OBJECT ORIENTED PROGRAMMING LAB REPORT

NAME: ANURAN CHAKRABORTY

CLASS: BCSE-II

SECTION: A1

ROLL NO.: 001610501020

CONTENTS

TOPICS
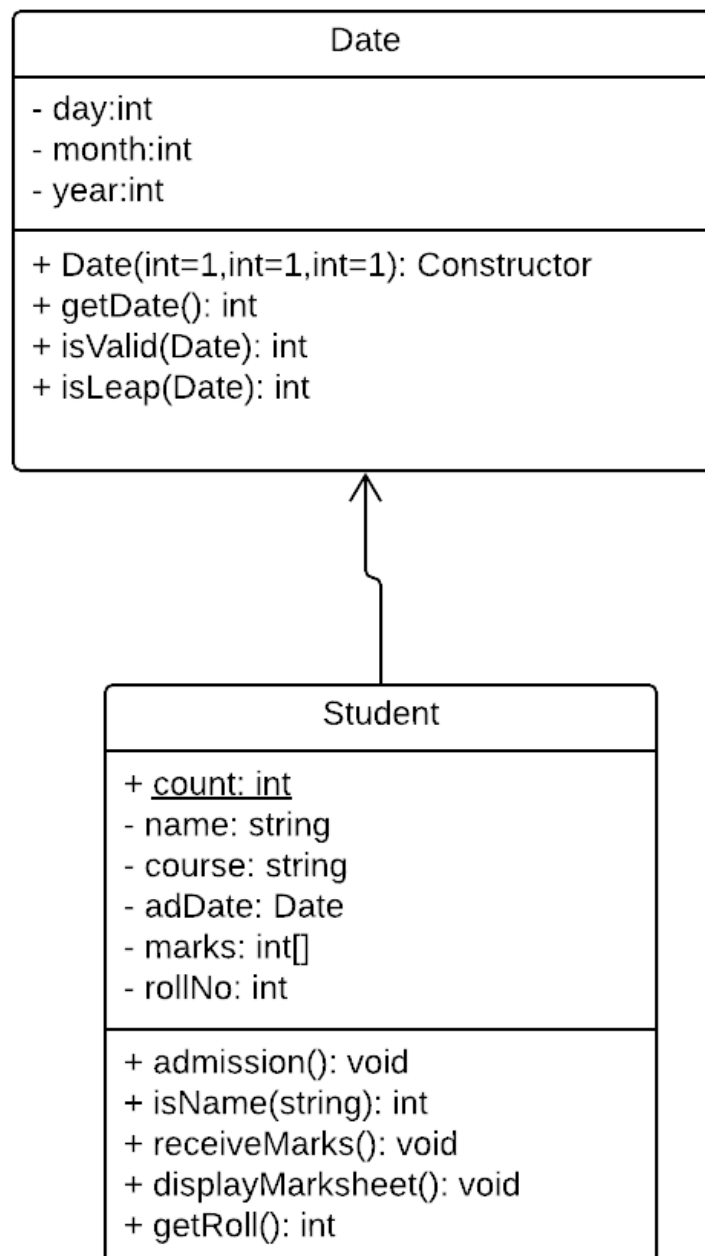
C++ Assignment 1:

PROBLEM STATEMENT:

Design a STUDENT class to store roll, name, course, admission date and marks in 5 subjects. Provide methods corresponding to admission date and receiving marks, preparing mark sheet. Support must be there to show the number of students who have taken admission.

