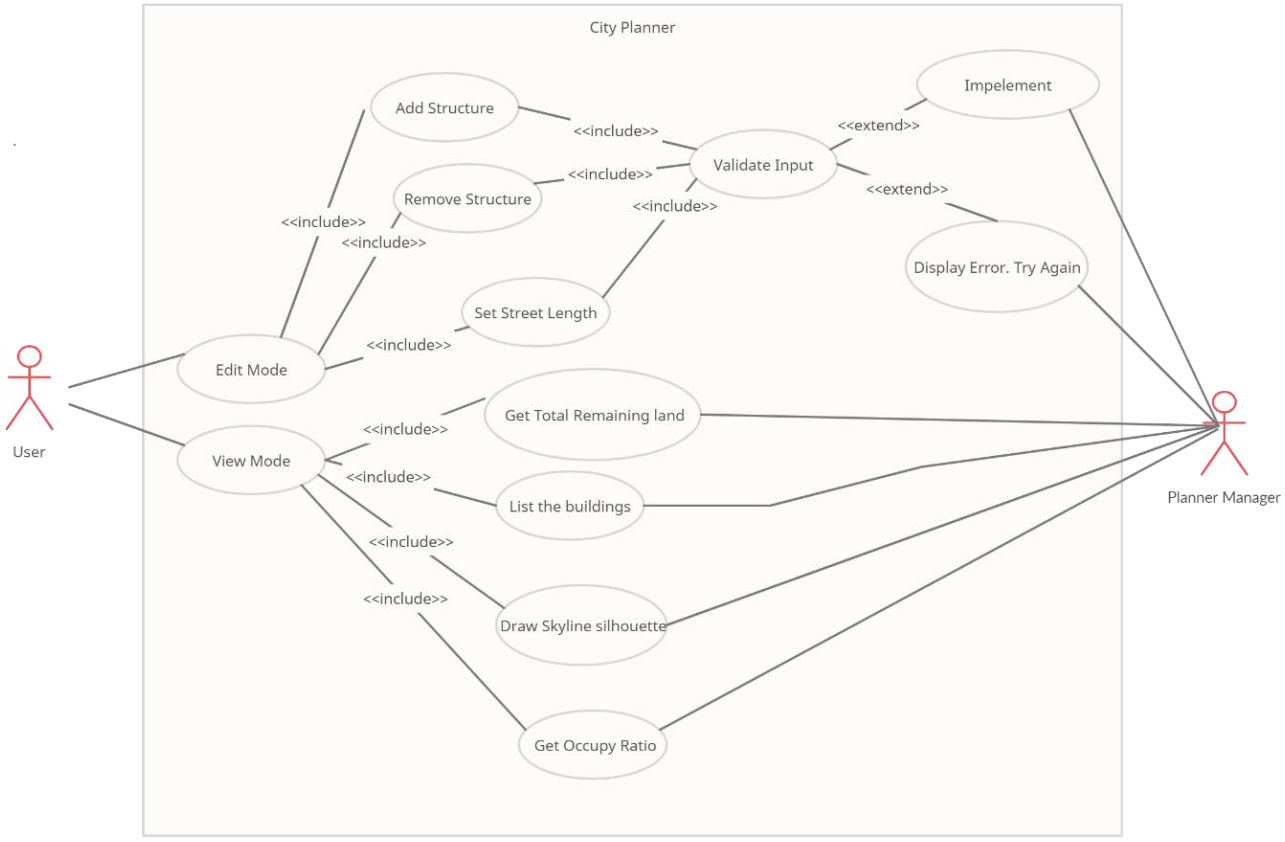
CityPlanner is a software that used for designing a city street. There are two mode which are editing and viewing.

In edit mode the user can set the length of the street and can add or remove structures (buildings or playgrounds) at both side of the street. Each structure has position, width, and height properties. In addition to these they also have own properties such as number of rooms for houses or opening/closing time for markets. The user can play these properties to design the street.

