

GTU Department of Computer Engineering
CSE 222/505 - Spring 2022
Homework #03 Report

Ali Kaya

1901042618

a.kaya2019@gtu.edu.tr

1.Detailed System Requirements

To use the program, the user must enter the number that will determine the length of the **Street**. After that the user need to give some decisions about how the program will work : Editing mode, viewing mode, adding new element vs.

For example:

```
Please enter street length:
70
1->Editing Mode
2->Viewing Mode
3->Exit
Please select program mode:
1
1->Adding new element
2->Deleting one element from last
3->Go previous menu
Please select :
1
1->Adding a House
2->Adding a Market
3->Adding a Office
4->Adding a Playground
5->Go Previous Menu
Please select :
1
Please enter side number 0 for back 1 for front:
0
Please enter position number:
0
Please enter Height number:
10
Please enter Length number:
10
Please enter Room number:
3
Please enter owner of the house:
Ali
Please enter color of the house:
Purple
Adding operation done successfully
```

```
buildTypes house1 = new house( pos: 0 , hei: 10 , len: 10 , rooms: 3 , owner: "Ali" , color: "Purple");
```

In here program create an house reference with these properties.

After these Program add house1 reference to the array (hw1 version);

```
private buildTypes[][] arr;
```

Program add house1 reference to the ArrayList<buildTypes> (ArrayList version);

```
private ArrayList< ArrayList<buildTypes> >arr;
```

Program add house reference to the LinkedList<buildTypes> (LinkedList version);
(In here i have to open two references because LinkedList doesn't support multiple dimension)

```
private LinkedList<buildTypes> listBack;
private LinkedList<buildTypes> listFront;
```

Program add house1 reference to the LinkedList<buildTypes>(LinkedList version);

(In here i have to open two references because LDLinkedList doesn't support multiple dimension)

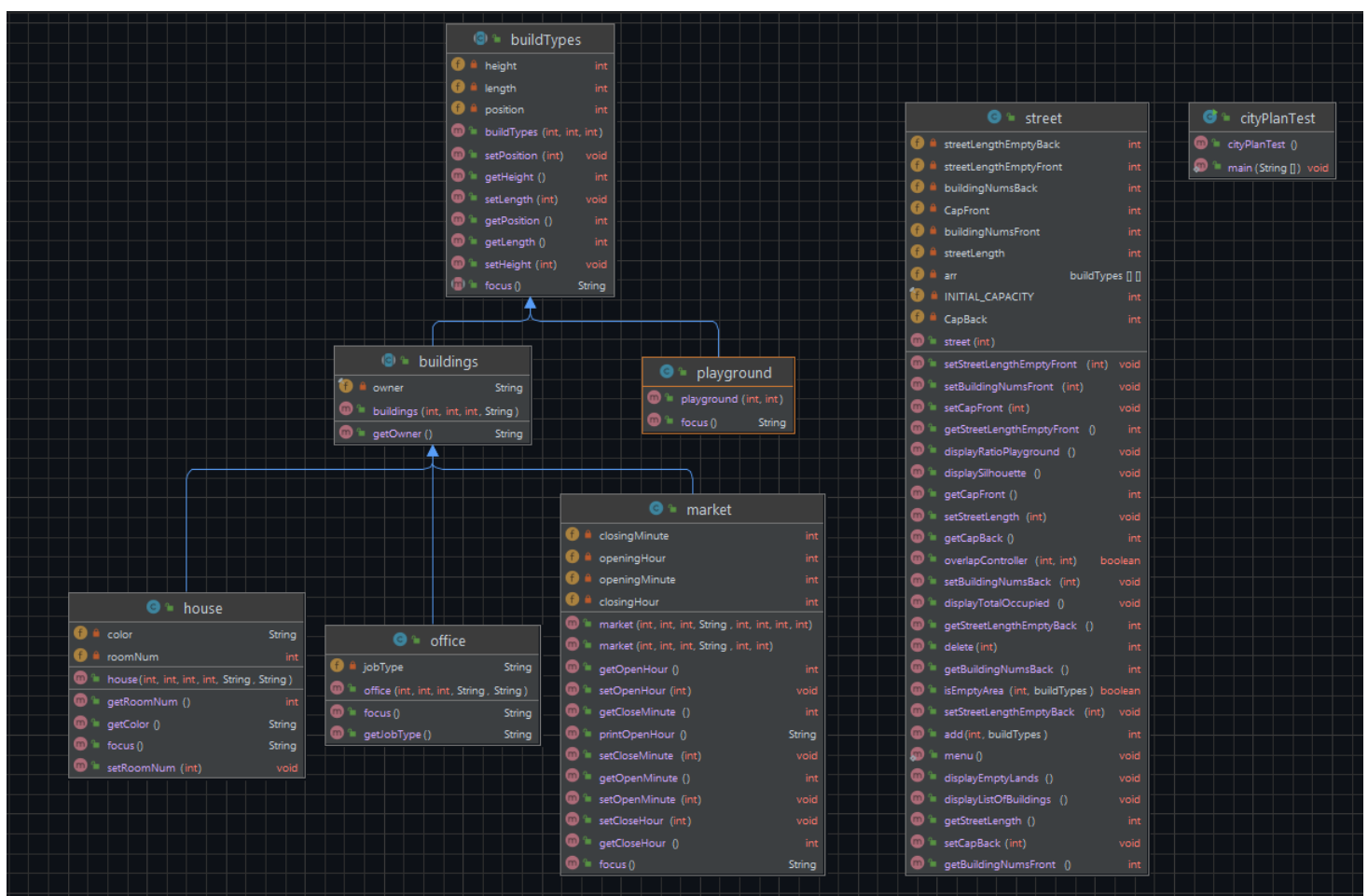
```
private LDLinkedList<buildTypes> listBack;
private LDLinkedList<buildTypes> listFront;
```

```
mainStreet.add( borf: 1, house1);
```