CLASS DIAGRAM:

OUTPUT:

```
1. New Admission
2. Marks-entry
3. Marksheet display
4. Display total no of students
5. Exit
Enter choice
1
Enter name of student:
Ravi

Enter course name: Arts

Enter admission date: Enter date in dd-mm-yyyy format
14-05-2017
Admission successfull
Roll number of new student is: 100001
1. New Admission
2. Marks-entry
3. Marksheet display
4. Display total no of students
5. Exit
Enter choice
4
Total number of students: 1
1. New Admission
2. Marks-entry
3. Marksheet display
4. Display total no of students
5. Exit
Enter choice
1
Enter name of student:
Raj

Enter course name: Science

Enter admission date: Enter date in dd-mm-yyyy format
14-13-2017
Invalid date entered
Enter date in dd-mm-yyyy format
15-04-2018
Admission successfull
Roll number of new student is: 100002
1. New Admission
2. Marks-entry
3. Marksheet display
4. Display total no of students
```

```
5. Exit
Enter choice
2
Enter roll number of student 100001
Enter marks in 5 subjects
90
85
75
96
99
1. New Admission
2. Marks-entry
3. Marksheet display
4. Display total no of students
5. Exit
Enter choice
3
Enter roll number of student 100001
=====================================================================
|       Name :Ravi          | Roll number: 100001           |
---------------------------------------------------------------------
|       SUBJECT             |             MARKS             |
---------------------------------------------------------------------
|       Subject 0           |              90               |
|       Subject 1           |              85               |
|       Subject 2           |              75               |
|       Subject 3           |              96               |
|       Subject 4           |              99               |
---------------------------------------------------------------------
|       Percentage: 89%                                     |
=====================================================================
1. New Admission
2. Marks-entry
3. Marksheet display
4. Display total no of students
5. Exit
Enter choice
4
Total number of students: 2
1. New Admission
2. Marks-entry
3. Marksheet display
4. Display total no of students
5. Exit
Enter choice
5
Quitting
```
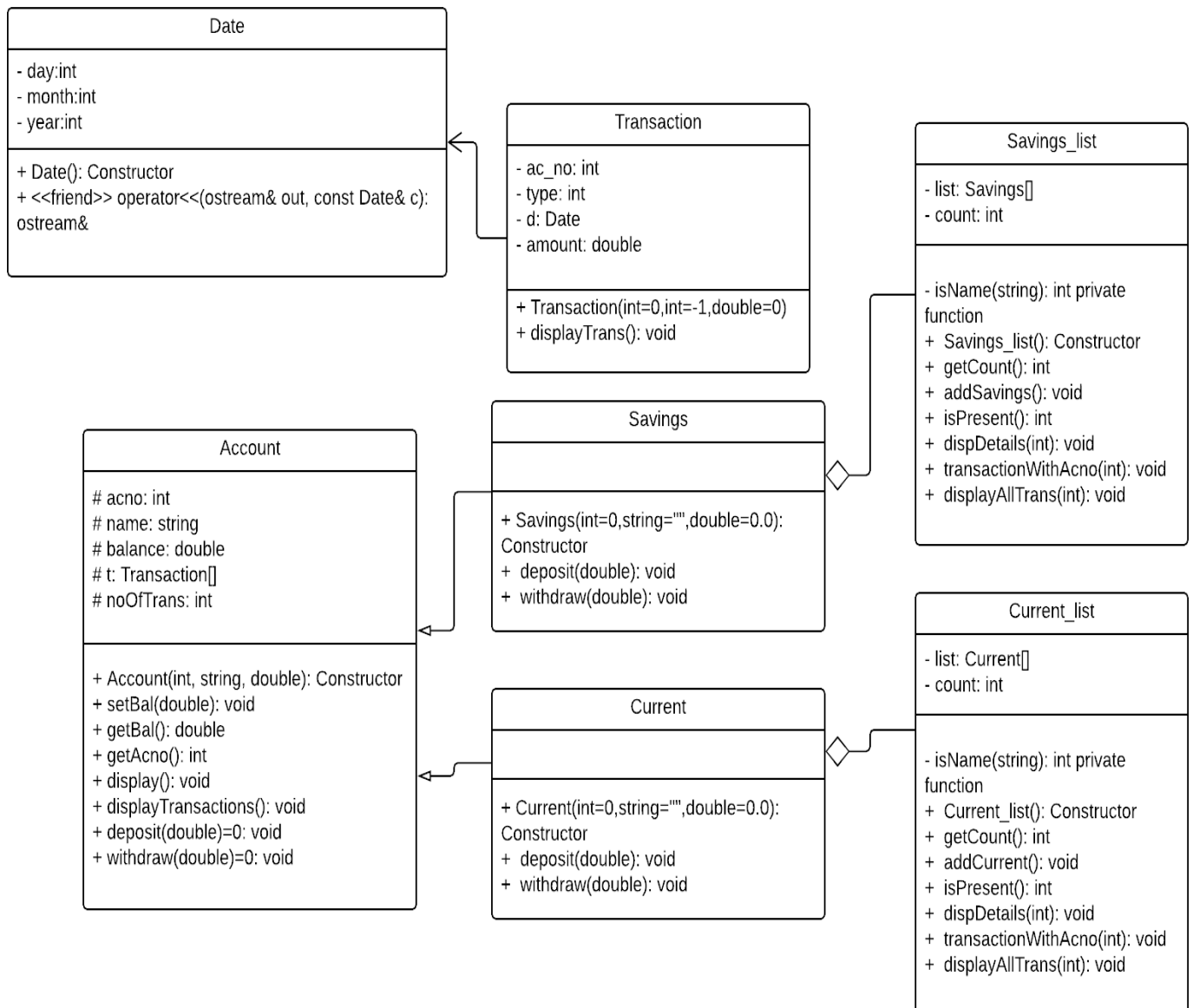
C++ ASSIGNMENT 2:

PROBLEM STATEMENT:

In a bank two types of accounts are there savings and current. For savings account a minimum deposit of Rs. 500 are to be kept. In current account overdraft up to Rs. 20,000 is allowed. Each transaction is noted. Design and implement the necessary classes.

CLASS DIAGRAM:

Date

- day:int
- month:int
- year:int

+ Date(): Constructor
+ <<friend>> operator<<(ostream& out, const Date& c): ostream&

Transaction

- ac_no: int
- type: int
- d: Date
- amount: double

+ Transaction(int=0,int=-1,double=0)
+ displayTrans(): void

Savings_list

- list: Savings[]
- count: int

- isName(string): int private function
+ Savings_list(): Constructor
+ getCount(): int
+ addSavings(): void
+ isPresent(): int
+ dispDetails(int): void
+ transactionWithAcno(int): void
+ displayAllTrans(int): void

Account

acno: int
name: string
balance: double
t: Transaction[]
noOfTrans: int

+ Account(int, string, double): Constructor
+ setBal(double): void
+ getBal(): double
+ getAcno(): int
+ display(): void
+ displayTransactions(): void
+ deposit(double)=0: void
+ withdraw(double)=0: void

Savings

+ Savings(int=0,string="",double=0.0): Constructor
+ deposit(double): void
+ withdraw(double): void

Current

+ Current(int=0,string="",double=0.0): Constructor
+ deposit(double): void
+ withdraw(double): void

Current_list

- list: Current[]
- count: int

- isName(string): int private function
+ Current_list(): Constructor
+ getCount(): int
+ addCurrent(): void
+ isPresent(): int
+ dispDetails(int): void
+ transactionWithAcno(int): void
+ displayAllTrans(int): void