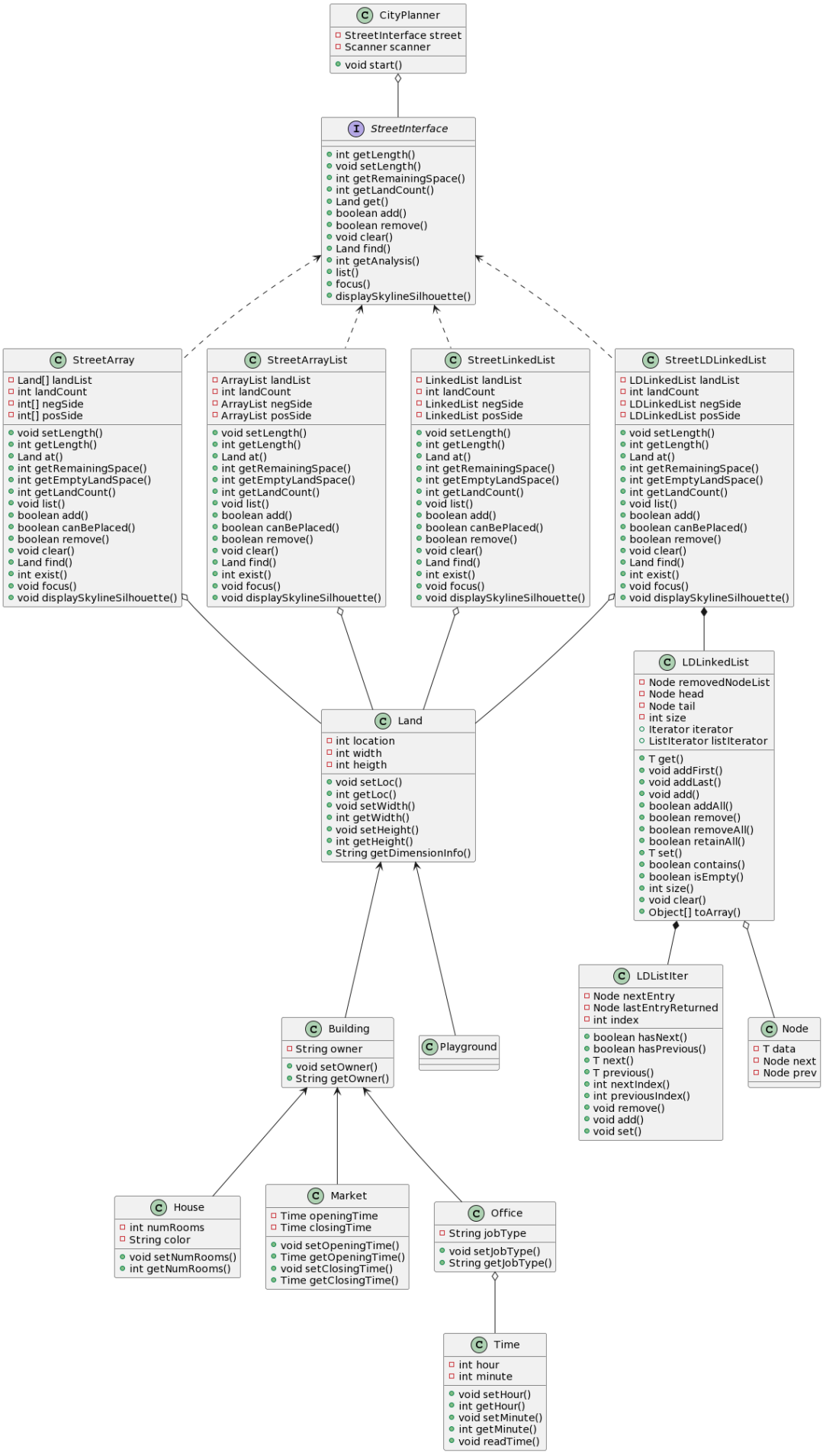
In view mode the user can see analysis of the street such as displaying number and occupy ratio of structures in the street. In addition to analysis information, software provides skyline silhouette of the street to show user what the street looks like.

In addition to user interface, CityPlanner implemented with four different data structure which are newly created data structures Array, LDLinkedList and the ones in java.util LinkedList, ArrayList. User can initialize CityPlanner with these four different structure by provided constructor. If not the default constructor initialized CityPlanner with Array data structure.

2. USE CASE AND CLASS DIAGRAMS



Use Case diagram



Class Diagram

3. PROBLEM SOLUTION APPROACH

My solution is based on OOP design. First, I defined **CityPlanner** class which will provide an interface to the user. A workplace for adding/removing structure I create **Street** class. A street has lands which are either empty or filled with any structure. Since every land has width, height, and location, I keep them in **Land** base class. And I defined empty land as any with, location but 0 height. So, street has lands which can be filled or empty. After that a land could be building or playground. So, I create two class **Building** and **Playground** which are extended from Land base class. The fundamental reason why Building, and Playground are separated is a building has its owner as opposite to playgrounds. In that point my class design is open for any extensions for future improvements. In summary Land class keeps all the structures (building, playground) and thanks to Land base class I can write generic methods that will work all the structures by power of polymorphism. For add some features and specifications I add subclasses **Market**, **House**, **Office** which are extended from Building class. Each subclass has its own properties such as opening/closing time for Market or number of rooms for House. In general, the required classes are defined as like that.

CityPlanner enables user to choose which data structure gone be used. To provide this feature I create the interface StreetInterface and implement four different implementations of it those are StreetArray, StreetLDLinkedList, StreetLinkedList, and StreetArrayList. CityPlanner has type StreetInterface field reference and this way CityPlanner can be used with four different Street implementation. With this way I prevent implementing CityPlanner four time for each Street data structure.