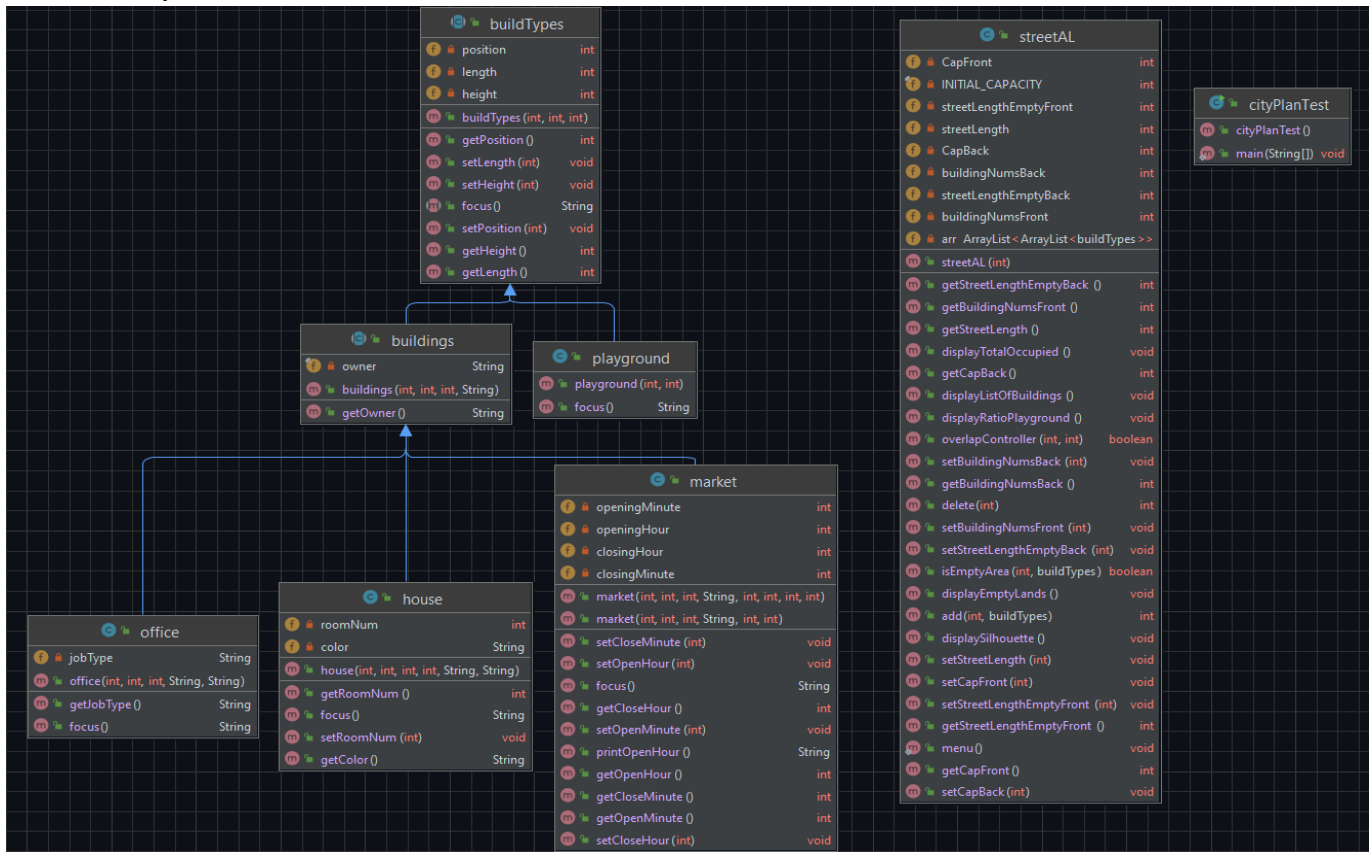
borf parameter indicates side of the Street
(0->back side , 1->front side)

