1. What is difference between Hash Table and Hash Map ?

Hash Map	Hash Table
i> Not synchronized	i> Synchronized
ii> Not thread safe and can't be share between many threads without proper synchronization	ii> Thread safe and can be shared with many threads
iii> Allow one null key and multiple null value	iii> Doesn't allow any null key or value
iv> It is new class introduced in JDK 1.2	iv> Legacy class
v> Fast	v> Slow
vi> Traversed by Iterator	vi> Traversed by enumerator and iterator
vii> Iterator in hash map is fail fast	vii> Enumerator in hash table is not fail fast
viii> Inherits abstract map class	viii> Inherits dictionary class

2 Linked list Vs Array list

Array list	Linked list
i) Use dynamic array to	i> Use doubly linked

store elements ii) Slow because it uses array iii) Bit shifted iv) Act as list v) Implements list vi) Better for storing and accessing data	list to store elements ii) Faster because it uses doubly linked list iii) Not bit shifting iv) Act as list and queue v) Implements list and deque vi) Better for manipulating data

3 Array list Vs Vector

Array list	Vector
i) Not synchronized ii) Increments 50% of current array size if number of elements exceeds from its capacity iii) Not legacy class, introduced in JDK 1.2 iv) Fast because not synchronized	i) Synchronized ii) Increments 100% means doubles the array size if total number of elements exceeds than its capacity iii) Legacy class iv) Slow because synchronized

v) Uses the iterator interface to traverse elements		v) Use iterator interface or enumeration interface to traverse elements

4. Difference between List, set and map

List	Set	Map
i) Ordered sequence	i) Ordered or unordered sequence	i) Ordered or unordered sequence
ii) Contains duplicate elements	ii) Contains only distinct elements	ii) Doesn't permit duplicate keys
iii) Allow any number of null values	iii) Contains at most one null element	iii) Allow null as key and value
iv) Implemented by Array List, Linked List	iv) Implemented by HashSet, TreeSet, Linked HashSet	iv) Implemented by HashMap, TreeMap, Linked HashMap

5. When to go for List ?

A List can be used when insertion order of elements need to maintained.

6. When to go for Map ?

A map can be used when data is key-value pairs and need fast retrival of value based on some key

7. When to go for set?

A set can be used when we need to maintain collection that contains no duplicates.

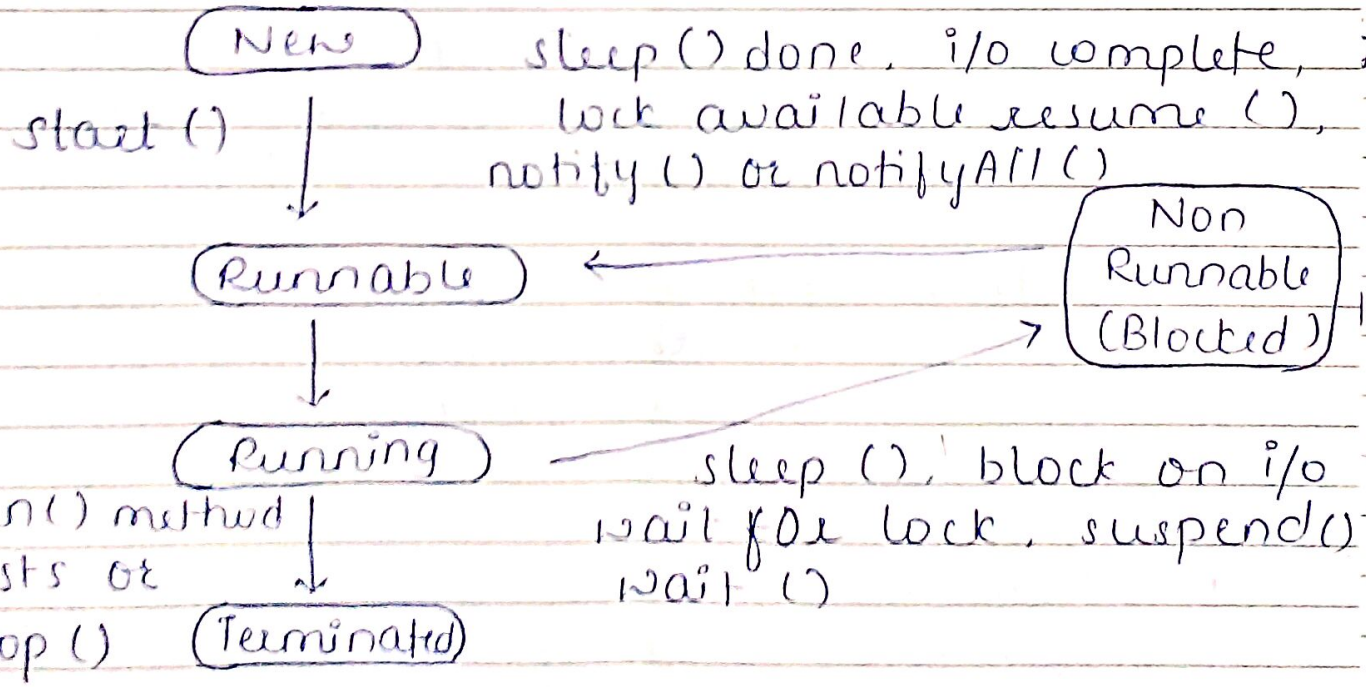
8. Difference between extending Thread class and implementing runnable interface.