Class	**Land List Reference**	**Used Data Structure**
StreetArray	Land[]	Basic array
StreetLDLinkedList	LDLinkedList<Land>	LDLinkedList (Lazy Deletion Linked List)
StreetArrayList	ArrayList<Land>	Java ArrayList (java.util)
StreetLinkedList	LinkedList<Land>	Java LinkedList (java.util)

Street class has Land list to keep the lands which can be Market, House, Playground etc. To represent the side of the street I use my own coordinate system. One side starts with 0 increases to right till the length of the street, for other side again starts with 0 and decreases to right till negative of the length of the street. So location value is in range negative of street length to street length. With this design choice software provide unique location for two sides of the street. The user can jump between street sides just by changing the location as positive or negative. I also keep two integer list which sizes are same as street length. These are used as keep the height of the structure at that location which also same as array index. By doing that I can easily detect which places are empty (0 height) or filled. With this way I can easily prevent superposition during new structure addition. In addition to that these two arrays are used to draw skyline silhouette of the street. To draw a skyline silhouette, it's enough to know the height of the land at that location. Not because its required, just for make software much more interesting I use ANSI Escape Sequences which we learn during CSE241 to change the color of the font and setting the cursor location to provide good visual representation. To do that I implement **AnsiEscape** class which contains useful static methods.

4. TIME COMPLEXITY

StreetInterface Method	Class			
-	StreetArray	StreetArrayList	StreetLinkedList	StreetLDLinkedList
getLength	θ(1)	θ(1)	θ(1)	θ(1)
setLength	θ(1)	θ(1)	θ(1)	θ(1)
getRemainingSpace	θ(n)	θ(n)	θ(n)	θ(n)
getLandCount	θ(1)	θ(1)	θ(1)	θ(1)
get	θ(1)	θ(1)	O(n)	O(n)
add	θ(1)	θ(1)	O(n)	O(n)
remove	θ(1)	O(n)	θ(1)	θ(1)
clear	θ(n)	θ(n)	θ(n)	θ(n)
find	θ(n)	θ(n)	θ(n)	θ(n)
getAnalysis	θ(n)	θ(n)	θ(n)	θ(n)
list	θ(n)	θ(n)	θ(n)	θ(n)
focus	θ(1)	θ(1)	θ(1)	θ(1)
displaySkylineSilhouette	θ(n)	θ(n)	θ(n)	θ(n)

5. RUNNING TIME

StreetArray		
Method	**Problem Size**	
-	**10**	**100**
add	35.49	314.57
remove	21.09	521.18
getRemainingSpace	10.94	19.91
get	0.65	4.31
find	19.84	30.41
list	49981.17	111410.06
displaySkylineSilhouette	33109.64	179181.92
clear	156.90	931.98

StreetArrayList		
Method	**Problem Size**	
-	**10**	**100**
add	58.44	481.89
remove	89.21	5246.21
getRemainingSpace	32.47	141.40
get	0.42	2.51
find	13.26	77.17
list	25618.38	134498.98
displaySkylineSilhouette	5838.40	58092.07
clear	90.46	3182.68

StreetLinkedList		
Method	**Problem Size**	
-	**10**	**100**
add	128.94	1075.35
remove	88.48	1354.92
getRemainingSpace	18.36	102.22
get	1.68	2.88
find	14.44	22.25
list	5600.16	20072.50
displaySkylineSilhouette	10367.93	61602.59
clear	101.75	1560.96

StreetLDLinkedList		
Method	**Problem Size**	
-	**10**	**100**
add	97.64	1971.64
remove	92.24	1484.30
getRemainingSpace	30.80	172.21
get	3.74	7.10
find	25.47	31.42
list	3207.36	18921.53
displaySkylineSilhouette	2401.91	62476.80
clear	347.31	929.97

StreetArrayList

```
Method                          Problem Size          Run Time (msec)
add                                       10                   58.44
add                                      100                  481.89
remove                                    10                   89.21
remove                                   100                 5246.21
getRemainingSpace                         10                   32.47
getRemainingSpace                        100                  141.40
get                                       10                    2.51
get                                      100                    0.42
find                                      10                   13.26
find                                     100                   77.17
list                                      10                25618.38
list                                     100               134498.98
displaySkylineSilhouette                  10                 5838.40
displaySkylineSilhouette                 100                58092.07
clear                                     10                   90.46
clear                                    100                 3182.68
Enter to continue
```

StreetArray

```
Method                          Problem Size          Run Time (msec)
add                                       10                   35.49
add                                      100                  314.57
remove                                    10                   21.09
remove                                   100                  521.18
getRemainingSpace                         10                   10.94
getRemainingSpace                        100                   19.91
get                                       10                    4.31
get                                      100                    0.65
find                                      10                   19.84
find                                     100                   30.41
list                                      10                49981.17
list                                     100               111410.06
displaySkylineSilhouette                  10                33109.64
displaySkylineSilhouette                 100               179181.92
clear                                     10                  156.90
clear                                    100                  931.98
```

StreetLDLinkedList

```
Method                          Problem Size          Run Time (msec)
add                                       10                   97.64
add                                      100                 1971.64
remove                                    10                   92.24
remove                                   100                 1484.30
getRemainingSpace                         10                   30.80
getRemainingSpace                        100                  173.21
get                                       10                    7.10
get                                      100                    3.74
find                                      10                   25.47
find                                     100                   31.42
list                                      10                 3207.36
list                                     100                18921.53
displaySkylineSilhouette                  10                 2401.91
displaySkylineSilhouette                 100                62476.80
clear                                     10                  347.31
clear                                    100                  929.97
Enter to continue
```