2. Class Diagrams

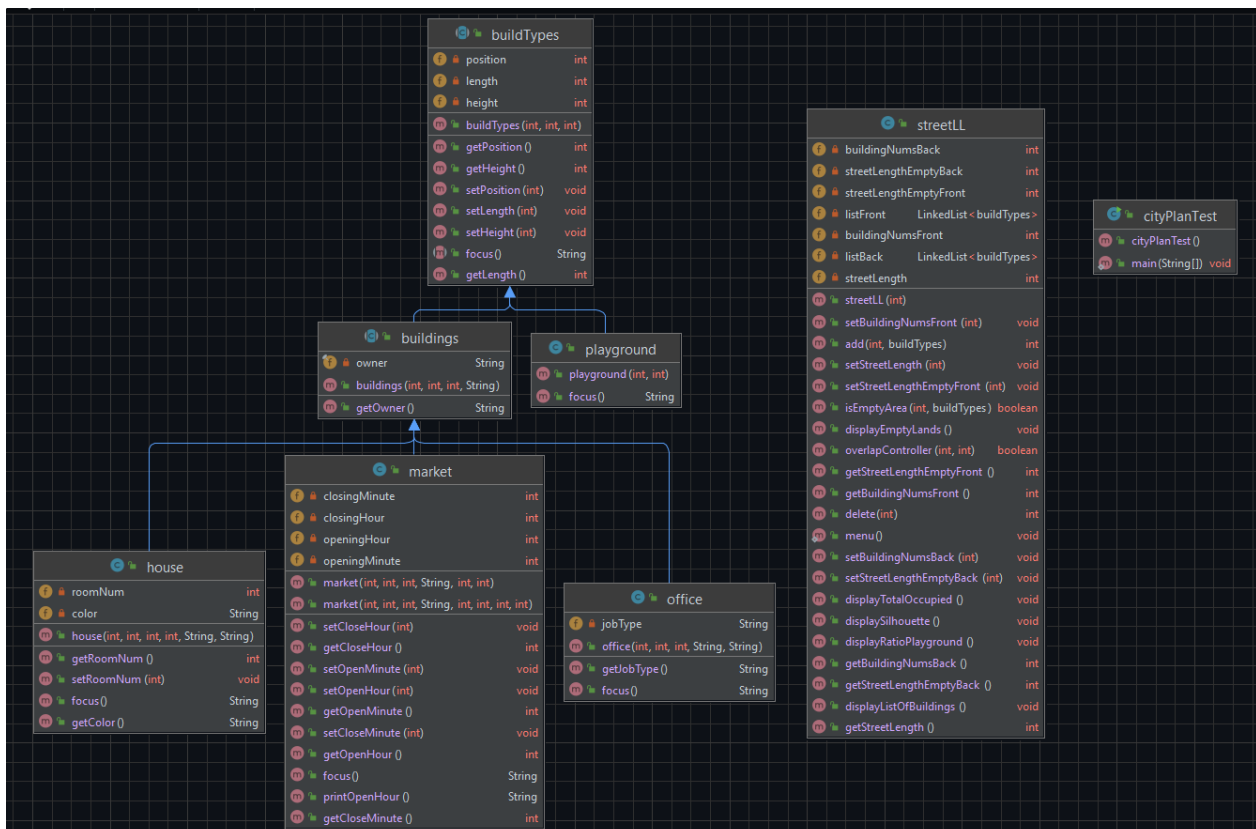
HW1 version;



