|  |  |
| --- | --- |
| HashMap | HashTable |
| It is non-synchronized in nature. | It is synchronized in nature. |
| Allows only one null key but multiple null values. | Doesn’t allow any null key or value. |
| Has faster processing. | Has slower processing. |
| Can be traversed by iterator. | Can be traversed by iterator and numeration. |
| Inherits abstracting class. | Inherits dictionary class. |

1. Difference between HashMap and HashTable.
2. LinkedList vs ArrayList.

|  |  |
| --- | --- |
| LinkedList | ArrayList |
| Implements doubly linked list internally to store elements. | Implements dynamic array internally to store elements. |
| Manipulation of elements is faster. | Manipulation of elements is slower. |
| Can act as a list and queue. | Can act only as a list. |
| Effective for data manipulation. | Effective for data storage and access. |

1. ArrayList vs Vector

|  |  |
| --- | --- |
| ArrayList | Vector |
| Non-synchronized in nature. | Synchronized in nature. |
| It is not a legacy class. | It is a legacy class. |
| Increases size by half of the array list. | Increases size by double of the array list. |
| It is not thread safe. | It is thread safe. |

1. Difference between List, Set and Map.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Interface | Duplicates allowed? | Null values allowed? | Insertion order preserved? | Iterator | Data Structure |
| List | Yes | Yes, multiple null values are allowed | Yes and can retrieve using index | Iterator | Array |
| Set | No | Yes, but only once | No | Iterator, ListIterator | Underlying Map implementation |
| Map | Not for Keys | Yes, but only once for keys, can have multiple null values | No | Through keyset value and entry set | Hashing techniques |

1. When to go for list?
   * ArrayList – Maintains insertion order.

No modifications.

Not safe thread.

* Linked List – Manipulation of list.

Frequent updates.

Not safe thread.

* Vector – Thread safe.

Maintains insertion order.

One thread at a time in multithreading.

1. When to go for set?

* HashSet – It removes duplicates.

It doesn’t maintain order.

It allows only one null value.

* Linked HashSet – It removes duplicates.

It maintains order.

* Tree Set – It removes duplicates.

It provides either ascending or descending order.

1. When to go for map?

* HashMap – It provides key, value pair relation based on index.

Inserting a key will give corresponding value.

Allows one null key and multiple null values.

Not thread safe.