StreetLinkedList

```
Method                          Problem Size          Run Time (msec)
add                                       10                  128.94
add                                      100                 1075.35
remove                                    10                   88.48
remove                                   100                 1354.92
getRemainingSpace                         10                   18.36
getRemainingSpace                        100                  102.22
get                                       10                    2.88
get                                      100                    1.68
find                                      10                   14.44
find                                     100                   22.25
list                                      10                 5600.16
list                                     100                20072.50
displaySkylineSilhouette                  10                10367.93
displaySkylineSilhouette                 100                61602.59
clear                                     10                  101.75
clear                                    100                 1560.96
Enter to continue
```

6. TEST CASES

1. **Test Case:** Add & Remove new entry from/to LDLinkedList.

```
public static void test1() {
    String[] ingredientList = {
        "White meat", "Egg", "Yoghurt", "Banana", "Turkey", "Oat", "Salmon",
        "Honey", "Vinegar", "Rice", "Peanut butter", "Nuts", "Bitter chocolate"
    };

    LDLinkedList<String> list = new LDLinkedList<>();

    for (int i = 0; i < ingredientList.length; ++i)
        list.add(ingredientList[i]);

    System.out.printf("List size: %d\n", list.size());

    list.remove("Yoghurt");
    list.remove("Rice");
    list.remove("Hamburger");   // not in list
    list.remove("Nutella");     // not in list
    list.remove("Pizza");       // not in list

    Object[] mealList = list.toArray();
    System.out.println("Meal list: ");
    for (int i = 0; i < mealList.length; ++i)
        System.out.print(mealList[i] + ((list.size() != i + 1) ? ", " : "\n"));
    System.out.printf("List size: %d\n", list.size());

    list.clear();
    System.out.printf("List size: %d\n", list.size());
}
```

2. **Test Case:** Add entry both head and tail of the LDLinkedList.

```
public static void test2() {
    LDLinkedList<Integer> list = new LDLinkedList<>();
    list.add(10);       // list: {10}
    list.addFirst(5);   // list: {5, 10}
    list.add(20);       // list: {5, 10, 20}
    list.addLast(25);   // list: {5, 10, 20, 25}
    list.add(2, 15);    // list: {5, 10, 15, 20, 25}

    System.out.println(list);
    System.out.printf("List size: %d\n", list.size());

    int e1 = 13, e2 = 15, e3 = 20;
    System.out.printf("List has entry %d: %s\n", e1, list.contains(e1));
    System.out.printf("List has entry %d: %s\n", e2, list.contains(e2));
    // list.remove(3); // remove 4th entry which is 20
    System.out.printf("List has entry %d: %s\n", e3, list.contains(e3));

    for (var e : list)
        list.remove(e);
    System.out.printf("List size: %d\n", list.size());
}
```

3. **Test Case:** Add/Remove/Retain all the entries inside of the given Collection from LDLinkedList.

```
    Collection<String> c1 = new java.util.ArrayList<>();
    c1.add("Maria");
    c1.add("Bruce");
    c1.add("Proteus");
    c1.add("Sinbat");
    c1.add("Jordan");
    c1.add("Carl");

    Collection<String> c2 = new java.util.LinkedList<>();
    c2.add("Jordan");
    c2.add("Bruce");
    c2.add("Montag");
    c2.add("Hary");

    Collection<String> c3 = new java.util.Vector<>();
    c3.add("Sinbat");
    c3.add("Maria");
    c3.add("Bruce");
    c3.add("Proteus");

    LDLinkedList<String> list = new LDLinkedList<>();
    list.addAll(c1);
    System.out.println(list);
    list.removeAll(c2);
    System.out.println(list);
    list.retainAll(c3);
    System.out.println(list);
}
```

4. **Test Case:** Try to add a new structure (Market, Office, Playground) to the empty part of the street.

```
public static void test1(StreetInterface street) {
    System.err.printf("New street which length is %d\n", street.getLength());

    // create a house at location 5 whose width is 4 height is 7
    House house = new House(5, 4, 7, "Alice", 5, "white");

    // create an office at location -10 whose width is 5 height is 5
    Office office1 = new Office(-10, 5, 5, "Bruce", "Information Technology");

    // create a market at location 30 whose width is 9 height is 5
    Time openingTime = new Time(9, 0);
    Time closingTime = new Time(22, 55);
    Market market = new Market(12, 5, 5, "Justin", openingTime, closingTime);

    // create a playground at location -6 whose width is 4
    Playground playground = new Playground(-6, 4);

    // create an office at location -30 whose width is 10 height is 12
    Office office2 = new Office(-30, 10, 12, "IstM", "Architect");

    // these strucutes are not cause any superposition
    // so expected result is all of them are added succesfully
    debug_add(street, house);
    debug_add(street, office1);
    debug_add(street, playground);
    debug_add(street, market);
    debug_add(street, office2);

    System.out.printf("Number of structure in street: %d\n\n", street.getLandCount());
```