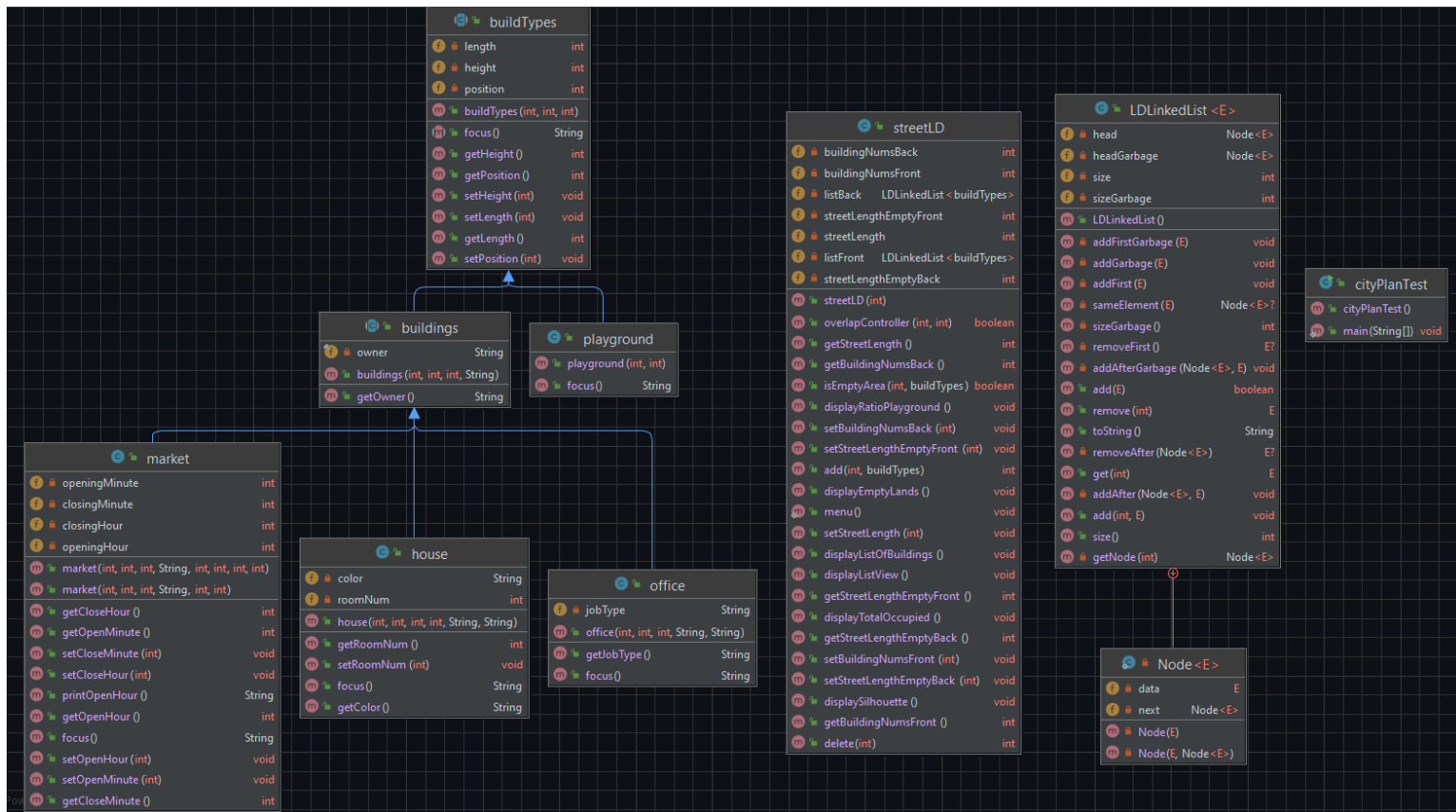
ArrayList<E> version;