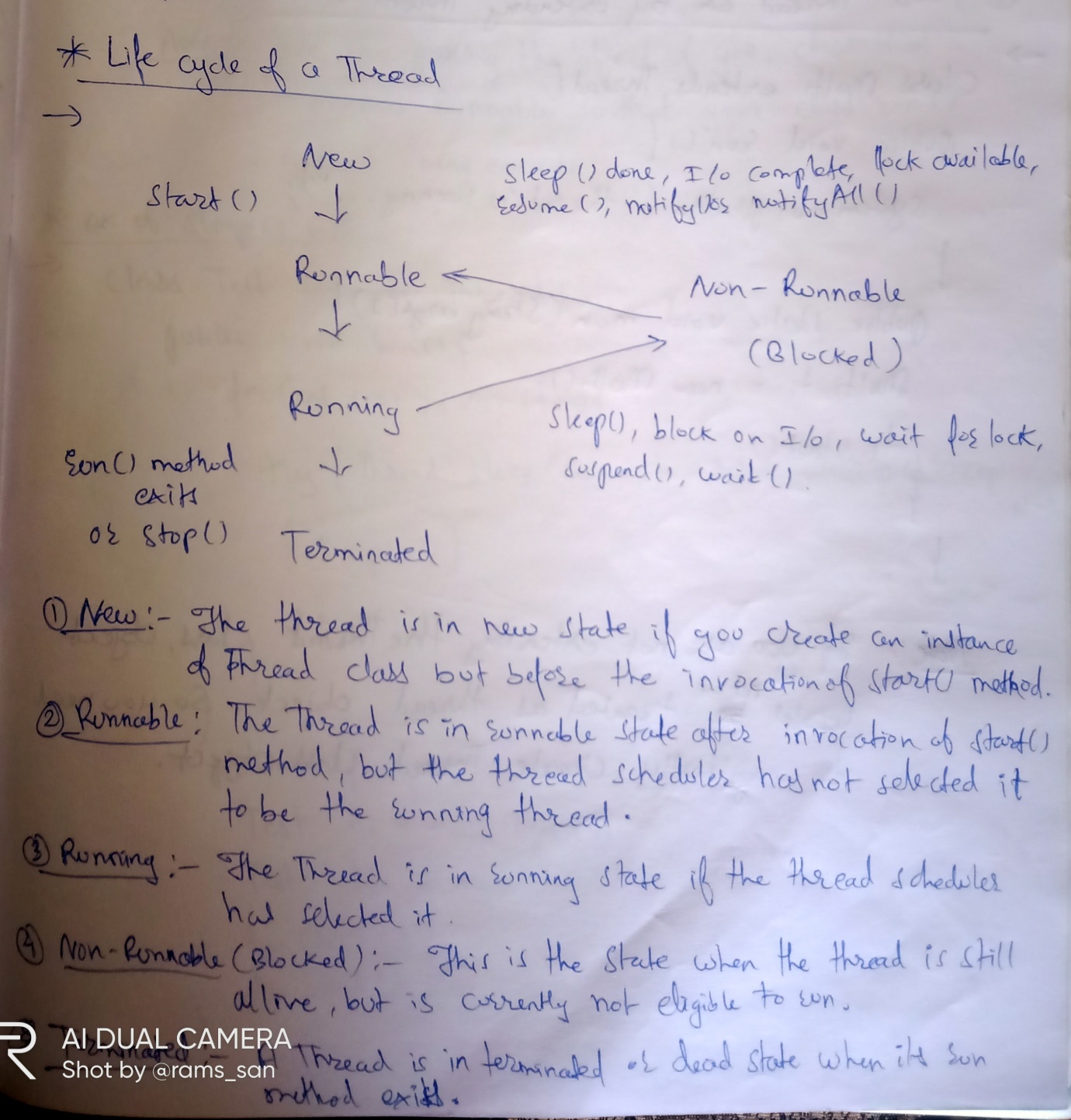
* Hash Table – It is thread safe.

It doesn’t allow any null key.