5. **Test Case:** Try to add new structure to the filled part of street.

```
// try to add any structure that cause superposition
// superposition with Alice's house
debug_add(street, new Office(17, 6, 4, "Harley", "Architecture"));
// superposition with Bruce's office
debug_add(street, new Market(6, 5, 4, "KIM"));

System.out.printf("Number of structure in street: %d\n", street.getLandNumber());
```

6. **Test Case:** Remove a land which street contain.

```
// remove the first structure of list
debug_remove(street, 0);
```

7. **Test Case:** Remove a land which street does not contain.

```
// try to remove structure that doesn't exist in the street
debug_remove(street, new Office(13, 4, 5, "Joker"));
debug_remove(street, street.getLandNumber());
```

8. **Test Case:** Check if there is a land at given location.

```
// try to find Bruce's office
loc = -12;
var l = street.find(loc);
if (l == null)
    System.out.printf("No structure exist at location %d", loc);
else {
    System.out.printf("There exist a structure at location %d\nFocus information\n", loc);
    street.focus(loc);
}
System.out.println();
debug_remove(street, l);
```

9. **Test Case:** List all the structures in the street.

```
// display all the added structures
street.listAllStructures();
```

10. **Test Case:** Display skyline silhouette of the street.

```
// display all the added structures
street.displaySkylineSilhouette();
```

11. **Test Case:** Clear the street by removing all the lands inside of it.

```
// clear the street
street.clear();
System.out.printf("\nNumber of structure in street: %d\n", street.getLandNumber());
```

12. **Test Case:** Check CityPlanner Editing mode
13. **Test Case:** Check CityPlanner Viewer mode
14. **Test Case:** Give bad inputs during CityPlanner user interface

7. RUNNING AND RESULTS

1. Run Time Result:

```
List size: 13
[Ljava.lang.Object;@5451c3a8
Meal list:
White meat, Egg, Banana, Turkey, Oat, Salmon, Honey, Vinegar, Peanut butter,
 Nuts, Bitter chocolate
List size: 11
List size: 0
```

2. Run Time Result:

```
{5, 10, 15, 20, 25}
List size: 5
List has entry 13: false
List has entry 15: true
List has entry 20: true
List size: 0
```

3. Run Time Result:

```
{Maria, Bruce, Proteus, Sinbat, Jordan, Carl}
{Maria, Proteus, Sinbat, Carl}
{Maria, Proteus, Sinbat}
```

4. Run Time Result:

```
New street which length is 40
(loc:  +5       w:   4  h:   7)       add(): SUCCESS
(loc: -10       w:   5  h:   5)       add(): SUCCESS
(loc:  -6       w:   4  h:   2)       add(): SUCCESS
(loc: +12       w:   5  h:   5)       add(): SUCCESS
(loc: -30       w:  10  h:  12)       add(): SUCCESS
Number of structure in street: 5
```

5. Run Time Result:

```
(loc: +40       w:   6  h:   4)       add(): FAIL
(loc: +17       w:   6  h:   4)       add(): FAIL
(loc:  +6       w:   5  h:   4)       add(): FAIL
Number of structure in street: 5
```

6. Run Time Result:

```
(loc:  +5       w:   4  h:   7)       remove(): SUCCESS
```

7. **Run Time Result:**

```
(loc: +13       w:    4  h:    5)       remove(): FAIL (No such element)
remove(): FAIL (Index out of bounds (max index: 3))
```

8. **Run Time Result:**

```
There exist a structure at location -12
Focus information
Office job Type: Information Technology
```

9. **Run Time Result:**

```
   Structure Type       Location   Width      Height

1  House                +5         4          4
2  Office               -10        5          5
3  Playground           -6         4          4
4  Market               +12        5          5
5  Office               -30        10         10
```

10. **Run Time Result:**

11. **Run Time Result:**

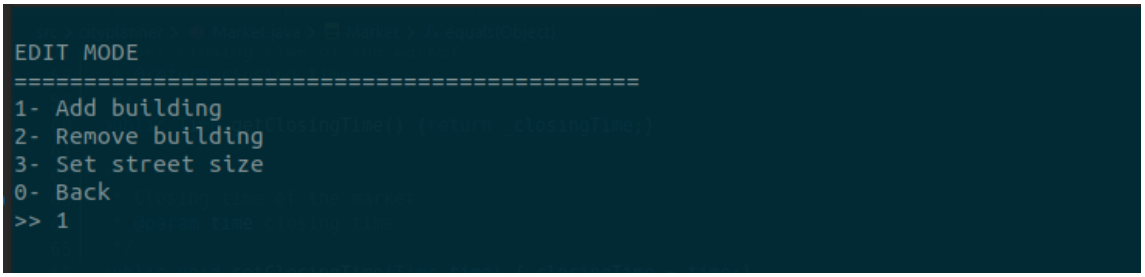
```
Number of structure in street: 0
```