Thread Class	Runnable Interface
i) Each thread create a unique object and gets associated with it	i) Multiple threads share the same object
ii) More memory is required	ii) less memory is required
iii) Multiple inheritance not allowed here it can extend only Thread class not any other class	iii) Implements the runnable interface so it has chance to extend one class
iv) Extends only if it want's to override the	iv) Implements only if we want to speci-

other methods of Thread class	alize run method
v) Tight coupling	v) Loose coupling

9. Thread life cycle with all the methods.

Life cycle of ~~method~~ thread in java is controlled by JVM.
The java thread states are as follows
i) New
ii) Runnable
iii) Running
iv) Non Runnable (Blocked)
v) Terminated



i) New - The thread is in new state if you create an instance of Thread class but before invokation of start() method

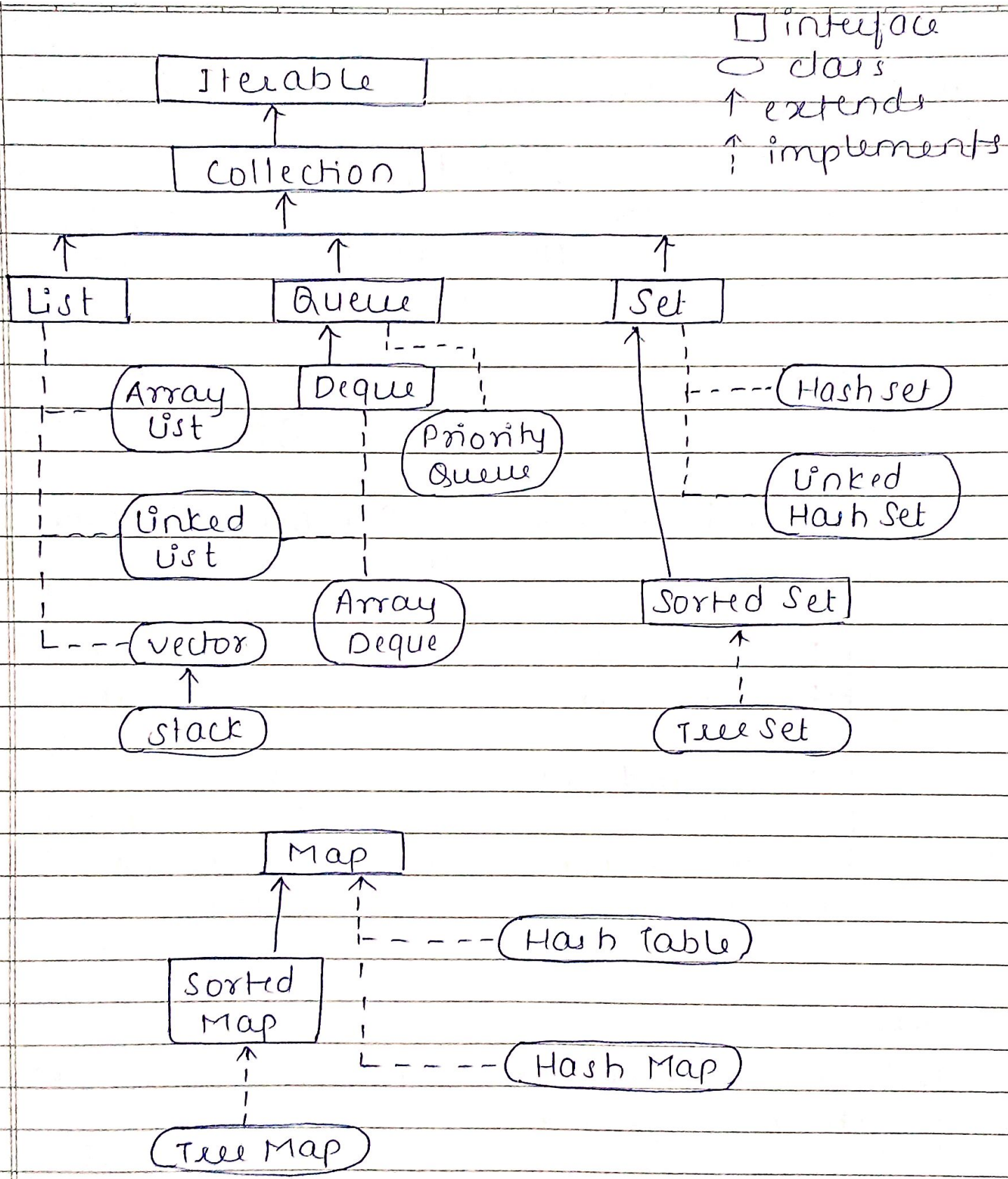
ii) Runnable - The thread is in runnable state after invokation of start() method but thread scheduler has not selected it to be running thread