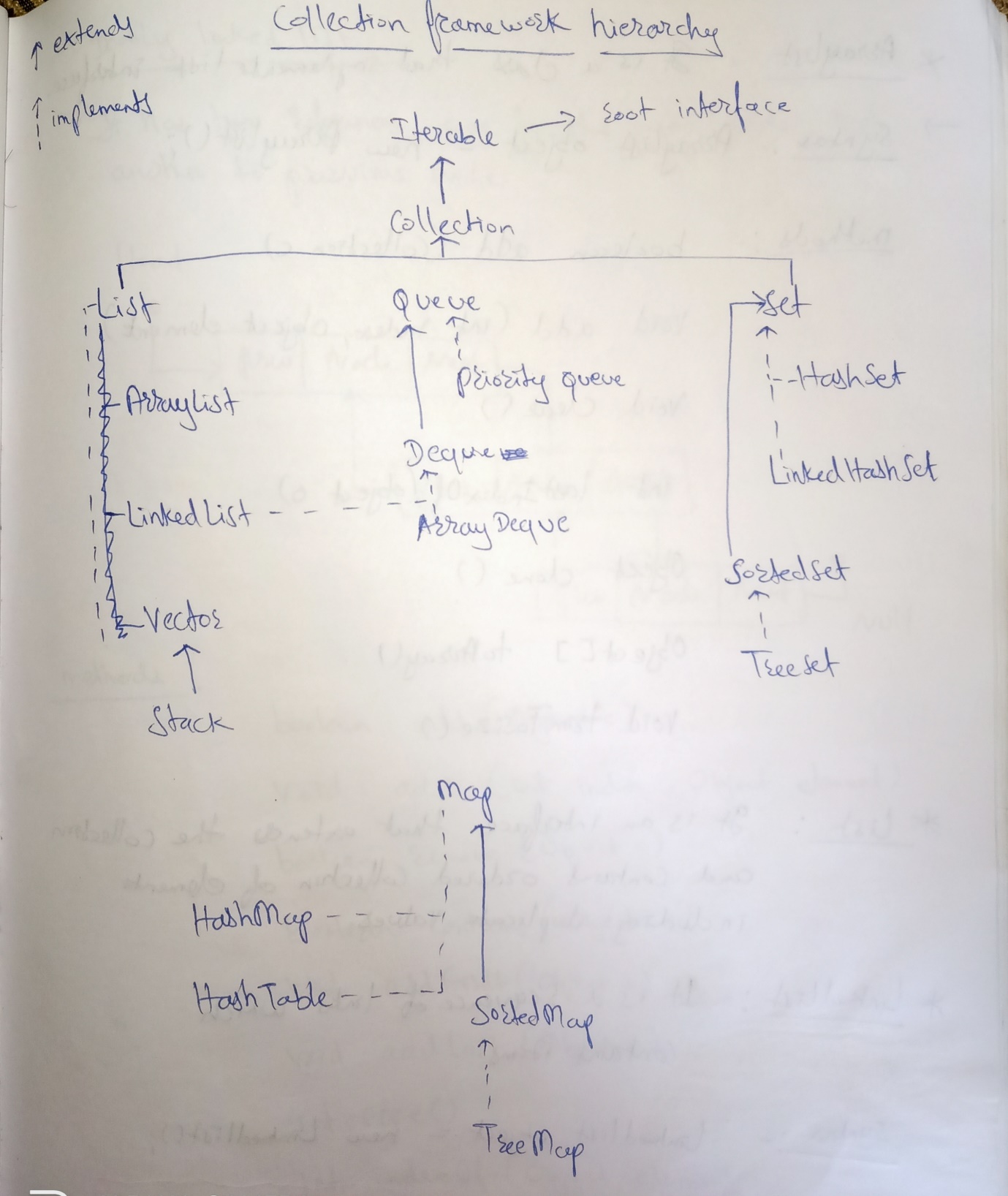
1. Difference between Extending Thread Class and Implementing Runnable Interface.

|  |  |
| --- | --- |
| Extending Thread Class | Implementing Runnable Interface |
| You cannot extend another class. | You can extend another class. |
| Task and runner are highly coupled. | Task and runner are loosely coupled. |
| No code reusability. | Gives better code reusability. |
| Overhead of additional method. | No overhead of any method. |
| Maintenance of code is not easy. | Maintenance of code is lot easier. |
| No need to create a thread class object explicitly. | Needs to create a thread class object explicitly and pass the object of class that implements runnable interface to the thread class constructor. |