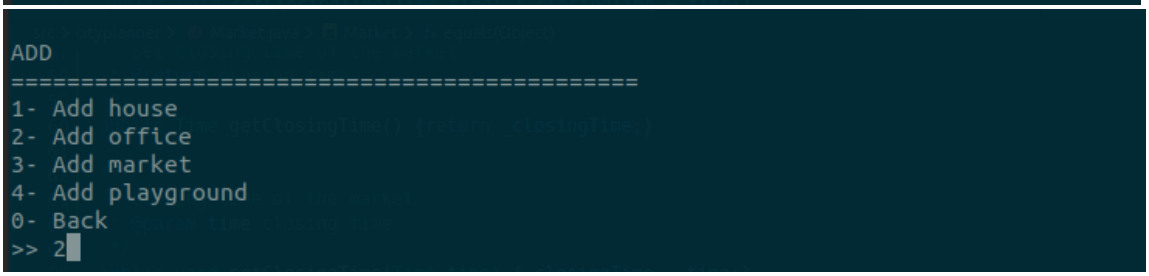
12. **Run Time Result:**

Select Edit mode from main menu.

```
CityPlanner
==========================================
1- Edit Mode
2- View Mode
0- Exit
>> 1
```

Add a building.



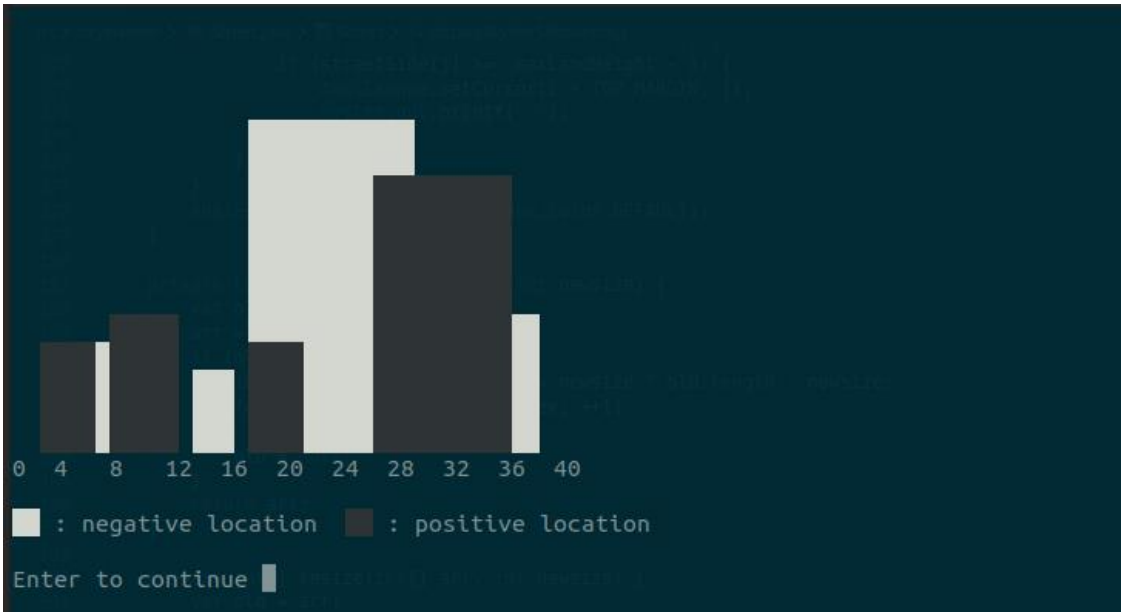


Enter properties of house.

Select an building to remove.

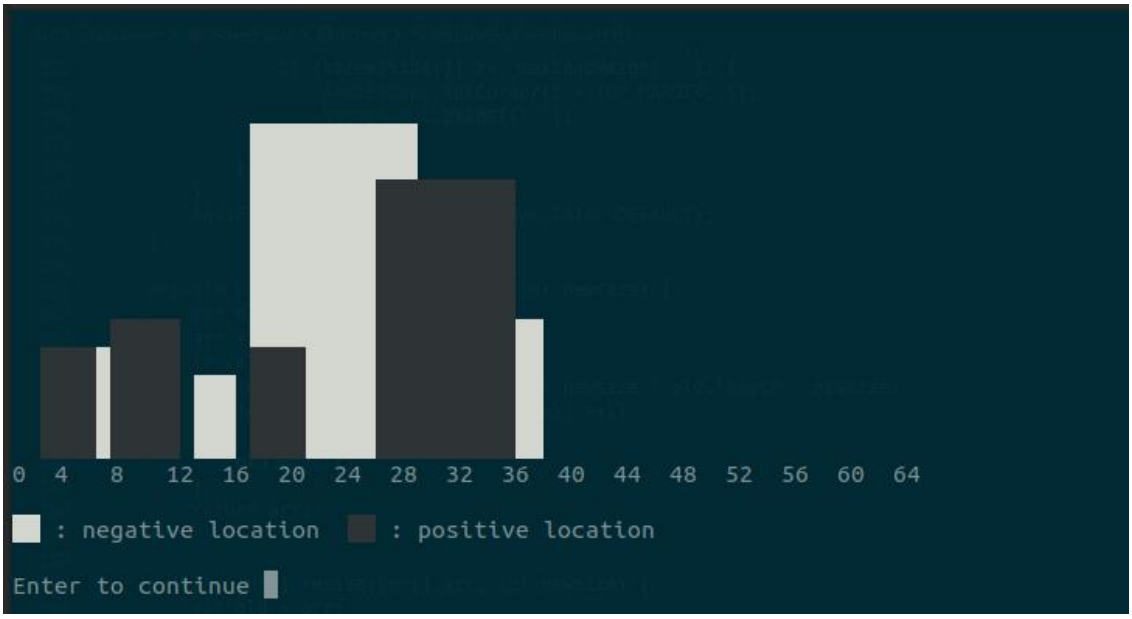
```
EDIT MODE
=============================================
1- Add building
2- Remove building
3- Set street size
0- Back
>> 2

   Structure Type       Location   Width      Height

1  House                +5         4          4
2  Market               -10        5          5
3  Playground           +20        4          4
4  Office               +10        5          5
5  Playground           -6         4          4
6  Office               -24        10         10
>>  4

Removed succesfully. Enter to continue
```

