امتحان 2014

أحمد سعيد محمد عبده (7)

فادي يسري (46)

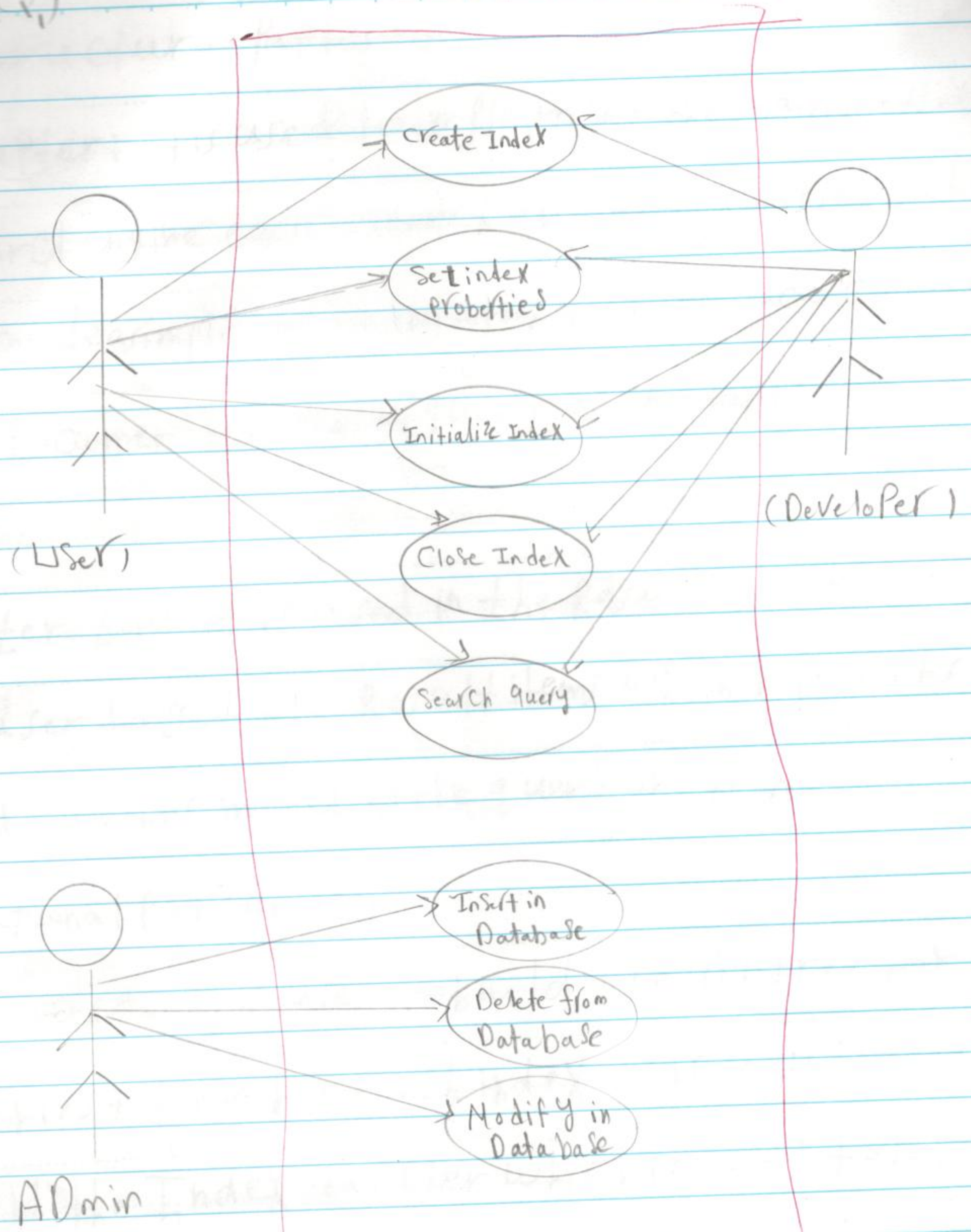
عمر خالد عبد العزيز (41)

محمد طارق (57)

عمرو العربي محمد (43)

B

(Q1)


Create Index
Set index Probeties
Initialize Index
Close Index
Search query
Insert in Database
Delete from Database
Modify in Database
(User)
(Developer)
ADmin

Q2) * structural patterns :-

① Adapter :- it is used to implement the main methods of search engine easily. search engine is complex but we provide a simple implementation of interface containing index(Document), search(query) as an adapter to the engine.