OUTPUT:

```
1. Add new savings account
2. Add new current account
3. Transact with existing savings account
4. Transact with existing current account
5. Display details of existing savings account
6. Display details of existing current account
7. Display transaction details of existing savings account
8. Display transaction details of existing current account
9. Exit
Enter choice
1
Enter name of account holder
Raj
Enter amount to deposit
5000
Your account details are
===================================================

YOUR ACCOUNT DETAILS ARE:
Account number: 10000
Name: Raj
Balance: Rs.5000 (if negative then consider overdraft)
===================================================

Account successfully added
1. Add new savings account
2. Add new current account
3. Transact with existing savings account
4. Transact with existing current account
5. Display details of existing savings account
6. Display details of existing current account
7. Display transaction details of existing savings account
8. Display transaction details of existing current account
9. Exit
Enter choice
3
Enter account number
10000
Enter type of transaction
0. Deposit
1. Withdrawal
0
```

```
Enter amount
4000
Deposit successful
1. Add new savings account
2. Add new current account
3. Transact with existing savings account
4. Transact with existing current account
5. Display details of existing savings account
6. Display details of existing current account
7. Display transaction details of existing savings account
8. Display transaction details of existing current account
9. Exit
Enter choice
5
Enter account number
10000
====================================================

YOUR ACCOUNT DETAILS ARE:
Account number: 10000
Name: Raj
Balance: Rs.9000 (if negative then consider overdraft)
=====================================================

1. Add new savings account
2. Add new current account
3. Transact with existing savings account
4. Transact with existing current account
5. Display details of existing savings account
6. Display details of existing current account
7. Display transaction details of existing savings account
8. Display transaction details of existing current account
9. Exit
Enter choice
3
Enter account number
10000
Enter type of transaction
0. Deposit
1. Withdrawal
1
Enter amount
30000
Cannot withdraw...minimum balance must be maintained
1. Add new savings account
2. Add new current account
```

```
3. Transact with existing savings account
4. Transact with existing current account
5. Display details of existing savings account
6. Display details of existing current account
7. Display transaction details of existing savings account
8. Display transaction details of existing current account
9. Exit
Enter choice
3
Enter account number
10000
Enter type of transaction
0. Deposit
1. Withdrawal
1
Enter amount
3000
Withdraw successful
1. Add new savings account
2. Add new current account
3. Transact with existing savings account
4. Transact with existing current account
5. Display details of existing savings account
6. Display details of existing current account
7. Display transaction details of existing savings account