Change the size of the street. Here's the initial view of the street.



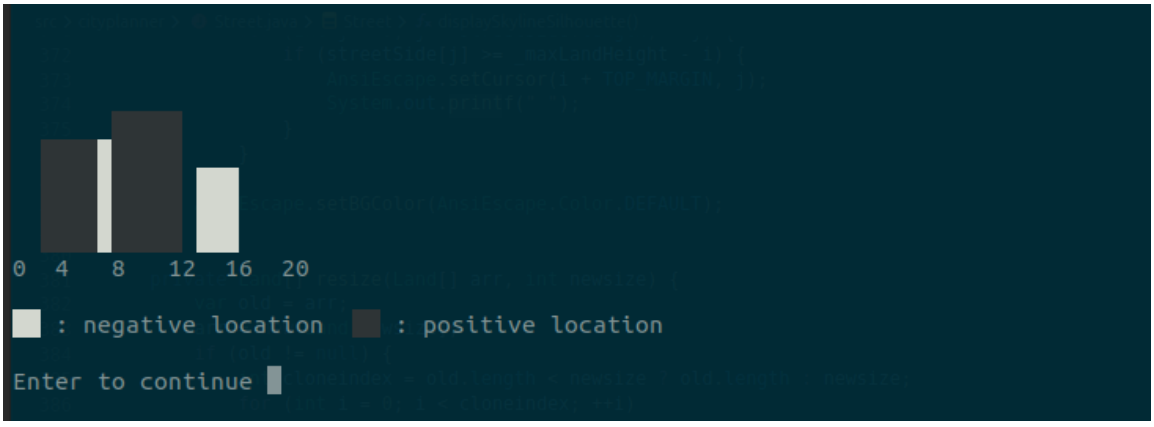

```
   Structure Type       Location   Width      Height

1  House                +5         4          4
2  Market               -10        5          5
3  Playground           +20        4          4
4  Office               +10        5          5
5  Playground           -6         4          4
6  House                -24        12         12
7  Playground           +32        10         10
8  House                -36        5          5
9  House                -15        3          3

Detailed View
=============================================
1- Focus
0- Back
>>
```

```
EDIT MODE
=============================================
1- Add building
2- Remove building
3- Set street size
0- Back
>> 3
Street Length: 64
New street length is 64
Enter to continue
```



As it can see no data lost after increasing the lenght of the street. Lents decrease the length.

```
EDIT MODE
=============================================
1- Add building
2- Remove building
3- Set street size
0- Back
>> 3
Street Length: 20
New street length is 20
Enter to continue
```



After decreasing the length of the street, program keeps only the structures that are located at new street bound.

13. Run Time Result:

```
CityPlanner
============================================
1- Edit Mode
2- View Mode
0- Exit
>> 2
```