② Iterator :- it is used in the Resultset. it allows the user to get the Result items which represents the documents matching the query one by one.

* creational patterns :-

① Builder :- it is used to build the (B+) tree for the first time for each index. it is mainly used to build the Index searcher which is costy to be built. It may also be used to delete the index searcher as this operation is also costy.

Page :
Date :

② objectPool: it is used as a Pool for index searchers in order to provide indexsearcher quickly and take it again after finishing its work to avoid the Long time taken in creating and deleting it it creates minimum number of objects init in the begining and when it is nearly empty it creates new objects without exceeding the maximum Limit.

③ Singleton: it is used in the Index writer object. it ensures that there is only one object of the index writer in order to keep the (B+) tree consistent.

K.M.S

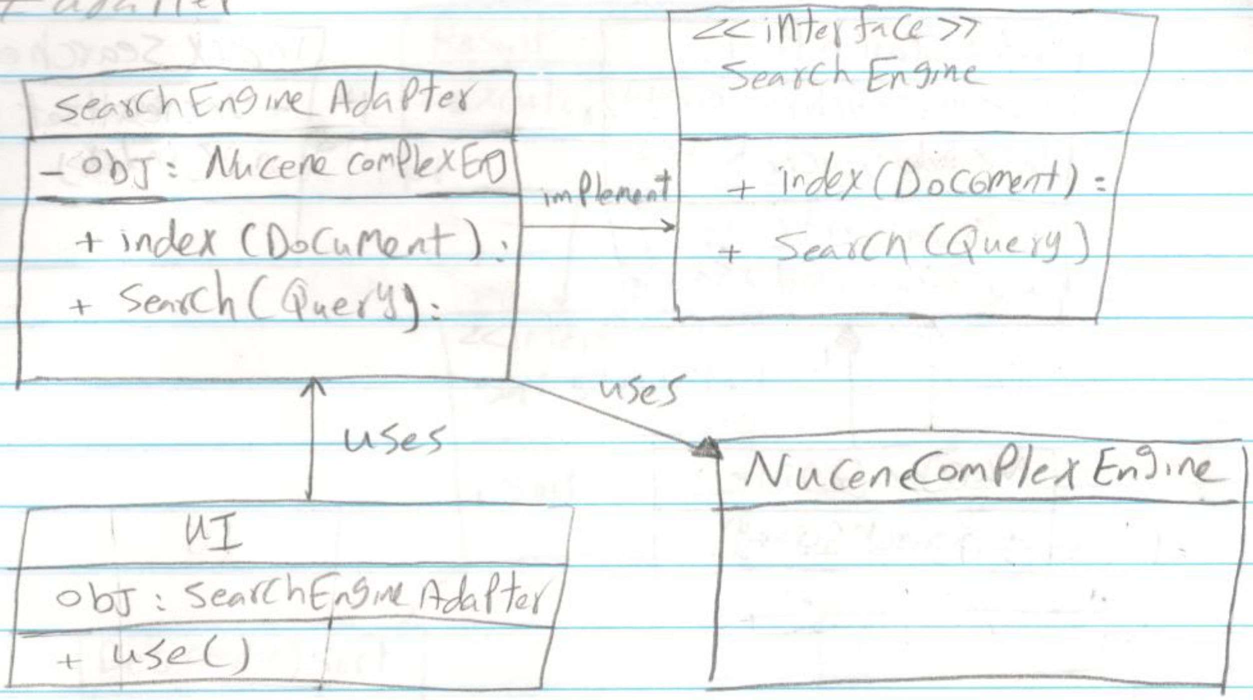
Page :
Date :

③ behavioral pattern:

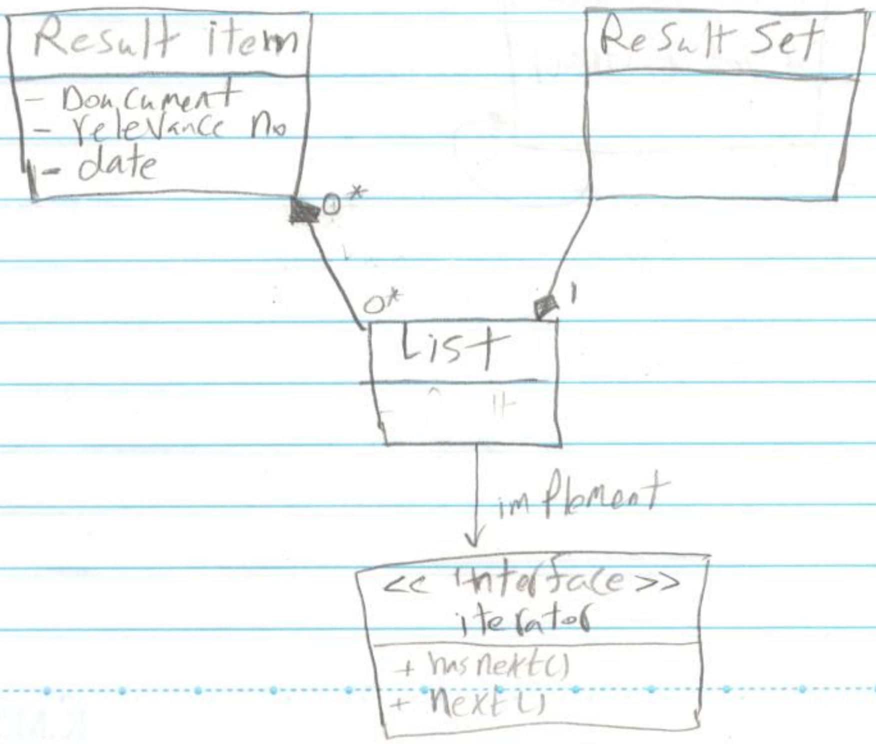
Strategy: it is used in choosing the proper sorting algorithm for the result set. it allows the result set to choose dynamically the sorting algorithm according to the query where 2 algorithms are implemented separately and one of them is chosen at runtime according to the query that the result set results from.