LinkedList<E> version;



LDLinkedList<E> version;



3. Problem Solution approach

To hold the data properly i need an array , ArrayList<E> , LinkedList<E> , LDLinkedList<E> but what type it should be and what size should it be?

First houses , markets and offices are building type in real life and playgrounds stay out of these because of this reason i need another superclass which is also superclass of buildings class and playground class and its name is buildTypes. house, market and office classes extends buildings class and buildings class extends buildType class. Also playground class extends buildTypes class too.

To hold all of these types my data structures's types need to be buildTypes type

```
/**Keeps buildTypes references*/
private buildTypes[][] arr;
```

```
private ArrayList< ArrayList<buildTypes> >arr;
```

```
private LinkedList<buildTypes> listBack;
private LinkedList<buildTypes> listFront;
```

```
private LDLinkedList<buildTypes> listBack;
private LDLinkedList<buildTypes> listFront;
```

Then my array need to have 2 rows ([0] for back side of the Street [1] for front side of the street)(hw1 version)

```
arr = new buildTypes[2][];  
arr[0] = new buildTypes[getCapBack()];  
arr[1] = new buildTypes[getCapFront()];
```

To make 2 dimensional ArrayList<E> i wrote the following codes in streetAL constructor;

```
arr = new ArrayList<>(initialCapacity: 2);  
ArrayList< buildTypes > arr0 = new ArrayList<>(getCapBack());  
ArrayList< buildTypes > arr1 = new ArrayList<>(getCapFront());  
arr.add(arr0);  
arr.add(arr1);
```

But LinkedList<E> and LDLinkedList<E> structures doesn't support multiple dimension.

I kept this data structures inside a data holder class which is **Street** for easier working

I wrote my methods in **Street** class.

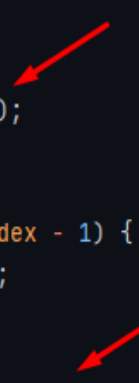
```
public int add(int borf, buildTypes newBuild) {  
    int ans = 0;
```

```
public int delete(int borf) {
```

To write lazy deletion strategy in my LDLinkedList<E> implementation i wrote following codes;

```
@Override
public E remove(int index) {
    int counter = 0;
    Node<E> temp = head;
    E rtr = null;

    if (index < size()) {
        if (index == 0) {
            addGarbage(temp.data);
            rtr = removeFirst();
        } else {
            while (counter != index - 1) {
                temp = temp.next;
                counter++;
            }
            addGarbage(temp.next.data);
            rtr = removeAfter(temp);
        }
    }
    return rtr;
}
```



```
private void addGarbage(E element)
{
    int index = sizeGarbage();

    if (index == 0) {
        addFirstGarbage(element);
    }
    else {
        Node<E> node = getNode(index: index-1);
        addAfterGarbage(node, element);
    }
}
```

When an element is wanted to be deleted from the list, I added this element to the garbage list ;

```
private void addFirstGarbage(E element) {
    headGarbage = new Node<E>(element, headGarbage);
    sizeGarbage++;
}
```

```
private void addAfterGarbage(Node<E> node, E element) {
    node.next = new Node<E>(element, node.next);
    sizeGarbage++;
}
```