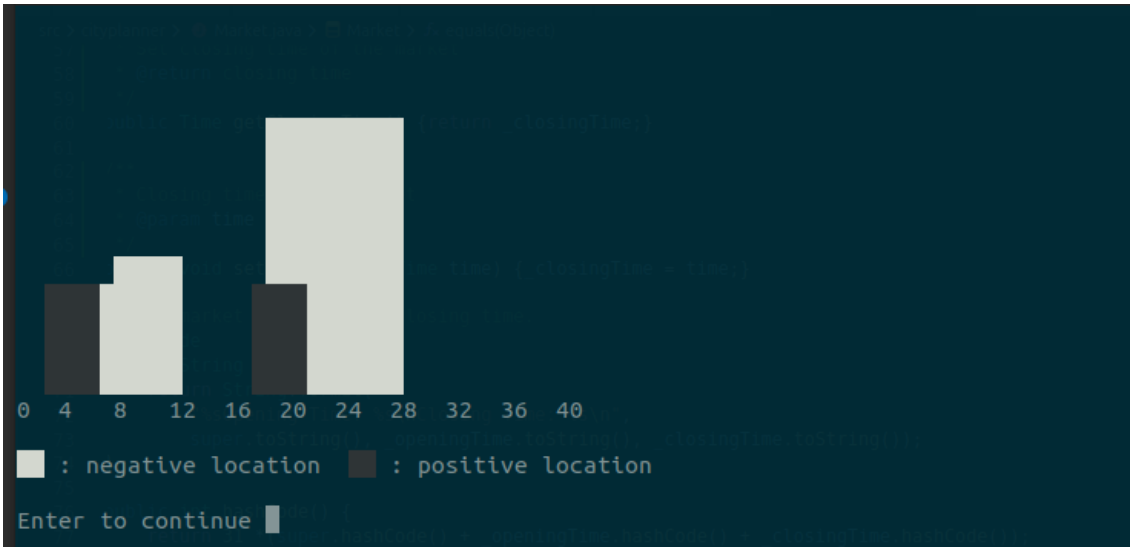
```
VIEW MODE
============================================
1- Total remaining length of lands on the street
2- List of buildings on the street
3- Skyline silhouette of the street
4- Total length of street occupied by the structures
0- Back
>> 1
Remaining length of lands: 53
Enter to continue
```

List all the buildings on the street. And select one of them to see specific information about it by focus function.

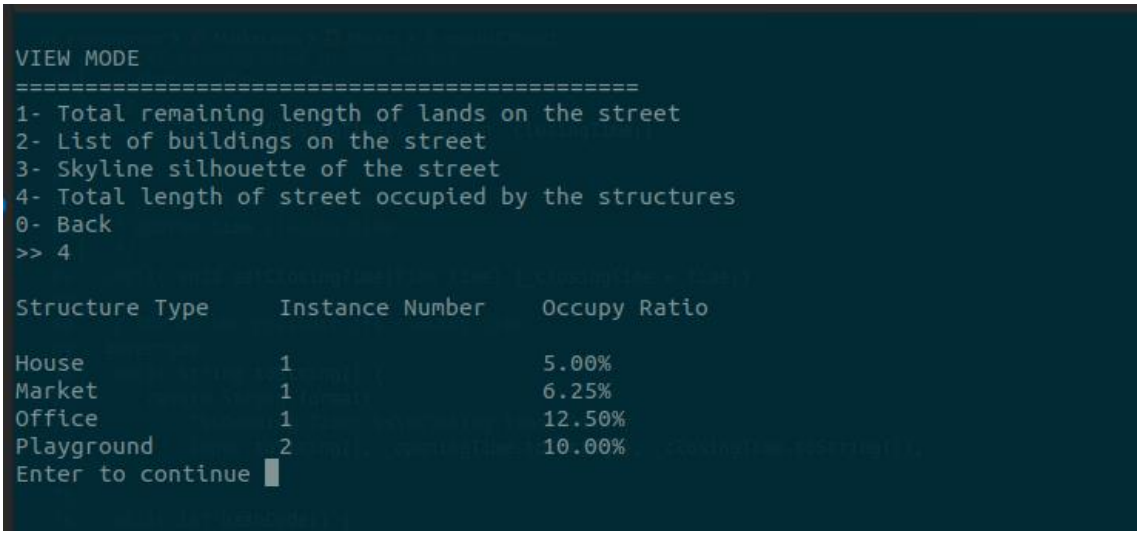
```
   Structure Type       Location   Width      Height

1  House                +5         4          4
2  Market               -10        5          5
3  Playground           +20        4          4
4  Office               -24        10         10
5  Playground           -6         4          4

Detailed View
============================================
1- Focus
0- Back
>>
```

```
   Structure Type       Location   Width      Height

1  House                +5         4          4
2  Market               -10        5          5
3  Playground           +20        4          4
4  Office               -24        10         10
5  Playground           -6         4          4

Detailed View
============================================
1- Focus
0- Back
>> 1
Structure: 1
House owner: Alice
Enter to continue
```

Display Skyline silhouette of the street



Display street analysis



14. Run Time Result:

Enter negative value for street lenght.

Enter stupid values, later enter 0 to turn back main menu.

```
EDIT MODE
============================================
1- Add building
2- Remove building
3- Set street size
0- Back
>> 2

   Structure Type       Location   Width      Height

1  House                +5         4          4
2  Market               -10        5          5
3  Playground           +20        4          4
4  Office               +10        5          5
5  Playground           -6         4          4
>>  999999
>>  fşsadnfsadof
Please make a proper choose
Enter to continue
>>  1lkd
Please make a proper choose
Enter to continue
>>  0
```

Try to add a structure to already filled place.

```
0  4   8   12  16  20  24  28  32  36  40

   : negative location    : positive location

Location: lafmdsaifl lfmdslkf 12*42
Please enter proper values
Location: 22
Width: 10
Height: 15
Owner: BIM
Opening time (hh : mm): 9:35
Closing time (hh : mm): 22:55
```

```
0  4   8   12  16  20  24  28  32  36  40

   : negative location    : positive location

Cannot add. Enter to continue
```

The End