Proje'de Kullanılan Veri Yapılarının Nedenleri ve açıklamaları

List

```
private List<DailyFoodMenu> menuList;
```

We used java Linked List Data structure to keep our daily menu information,since we will add a new menu everyday and **adding end of the linked list is in O(1)** time as you can see following picture;

Operation \ List	LinkedList	ArrayList
get(int index)	O(n/4) average	O(1)
add(E element)	O(1)	O(1) amortized O(n) worst-case
add(int index, E element)	O(n/4) average O(1) if index is 0	O(n/2) average
remove	O(n/4) average	O(n/2) average
Iterator.remove()	O(1)	O(n/2) average
ListIterator.add(E element)	O(1)	O(n/2) average

Queue

```
private Queue<HealthAppointment> healthAppointmentsQueue;
```

We used java Queue Data structure to keep our health appointment information for prison doctor, since queue data structures is a first in first out data structure, it is good for appointments queue because who first create an appointment will be checking first by doctor.

And also other advantages of queue getting top element is in O(1) time.

Balanced Binary Search Tree.

```
private AVLTree<Inmate> prisonersTree;
```

For the inmates information in the prison we used AVLTree implementation from book as balanced search tree since we will do many search on inmates and by using avl tree we can search an inmate in O(logn) time.

Note: We change BinaryTree to AVLTree since we compare inmates according to their IDs and IDs is given in order and if we used binary search tree we will get only right subtrees so searching a binary search tree with only right subtrees is in O(n) time, but now since AVL tree is balanced itself doing rotation so our tree will always be balanced.

Priority Queue

```
private PriorityQueue<ToDo> activeToDoQueue;
```

We used java PriorityQueue Data structure to keep our tasks to be done for personnel according to importance of that task. A personnel must do most important job to do other jobs. The important thing about PriorityQueue a personnel can not do other tasks without doing most important job, In other words, personnel can perform its tasks in an importance order.

Set – Map

```
private Map<Inmate, Set<Visitor>> visitorsMap;
```

In our system we have visitor information for inmates. Since each inmate has its own visitors we thought that we can do mapping visitors to the inmates. In this data structure;

key : Inmate

value: Set<Visitor>

In this case we can reach easily visitors of an inmate.

SkipList

```
SkipList<Personnel> allPersonnel;
```

For the personnel information in the prison we used SkipList implementation from book as since while a personnel login to the system we need to search for ID and password to verify. So doing search in skip list is in O(logn) and it provide us a fast search while password checking.

Graph

```
ListGraph<Block> graph = new ListGraph<>(3,true);
```

By using graph we will represent the our prison structure and information. We assumed we have 3 block in the prison as in picture and blocks are connected each other each block has 2 floor one for inmate second one for personnel. Also in the graph we have this informations.

By using graph we can see all information about the prison structure. For a generic graph data type we used 2nd part implementation of 8th homework.

A sample Graph representation is;