If the same element is wanted to be added to the list again later, I add this element to the main list from the garbage list.

```
private void addFirst(E element) {
    Node<E> temp = sameElement(element);
    if (temp != null)
    {
        temp.next = head;
        head = temp;
    }
    else
        head = new Node<E>(element, head);

    size++;
}
```

```
private void addAfter(Node<E> node, E element) {
    Node<E> temp = sameElement(element);
    if (temp != null)
    {
        temp.next = node.next;
        node.next = temp;
    }
    else
        node.next = new Node<E>(element, node.next);

    size++;
}
```

To do that we have to control the elements in garbage list is fits our wants or not, sameElement(E element) method do this job very well;

```
private Node<E> sameElement(E element) {
    int ans = 0;
    Node<E> temp = headGarbage;

    while (temp != null) {
        if (element.getClass().getSimpleName().equals(temp.data.getClass().getSimpleName())) {
            if (element instanceof house)
            {
                && ((house) element).getOwner().equals(((house) temp.data).getOwner())
                && ((house) element).getColor().equals(((house) temp.data).getColor())
                && ((house) element).getRoomNum() == ((house) temp.data).getRoomNum()
                && ((house) element).getPosition() == ((house) temp.data).getPosition()
                && ((house) element).getHeight() == ((house) temp.data).getHeight()
                && ((house) element).getLength() == ((house) temp.data).getLength() {
                    ans = 1;
                    break;
                }
            }
            else if (element instanceof market)
            {
                && ((market) element).getOwner().equals(((market) temp.data).getOwner())
                && ((market) element).getOpenHour() == ((market) temp.data).getOpenHour()
                && ((market) element).getOpenMinute() == ((market) temp.data).getOpenMinute()
                && ((market) element).getCloseHour() == ((market) temp.data).getCloseHour()
                && ((market) element).getCloseMinute() == ((market) temp.data).getCloseMinute()
                && ((market) element).getPosition() == ((market) temp.data).getPosition()
                && ((market) element).getHeight() == ((market) temp.data).getHeight()
                && ((market) element).getLength() == ((market) temp.data).getLength() {
                    ans = 1;
                    break;
                }
            }
            else if (element instanceof office)
            {
                && ((office) element).getOwner().equals(((office) temp.data).getOwner())
                && ((office) element).getJobType().equals(((office) temp.data).getJobType())
                && ((office) element).getPosition() == ((office) temp.data).getPosition()
                && ((office) element).getHeight() == ((office) temp.data).getHeight()
                && ((office) element).getLength() == ((office) temp.data).getLength() {
                    ans = 1;
                    break;
                }
            }
            else if (element instanceof playground)
            {
                && ((playground) element).getPosition() == ((playground) temp.data).getPosition()
                && ((playground) element).getHeight() == ((playground) temp.data).getHeight()
                && ((playground) element).getLength() == ((playground) temp.data).getLength() {
                    ans = 1;
                    break;
                }
            }
        }
        temp = temp.next;
    }

    if (ans == 1) return temp;
    else return null;
}
```


5. Test Cases

Create a Street which have 50 length;

```
street mainStreet = new street( newLength: 50);
```

Open an house reference with these properties;

```
System.out.println("Let's open a house reference which have following properties ; ");
System.out.println("Let its name be house1 , its Position be 0 , its Height be 10 , its Length be 10 ,\n" +
    " its Room number be 3 , its Owner be Ali and its color be Purple\n");
buildTypes house1 = new house( pos: 0 , hei: 10 , len: 10 , rooms: 3 , owner: "Ali" , color: "Purple");
```

Open another house reference but its length value is longer than Street length

```
buildTypes test123 = new house( pos: 30 , hei: 10 , len: 125 , rooms: 3 , owner: "test" , color: "test");
```

```
System.out.println("First ,Let's try to call delete() method to see the error message when street has no building :");
mainStreet.delete( borf: 1);
```

```
System.out.println("Secondly ,Let's try to add house that is longer than the length of the street \n" +
    "with add() method to see the error message: ");
mainStreet.add( borf: 1, test123);
```

```
First ,Let's try to call delete() method to see the error message when street has no building :
Deleting operation failed there is no building on this side!!
```

```
Secondly ,Let's try to add house that is longer than the length of the street
with add() method to see the error message:
The empty area of the front side of the street is not enough for this action!!
Adding operation failed !!
```