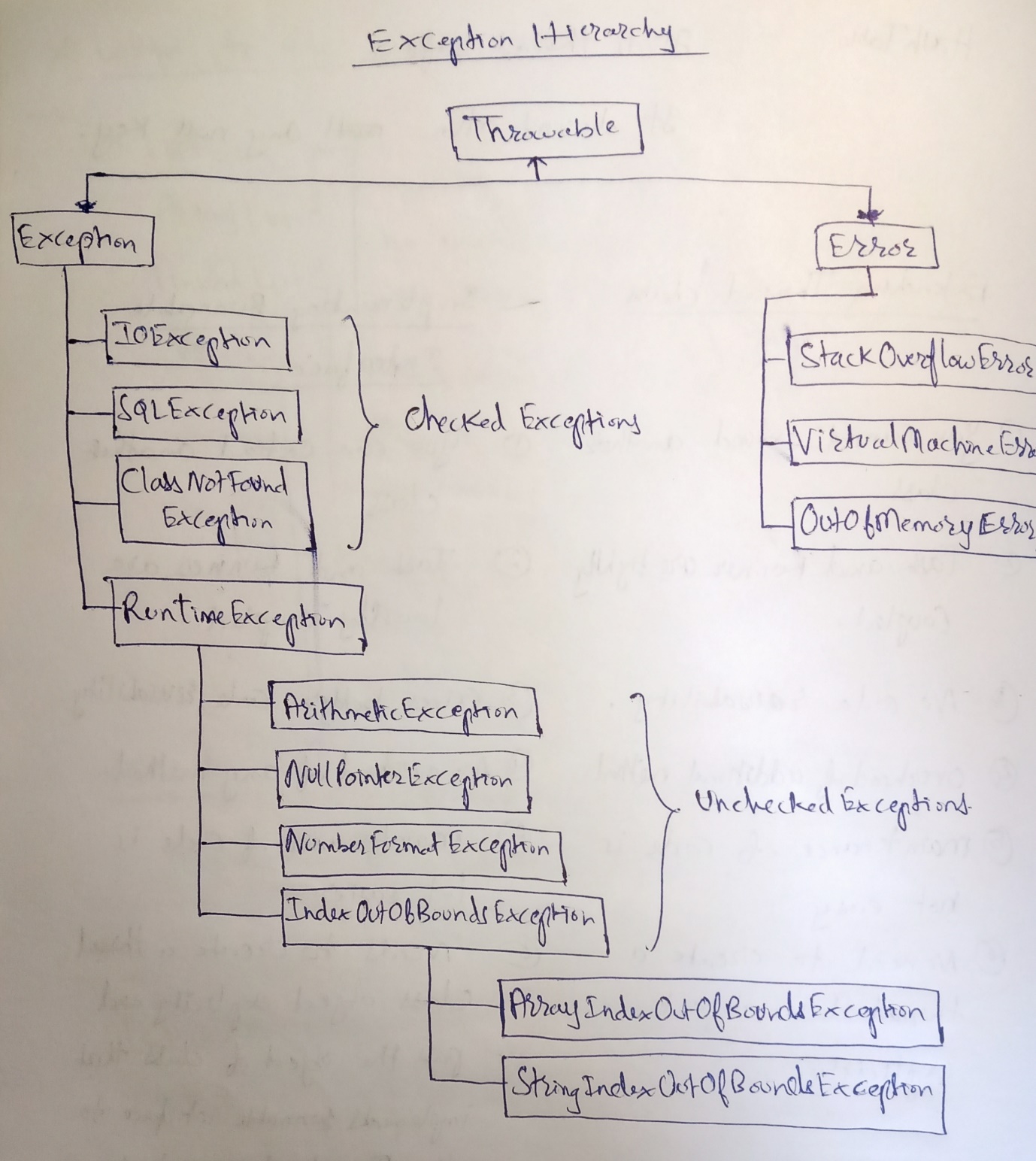
1. Thread life cycle with all methods.



1. Collection hierarchy including map.



1. Exception hierarchy.



1. Difference between throw and throws keyword.

|  |  |
| --- | --- |
| throw | throws |
| It is used to explicitly throw an exception. | It is used to declare an exception |
| Checked exception cannot be propagated using throw only. | Checked exception can be propagated using throws. |
| Throw is followed by an instance. | Throws is followed by class. |
| Throw is used within the method. | Throws is used with the method signature. |
| You cannot throw multiple exception. | You can declare multiple exceptions. |

1. To get a synchronized list from an array list.

We use the synchronized list (List<T>) method in java. It accepts the array list as an argument and returns a thread safe list.