Q3

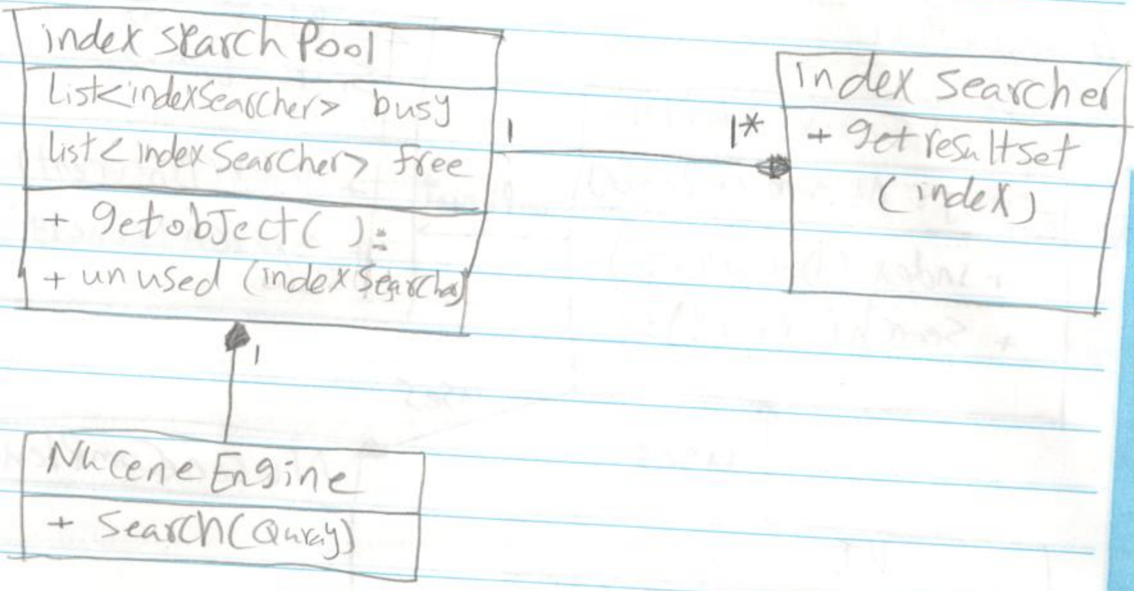
adapter



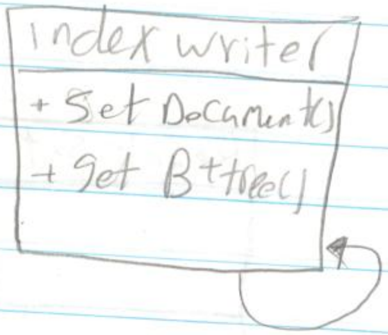
Iterator



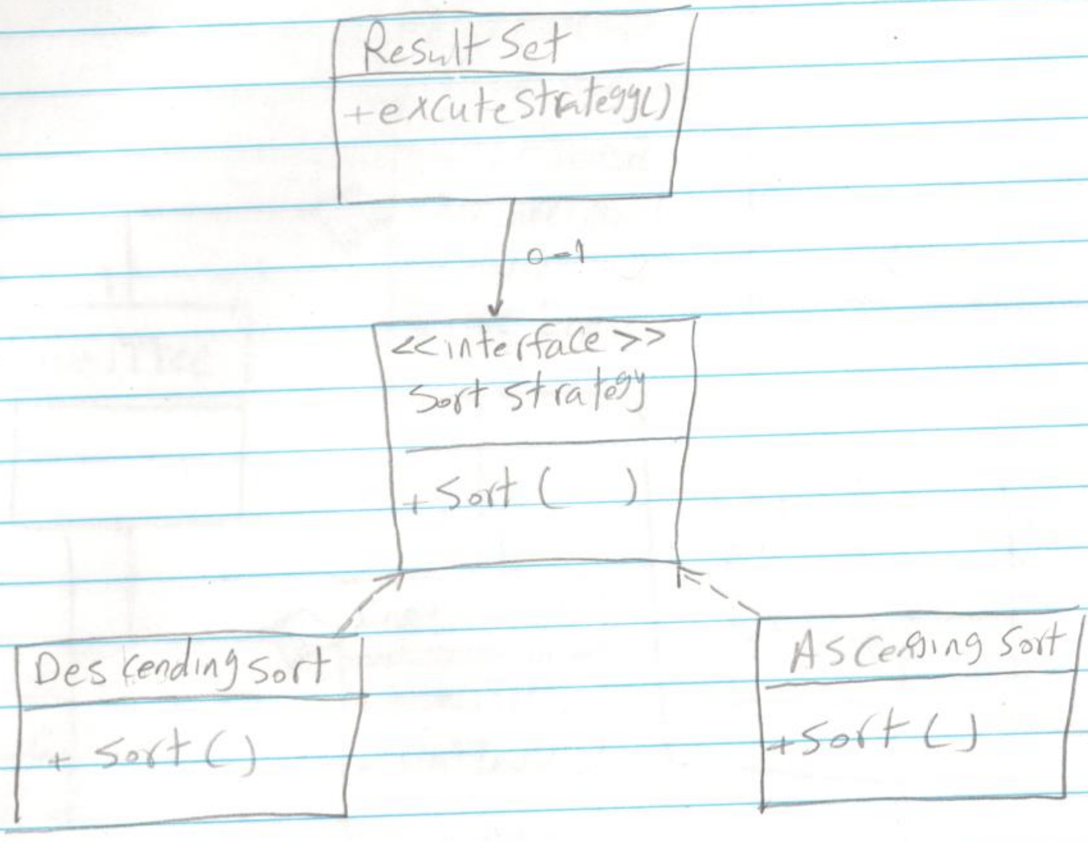
Pool



singlton :-