Analyzing time complexity of four versions;

a) Hw1 version;

Theoretical running time: $O(n)$

Experimental: $O(n)$

Running time for a street with 150 buildings: 0.0600749 sec

Running time for a street with 1050 buildings: 0.2039011 sec

Running time for a street with 1550 buildings: 0.2683359 sec

Running time for a street with 10050 buildings: 1.306699 sec

b) ArrayList<E> version;

Theoretical running time: $O(n)$

Experimental: $O(n)$

Running time for a street with 150 buildings: 0.0651533 sec

Running time for a street with 1050 buildings: 0.2510022 sec

Running time for a street with 1550 buildings: 0.3152348 sec

Running time for a street with 10050 buildings: 1.4349327s sec

c) LinkedList<E> version;

Theoretical running time $O(n^2)$

Experimental: $O(n^2)$

Running time for a street with 150 buildings: 0.0795932 sec

Running time for a street with 300 buildings: 0.1782959 sec

Running time for a street with 600 buildings: 0.5886202 sec

Running time for a street with 1800 buildings: 14.0563845 sec

Running time for a street with 3600 buildings: 116.9775756 sec

d) LDLinkedList<E> version;

Theoretical running time: $O(n^2)$

Experimental: $O(n^2)$

Running time for a street with 150 buildings: 0.1000721sec

Running time for a street with 300 buildings: 0.1927635 sec

Running time for a street with 600 buildings: 0.8402017 sec

Running time for a street with 1800 buildings: 33.91336 sec

```

20
Please enter owner of the market:
gdf
Please enter opening hour:
543
INVALID
1->Editing Mode
2->Viewing Mode
3->Exit
Please select program mode:
1
1->Adding new element
2->Deleting one element from last
3->Go previous menu
Please select :
1
1->Adding a House
2->Adding a Market
3->Adding a Office
4->Adding a Playground
5->Go Previous Menu
Please select :

2
Please enter side number 0 for back 1 for front:
0
Please enter position number:
23
Please enter Height number:
23
Please enter Length number:
23
Please enter owner of the market:
drfs
Please enter opening hour:
23
Please enter opening minute:
234
INVALID
1->Editing Mode

```

Also Drive code testing many other approaches..