iii) Running - The thread is in running state if thread sheduler has selected it

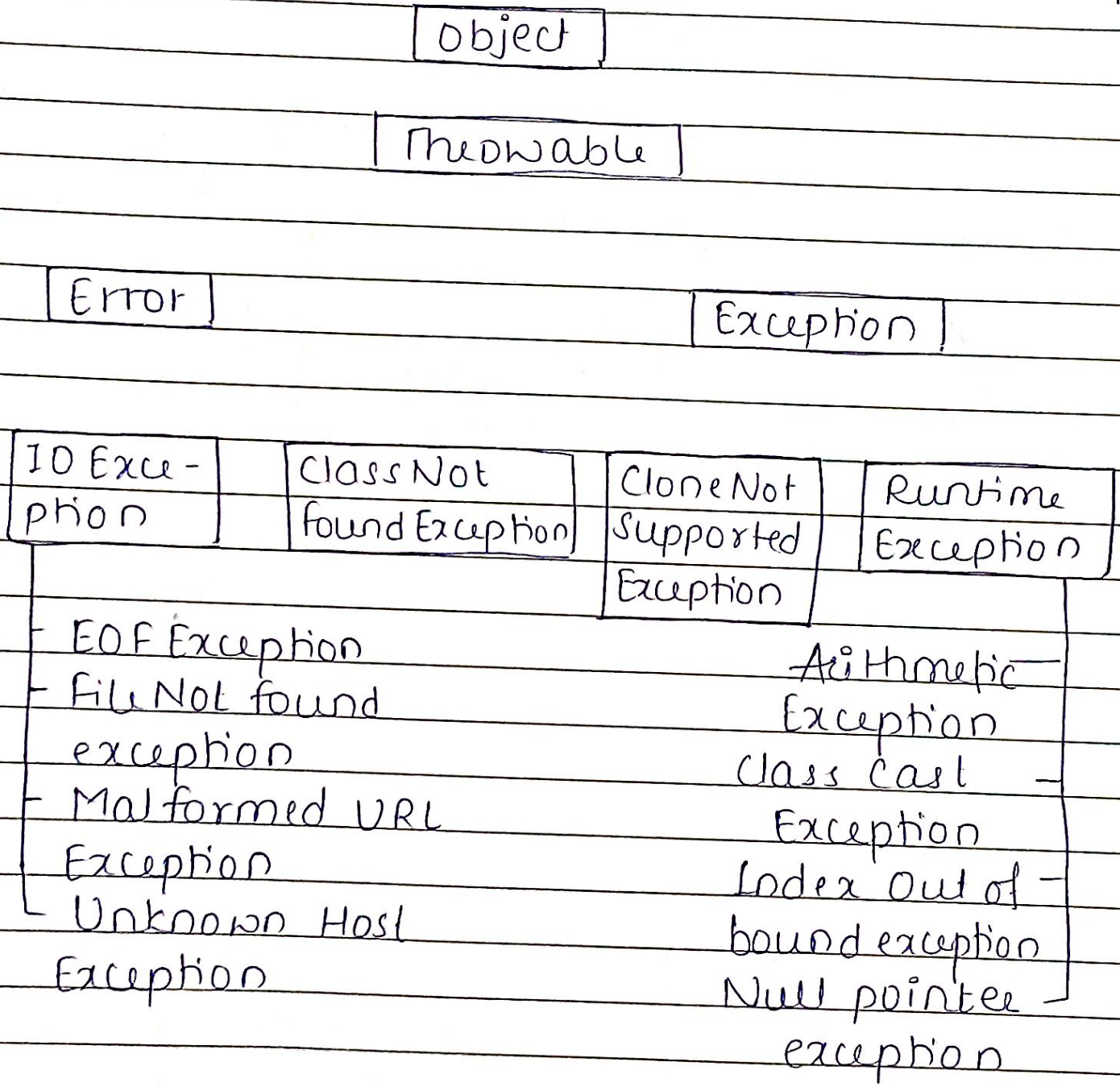
iv) Non-Runnable (Blocked) - This is state when thread is still alive, but is currently not eligible to run

v) Terminated - A thread is terminated or dead state when its run() method exists.

10. Draw the diagram for collection hierarchy (including map on side)


interface
class
extends
implements
Iterable
Collection
List
Queue
Set
Array List
Deque
Hash set
Priority Queue
Linked List
Linked Hash Set
Array Deque
Sorted Set
Vector
Stack
Tree Set
Map
Hash Table
Sorted Map
Hash Map
Tree Map

11. Draw the diagram for exception hierarchy



12. What is difference between throw and throws keyword ?

Throw	Throws
i) Used to explicitly throw an exception	i) Used to declare an exception
ii) Checked exception can not be propagated using throw only	ii) Checked exception can be propogated using throws
iii) Followed by an instance	iii) Followed by class
iv) Used within method	iv) Used with method signature it use
v) Can not throw multiple exception	v) declare multiple exceptions

13. How to make Array list synchronized ?

There are two way to create synchronized array list.

i) Collections.synchronizedList() method
ii) Using copy on write array list

i) Using collections.synchronized list() method
- Example

```
List<String> list = collections.SynchronizedList
    (new ArrayList<String>());
```

ii) Using copy on write array list
- Example.

CopyOnWriteArrayList <T> threadsafeList = new CopyOnWriteArrayList <T> ();

If we try to modify copy on write array list through iterator's own method then it throws UnsupportedOperationException.