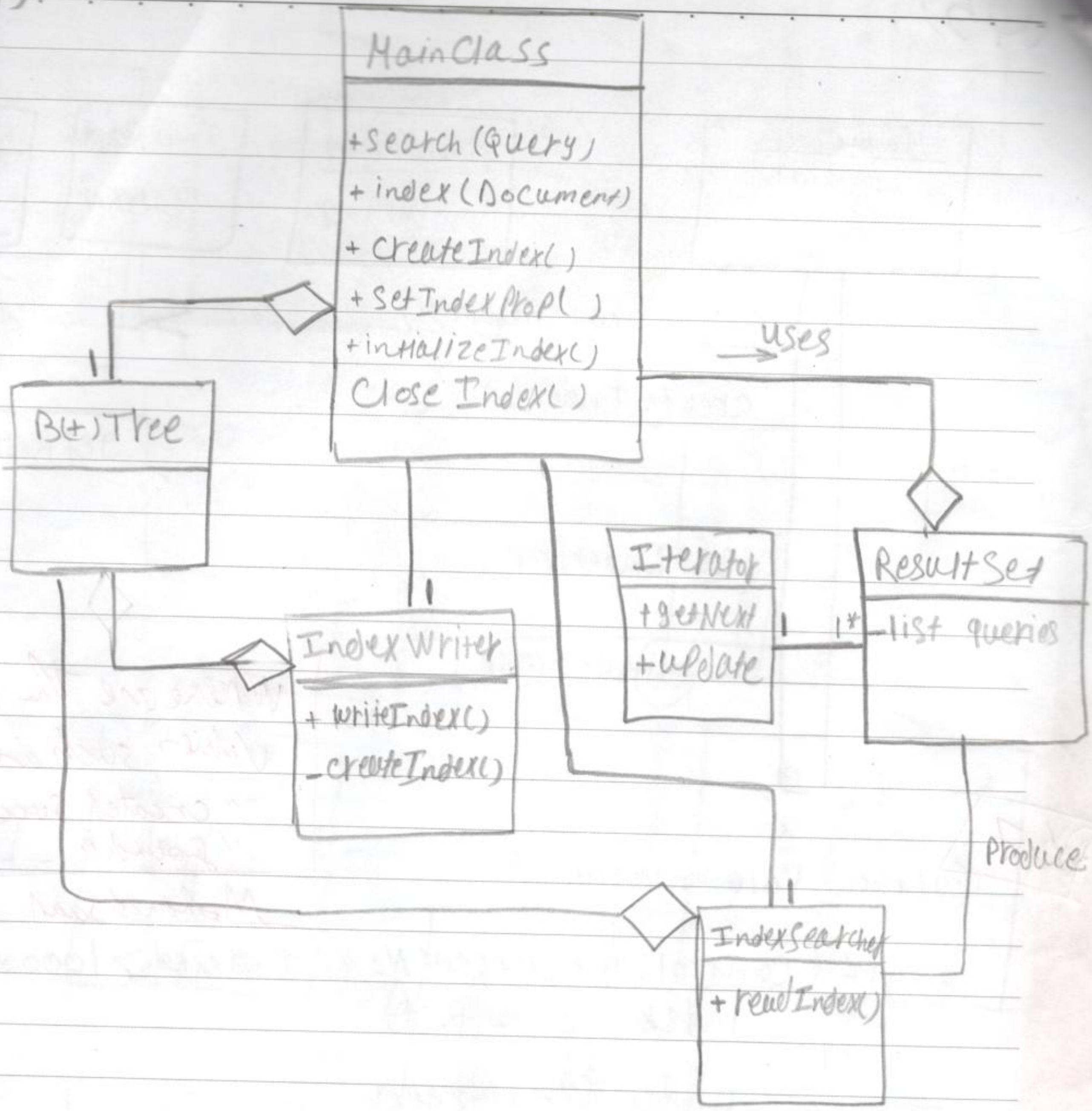


strategy:-


Result Set
+excuteStrategy()
0-1
<<interface>>
Sort strategy
+Sort ( )
Descending sort
+ Sort ( )
AScending Sort
+Sort ( )