5. Running command and results

```
Please enter street length:
50
1->Editing Mode
2->Viewing Mode
3->Exit
Please select program mode:
1
1->Adding new element
2->Deleting one element from last
3->Go previous menu
Please select :
1
1->Adding a House
2->Adding a Market
3->Adding a Office
4->Adding a Playground
5->Go Previous Menu
Please select :
1
Please enter side number 0 for back 1 for front:
0
Please enter position number:
0
Please enter Height number:
20
Please enter Length number:
15
Please enter Room number:
16
Please enter owner of the house:
Ali
Please enter color of the house:
Red
Adding operation done successfully
```

```
1->Editing Mode
2->Viewing Mode
3->Exit
Please select program mode:
1
1->Adding new element
2->Deleting one element from last
3->Go previous menu
Please select :
1
1->Adding a House
2->Adding a Market
3->Adding a Office
4->Adding a Playground
5->Go Previous Menu
Please select :

3
Please enter side number 0 for back 1 for front:
0
Please enter position number:
18
Please enter Height number:
23
Please enter Length number:
12
Please enter owner of the office:
Dany
Please enter job type of the office:
Software Development
Adding operation done successfully
```

```
W 1->Adding a House
n 2->Adding a Market
  3->Adding a Office
  4->Adding a Playground
  5->Go Previous Menu
  Please select :

2
Please enter side number 0 for back 1 for front:
0
Please enter position number:
5
Please enter Height number:
24
Please enter Length number:
23
Please enter owner of the market:
deneme
Please enter opening hour:
9
Please enter opening minute:
0
Please enter closing hour:
23
Please enter closing minute:
59
The specified land is already occupied !!
Adding operation failed !!
```

```
1 1->Adding a House
  2->Adding a Market
  3->Adding a Office
  4->Adding a Playground
  5->Go Previous Menu
  Please select :

2
Please enter side number 0 for back 1 for front:
1
Please enter position number:
15
Please enter Height number:
24
Please enter Length number:
17
Please enter owner of the market:
Kaya
Please enter opening hour:
8
Please enter opening minute:
50
Please enter closing hour:
23
Please enter closing minute:
50
Adding operation done successfully
```

```

Please select program mode:
2
  1->Display the total remaining length of lands on the street
  2->Display the List of Buildings on the street
  3->Display the number and ratio of length of playgrounds in the street
  4->Display the total length of street occupied by the markets, houses or offices
  5->Display the Skyline Silhouette of the street
  6->Go Previous Menu
Please select :

1
Total remaining length of lands:
w Back side of the street: 23
m Front side of the street: 33
  1->Editing Mode
  2->Viewing Mode
  3->Exit
Please select program mode:
4
Wrong Input

  1->Editing Mode
  2->Viewing Mode
  3->Exit
Please select program mode:
2
  1->Display the total remaining length of lands on the street
  2->Display the List of Buildings on the street
  3->Display the number and ratio of length of playgrounds in the street
  4->Display the total length of street occupied by the markets, houses or offices
  5->Display the Skyline Silhouette of the street
  6->Go Previous Menu
Please select :

4
Total length of street occupied by ;
Houses: 15
Markets: 17
Offices: 12

```

```
1->Editing Mode
2->Viewing Mode
3->Exit
Please select program mode:
2
1->Display the total remaining length of lands on the street
2->Display the List of Buildings on the street
3->Display the number and ratio of length of playgrounds in the street
4->Display the total length of street occupied by the markets, houses or offices
5->Display the Skyline Silhouette of the street
6->Go Previous Menu
Please select :
2

Features of Ali's house ;
Owner printing with focus method!
Color: Red
Number of rooms: 16
Position: 0
Height of the house: 20
Length of the house: 15

Features of Dany's office ;
Job Type of the office: Software Development (printing with focus method)
Position: 18
Height of the office: 23
Length of the office: 12

Features of Kaya's market ;
Opening time: 08:50
Closing time: 23:50 (printing with focus method)
Position: 15
Height of the market: 24
Length of the market: 17
```

```
2
1->Display the total remaining length of lands on the street
2->Display the List of Buildings on the street
3->Display the number and ratio of length of playgrounds in the street
4->Display the total length of street occupied by the markets, houses or offices
5->Display the Skyline Silhouette of the street
6->Go Previous Menu
Please select :
```

5




```

1->Adding new element
2->Deleting one element from last
3->Go previous menu
Please select :
1
  1->Adding a House
  2->Adding a Market
  3->Adding a Office
  4->Adding a Playground
  5->Go Previous Menu
Please select :
4
Please enter side number 0 for back 1 for front:
0
Please enter position number:
30
Please enter Length number:
15
Adding operation done successfully

1->Editing Mode
2->Viewing Mode
3->Exit
Please select program mode:
2
  1->Display the total remaining length of lands on the street
  2->Display the List of Buildings on the street
  3->Display the number and ratio of length of playgrounds in the street
  4->Display the total length of street occupied by the markets, houses or offices
  5->Display the Skyline Silhouette of the street
  6->Go Previous Menu
Please select :
3

Total length of the playgrounds in the street: 15
Ratio of length of playgrounds in the street: 0.150000
1->Editing Mode

```



Another Street i made.

for LDLinkedList<E> version displayListView() method;

_____TESTING displayListView() METHOD_____

In this section displayListView() method displays the elements of lists.

view of back list :

house@4f3f5b24 ==> market@15aeb7ab ==> playground@7b23ec81

view of front list :

house@7cc355be ==> house@6e8cf4c6 ==> market@12edcd21 ==> office@34c45dca