Ex:- List<String> list = new ArrayList <String>();

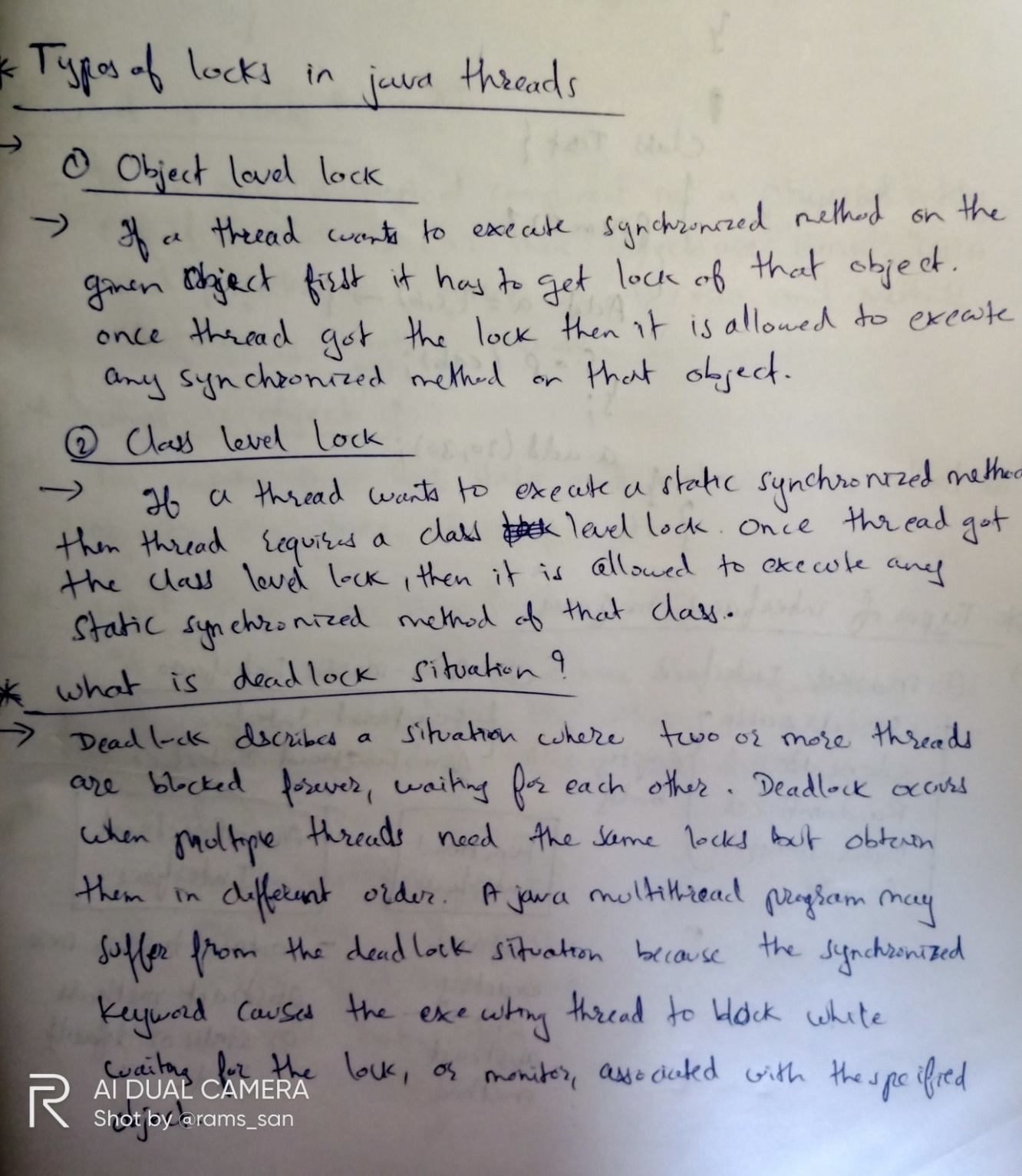
List.add(“hello”);

List.add(“Sir”);