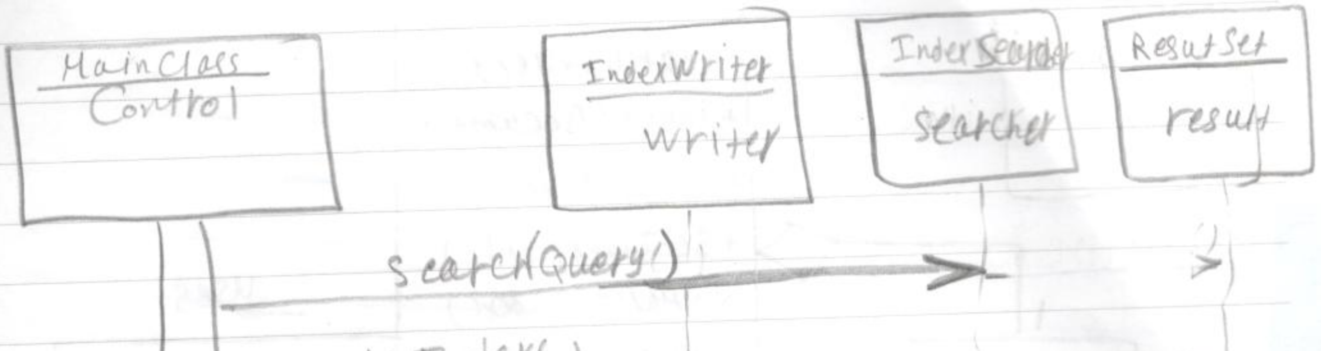
Page
Date

1(Q4)



Page
Date

(Q5)



Where are the return values such as
"created Successfully"
"found" ...
Mohamed Saad

Q6,7

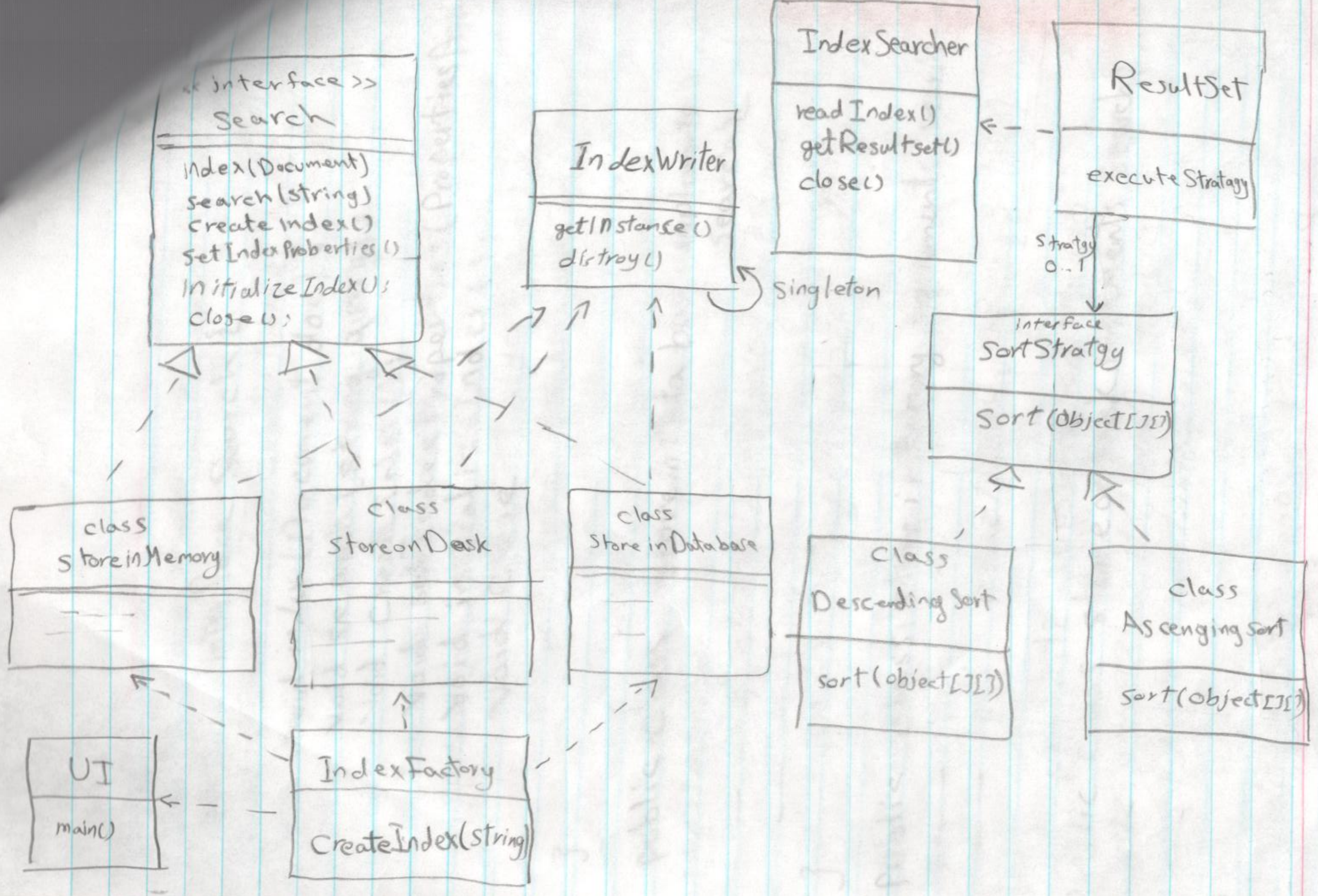
```
public void main() {
    if (control.getCurrentNoOfIndexed > 1000000)
        IndexSize = RED
    else
        indexSize = green

    if (Control.getCurNumOfRunning > 99)
        WorkLoad = red

    else
        WorkLoad = green.
}
```