14. Types of locks in Java Threads.

There are two types of locks in java thread
i) Object level lock
ii) class level lock

i) Object level lock
- Object level lock is mechanism when we want to synchronize a non static method or non static code block such that only one thread will be able to execute the code block on give instance of class.

Example
```
public class Demo {
    public synchronized void demo () {
    }
}
```

OR

```
public class Demo {
    public void demo () {
        synchronized (this) {
        }
    }
}
```

OR

```
public class Demo {
    private final object lock = new Object ();
    public void demo () {
        synchronized (lock) {
        }
    }
}
```

ii) Class level lock

It prevents multiple threads to enter in synchronized block in any of all available instance of class on runtime

Example

```
public class Demo {
    public synchronized static void demo () {
    }
}
```

OR

```
public class Demo {
    public void demo () {
        synchronized (Demo.class) {
        }
    }
}
```

OR

```
public class Demo {
    private final static Object lock =
                                new Object();
    public void demo () {
        synchronized (lock) {
        }
    }
}
```

15. What is deadlock situation

- It is part of multithreading
- Deadlock can occur in a situation when thread is waiting for an object lock that is acquired by another thread and second thread is waiting for an object lock that is acquired by first thread
- Since both threads are waiting for

each other to release the lock, the condition is called deadlock.

Example.

```
public class Demo {
  public static void main (String [] args ){
    final string resource1 = "Manasi" ;
    final string resource2 = "Gayatri" };
    Thread t1 = new Thread () {
      public void run () {
        synchronized (resource 1) {
          syso ("Thread1 " );
        try {
          Thread.sleep (100);
        }catch ( Exception e){
        }
          synchronized (resource 2) {
            syso (" Thread 1 resource 2");
          }
      }
    }
  };
    Thread t2 = new Thread () {
    public void run () {
      synchronized (resource 2) {
      syoso (" Thread 2 resource 2" );
      try {
        Thread.sleep (100);
```

```
        }catch (Exception e) {
        }
        synchronized (resource1) {
            sysо ("Thread 2");
        }
    }
  }
};
    t1.start ();
    t2.start ();
  }
}
```

o/p - Thread 1
Thread 2 resource 2

16. Types of interfaces in java

Mainly there are two types of interfaces,
i) Marker interface ii) Non marker interface
Again Non marker interface divided into two i) functional interface ii) Non functional interface

i) Marker interface
- Marker Interface is interface with no fields or methods or in simple word

empty interface in java is called marker interface
Example -
Serializable, Clonable, Remote Interface

ii) Functional Interface
Functional interface is an interface that contains only one abstract method. A functional interface can have any number of default methods.
Example -
Runnable, ActionListener, Comparable

iii) Non functional Interface
Non functional interface is an inteface that contains any number of abstract methods and default methods.

17. What is lambda expression

- It provides a clear and concise way to represent one method interface using an expression
- Very useful in collection library
- It helps to iterate, filter and extract data from collection
- lambda expression is treated as function, so

compiler does not create .class file
- We use lambda expression to provide implementation of functional interface
- less coding

Syntax

(argument-list) → {body}

- (argument-list): empty or non empty
- →: used to link argument list and body expression
- {body}: contains expression & statement for lambda expression

No parameter syntax

() → { }

One parameter syntax

(p1) → { }

Two parameter syntax

(p1, p2) → { }

Example

```
interface Drawable {
    public void draw();
}
public class Demo {
    public static void main(String[] args) {
        int width = 10;
```

```
        Drawable d2 = () -> {
            syso ("Drawing" + width );
        };
        d2.draw ();
    }
}
```

o/p - Drawing 10

18 How to handle null pointer exception without using try-catch block or throw and throws keyword.

- Ternary operator
- Use apache commons string utils for string operations
- check method arguments for null key early
- consider primitive rather than object
- using string.valueOf () rather than toString ()
- avoid returning null from your methods.