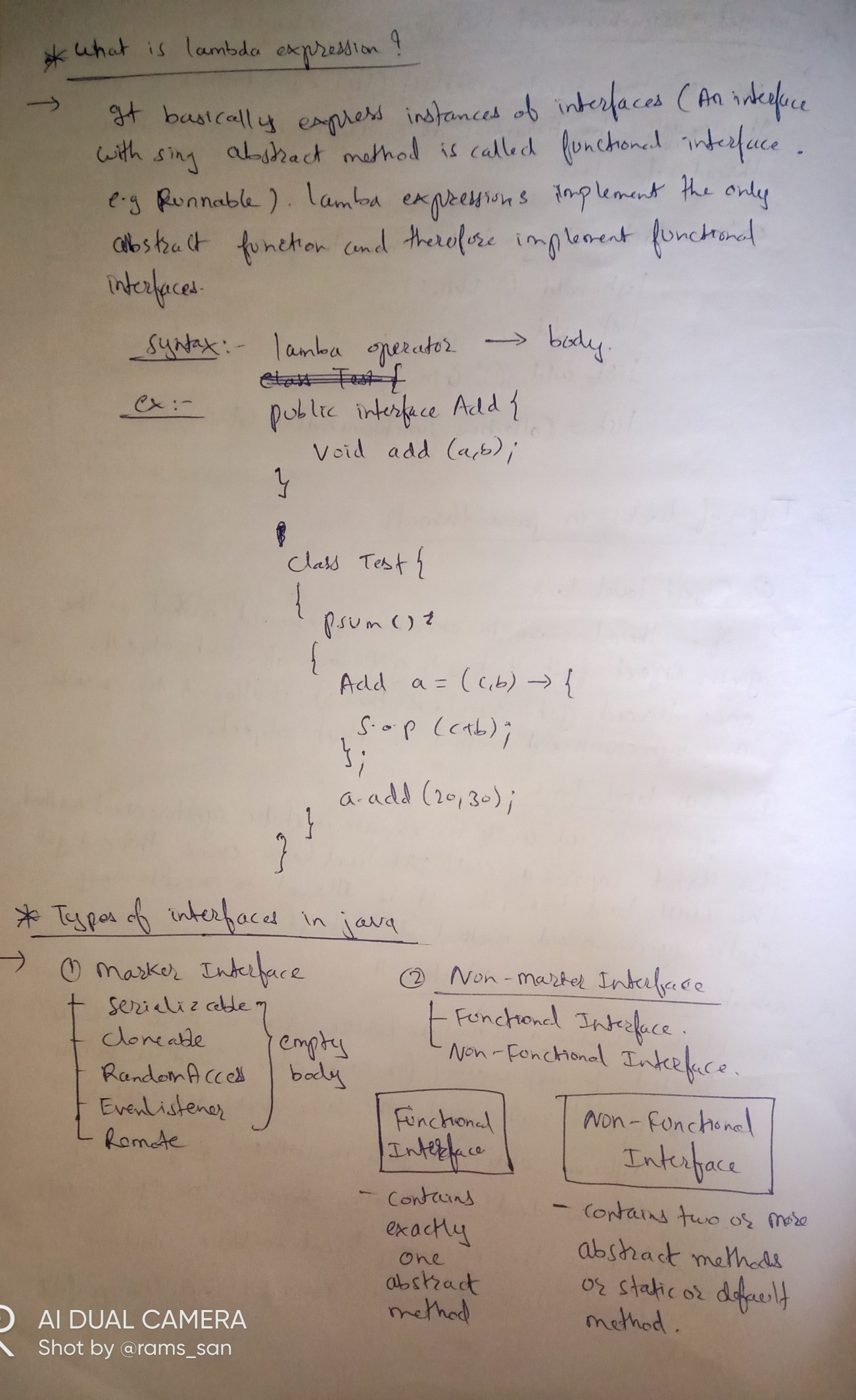
List.add(“GM”);

List = Collections.synchronizedList(list);

1. Types of lock in java threads.
2. 15.What is deadlock situation?



1. Types of interfaces in java.
2. What is lambda expression?



1. How to handle null pointer exception ?