This is Completely Wrong — Mohamed Saad
We need Design Pattern for Monitoring
and Sketch for only that not for the whole system !!

For the memories of yesterday,
For the happiness of today...



<< interface >>
Search
index(Document)
search(string)
create index()
Set Index Proberties ()
Initialize Index();
close();
IndexWriter
getInstanse()
distroy()
Singleton
IndexSearcher
read Index()
getResultset()
close()
ResultSet
execute Stratagy
Stratgy
0..1
interface
SortStratgy
Sort(Object[][])
class
storeinMemory
Class
StoreonDesk
class
Store in Database
Class
Descending Sort
sort(object[][])
class
Ascenging sort
sort(object[][])
UI
main()
IndexFactory
CreateIndex(string)

```
public interface Search{

      void index(Document doc);
      void search(string quary);
      Void Create Index();
      Void setIndexProperties(PropertiesProp
      void initializeIndex();
       void close

}

public class storeinData base implements
                                   search{

   ___ ___

   ______

}
public class storeinMemory implements search{
 ____
  ____

}
public class storeonDesk implements search{
 ____

 ____
}
```

```
public interface SortStrategy{
    void sort(object[][] data);

}
public class AscengingSort implements
                              sortSstrategy{
    void sort(object[][] data){
      // Asceng ing according

    }
public class DescendingSort implements
                              Sort Strategy{
    void sort(object[][] data){

       \\ descending according

    }

public class ResultSet{
        private Sortstrategy  strategy
        public  ResultSet( sortStratgy strategy){
               this.strategy=stratgy;

        }
        public void execute Strategy(object[][] da
        }  strategy.sort(data);
        }
```

[illegible] wat for the whole system !!!

```
public class IndexFactory {
    public Search creatIndex(string type){
        if (type == null){
            return null;
        }
        else if (type.equalsIgnoreCase("Disk")){
            return new StoreonDask();
        }
        else if(type.equalsIgnoreCase("Database")){
            return new store in Database();
        }
        else if(type.equalsIgnoreCase("memory")){
            return new store inMemory();
        }
```

```
public class Indexwriter {
private static IndexWriter instance = null;
   private Indexwriter(){

   }
   public static IndexWriter getInstance(){
      if(instance == null){
        instance = new IndexWriter();
      }
      return instance;
   }
   public static void destroy(){
        instance = null;
   }
}
```