8. Display transaction details of existing current account
9. Exit
Enter choice
5
Enter account number
10000
===================================================

YOUR ACCOUNT DETAILS ARE:
Account number: 10000
Name: Raj
Balance: Rs.6000 (if negative then consider overdraft)
===================================================

1. Add new savings account
2. Add new current account
3. Transact with existing savings account
4. Transact with existing current account
5. Display details of existing savings account
6. Display details of existing current account
7. Display transaction details of existing savings account
```

```
8. Display transaction details of existing current account
9. Exit
Enter choice
2
Enter name of account holder
Raju
Enter amount to deposit
30000
Your account details are
==================================================

YOUR ACCOUNT DETAILS ARE:
Account number: 20000
Name: Raju
Balance: Rs.30000 (if negative then consider overdraft)
==================================================

Account successfully added
1. Add new savings account
2. Add new current account
3. Transact with existing savings account
4. Transact with existing current account
5. Display details of existing savings account
6. Display details of existing current account
7. Display transaction details of existing savings account
8. Display transaction details of existing current account
9. Exit
Enter choice
4
Enter account number
20000
Enter type of transaction
0. Deposit
1. Withdrawal
1
Enter amount
50000
Withdraw successful
1. Add new savings account
2. Add new current account
3. Transact with existing savings account
4. Transact with existing current account
5. Display details of existing savings account
6. Display details of existing current account
7. Display transaction details of existing savings account
8. Display transaction details of existing current account
```

```
9. Exit
Enter choice
6
Enter account number
20000
====================================================

YOUR ACCOUNT DETAILS ARE:
Account number: 20000
Name: Raju
Balance: Rs.-20000 (if negative then consider overdraft)
=====================================================

1. Add new savings account
2. Add new current account
3. Transact with existing savings account
4. Transact with existing current account
5. Display details of existing savings account
6. Display details of existing current account
7. Display transaction details of existing savings account
8. Display transaction details of existing current account
9. Exit
Enter choice
4
Enter account number
20000
Enter type of transaction
0. Deposit
1. Withdrawal
1
Enter amount
4000
Cannot withdraw..overdraft limit reached
1. Add new savings account
2. Add new current account
3. Transact with existing savings account
4. Transact with existing current account
5. Display details of existing savings account
6. Display details of existing current account
7. Display transaction details of existing savings account
8. Display transaction details of existing current account
9. Exit
Enter choice
9
Quitting
```
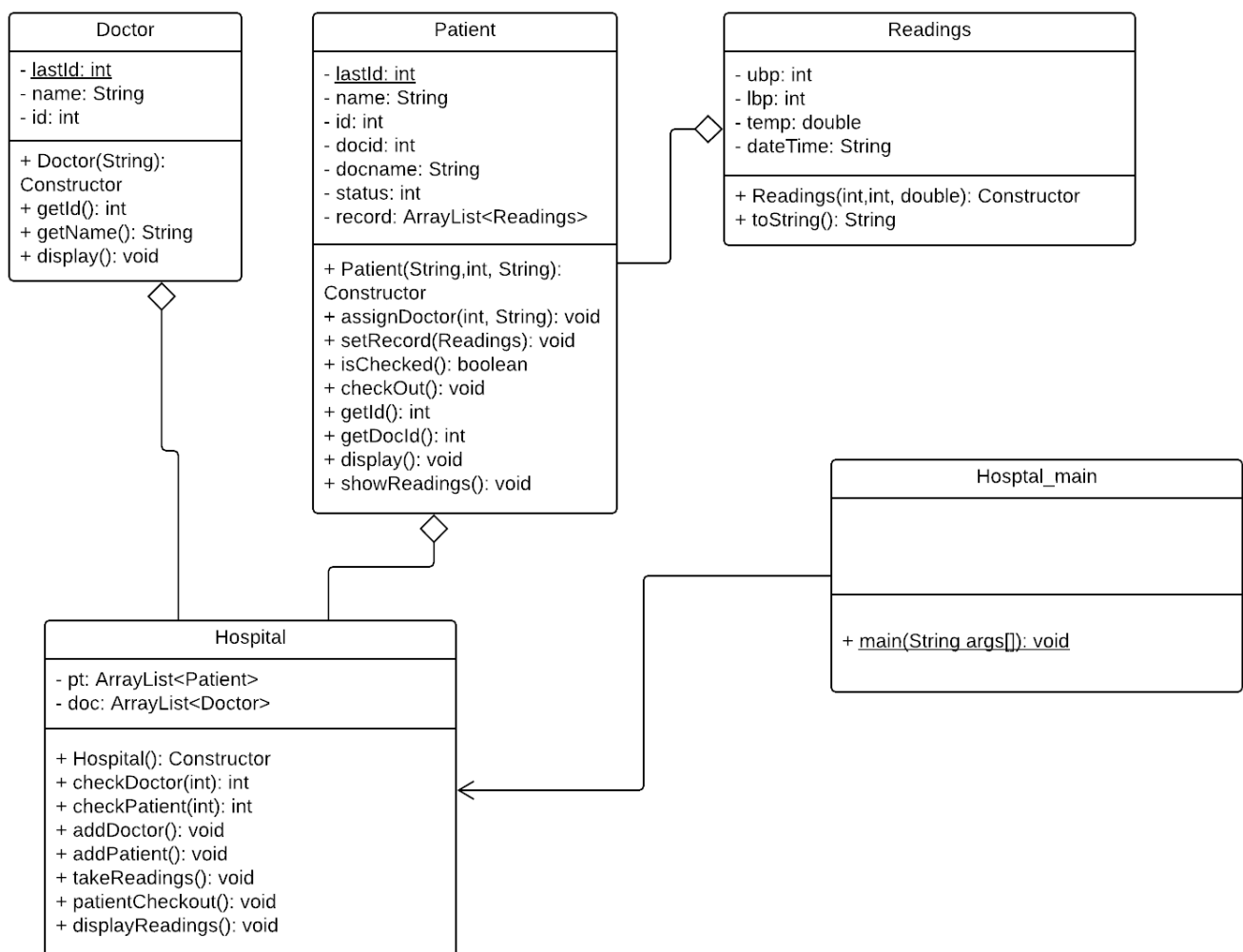
JAVA ASSIGNMENT 1:

PROBLEM STATEMENT:

Design and create a hospital information system with the following scenarios.

● Register a new patient.

● Each patient is assigned to one doctor, but a doctor can have any number of patients. Patients check in to the hospital and assigned a doctor if they don't already have one.

● While in the hospital, doctors record various observations about each patient at various times. Examples of observations are blood pressure and temperature. Record test results for a patient.

● The hospital keeps track of all the observations for a given patient until they check out of the hospital. Obtain all of a patient's information given the social security number.

CLASS DIAGRAM:

Doctor

- lastId: int
- name: String
- id: int

+ Doctor(String): Constructor
+ getId(): int
+ getName(): String
+ display(): void

Patient

- lastId: int
- name: String
- id: int
- docid: int
- docname: String
- status: int
- record: ArrayList<Readings>

+ Patient(String,int, String): Constructor
+ assignDoctor(int, String): void
+ setRecord(Readings): void
+ isChecked(): boolean
+ checkOut(): void
+ getId(): int
+ getDocId(): int
+ display(): void
+ showReadings(): void

Readings

- ubp: int
- lbp: int
- temp: double
- dateTime: String

+ Readings(int,int, double): Constructor
+ toString(): String

Hosptal_main

+ main(String args[]): void

Hospital

- pt: ArrayList<Patient>
- doc: ArrayList<Doctor>

+ Hospital(): Constructor
+ checkDoctor(int): int
+ checkPatient(int): int
+ addDoctor(): void
+ addPatient(): void
+ takeReadings(): void
+ patientCheckout(): void
+ displayReadings(): void

OUTPUT:

```
1. Add a new doctor
2. Add a new patient
3. Take readings
4. Checkout patient
5. Display all readings of a patient
6. Exit
Enter choice
1
Enter name of doctor
Dr. Ravi
Id: 20000
Name: Dr. Ravi
1. Add a new doctor
2. Add a new patient
3. Take readings
4. Checkout patient
5. Display all readings of a patient
6. Exit
Enter choice
1
Enter name of doctor
Dr. Ram
Id: 20001
Name: Dr. Ram
1. Add a new doctor
2. Add a new patient
3. Take readings
4. Checkout patient
5. Display all readings of a patient
6. Exit
Enter choice
2
Enter name of patient
Anu
Enter doctor id to assign to
23
Invalid doctor id
Enter doctor id to assign to
20001
1. Add a new doctor
2. Add a new patient
3. Take readings
4. Checkout patient
5. Display all readings of a patient
6. Exit
Enter choice
3
```

```
Enter patient id
1000
Invalid patient id
Enter patient id
10000
Enter upper and lower blood pressure
90
140
Invalid pressure
Enter upper and lower blood pressure
80
120
Invalid pressure
Enter upper and lower blood pressure
120
80
Enter body temperature in fahrenheit
98
1. Add a new doctor
2. Add a new patient
3. Take readings
4. Checkout patient
5. Display all readings of a patient
6. Exit
Enter choice
5
Enter patient id
10000
Taken by: Dr. Ram
-----------------------------------------------
15 Apr 2018 12:11:21
BP: 120/80
TEMPERATURE: 98.0
----------------------------------------------
1. Add a new doctor
2. Add a new patient

3. Take readings
4. Checkout patient
5. Display all readings of a patient
6. Exit
Enter choice
4
Enter patient id..enter -1 to exit
10000
Patient successfully checked out
1. Add a new doctor
2. Add a new patient
3. Take readings
```

```
4. Checkout patient
5. Display all readings of a patient
6. Exit
Enter choice
4
Enter patient id..enter -1 to exit
10000
Patient already checked out
1. Add a new doctor
2. Add a new patient
3. Take readings
4. Checkout patient
5. Display all readings of a patient
6. Exit
Enter choice
6
Now quitting
```
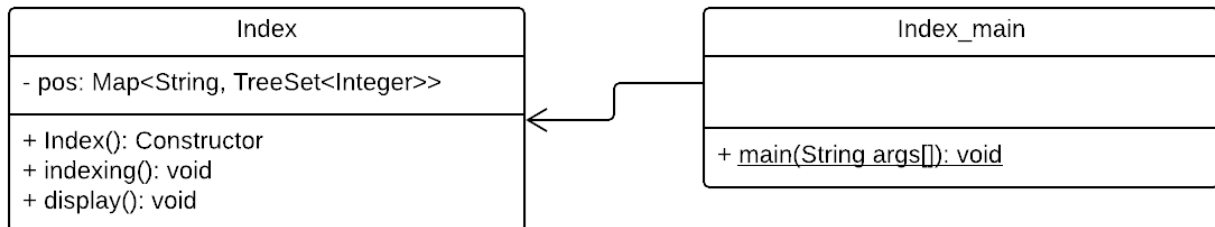
JAVA ASSIGNMENT 2:

PROBLEM STATEMENT:

Indexing a book. Write a program that reads in a text file from standard input and compiles an alphabetical index of which words appear on which lines, as in the following input. Ignore case and punctuation. For each word maintain a list of location on which it appears. Try to use HashTable and/or HashMap class (of java.util).

CLASS DIAGRAM:

Index
- pos: Map<String, TreeSet<Integer>>
+ Index(): Constructor + indexing(): void + display(): void

Index_main
+ main(String args[]): void

OUTPUT:

Given the following file as input:

Now now indulgence dissimilar for his thoroughly has terminated. Agreement offending commanded my an. Change wholly say why eldest period. Are projection put celebrated particular unreserved joy unsatiable its. In then dare good am rose bred or. On am in nearer square wanted.

Are own design entire former get should. Advantages boisterous day excellence boy. Out between our two waiting wishing. Pursuit he he garrets greater towards amiable so placing. Nothing off how norland delight. Abode shy shade she hours forth its use. Up whole of fancy ye quiet do. Justice fortune no to is if winding morning forming. now

```
Enter name of file to read
problem5/test.txt
Invalid file
Enter name of file to read
D:\Eclipseprojects\problem5\src\problem5\test.txt
: [5]
abode: [9]
advantages: [6]
agreement: [1]
am: [4]
amiable: [8]
an: [2]
are: [2, 6]
between: [7]
boisterous: [6]
boy: [7]
bred: [4]
celebrated: [3]
change: [2]
commanded: [2]
dare: [4]
day: [6]
delight: [9]
design: [6]
dissimilar: [1]
```

do: [10]
eldest: [2]
entire: [6]
excellence: [7]
fancy: [10]
for: [1]
former: [6]
forming: [11]
forth: [9]
fortune: [10]
garrets: [8]
get: [6]
good: [4]
greater: [8]
has: [1]
he: [7]
his: [1]
hours: [9]
how: [9]
if: [10]
in: [4]
indulgence: [1]
is: [10]
its: [3, 9]
joy: [3]
justice: [10]
morning: [11]
my: [2]
nearer: [4]
no: [10]
norland: [9]
nothing: [8]
now: [1, 11]
of: [10]
off: [8]
offending: [2]
on: [4]
or: [4]
our: [7]
out: [7]
own: [6]
particular: [3]
period: [2]
placing: [8]
projection: [3]
pursuit: [7]
put: [3]
quiet: [10]
rose: [4]

```
say: [2]
shade: [9]
she: [9]
should: [6]
shy: [9]
so: [8]
square: [4]
terminated: [1]
then: [4]
thoroughly: [1]
to: [10]
towards: [8]
two: [7]
unreserved: [3]
unsatiable: [3]
up: [10]
use: [9]
waiting: [7]
wanted: [4]
whole: [10]
wholly: [2]
why: [2]
winding: [10]
wishing: [7]
ye: [10]
```