GTU Department of Computer Engineering

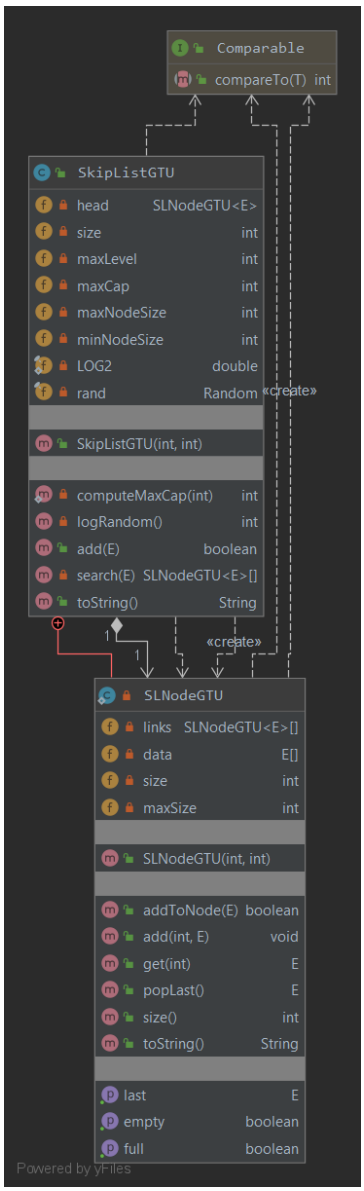
CSE 222/505 - Spring 2020

Homework 7

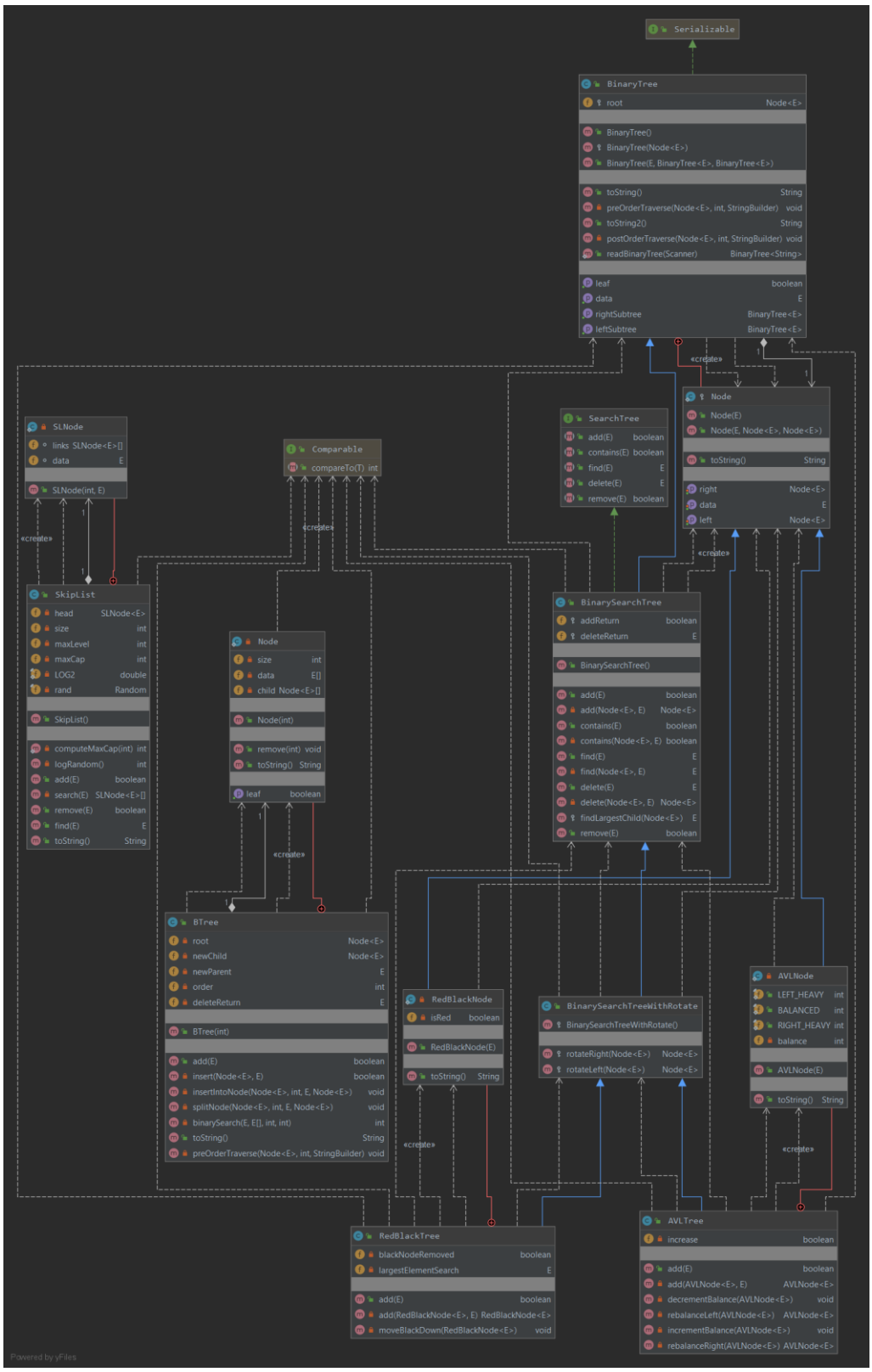
Report

Ali Bahar-171044066

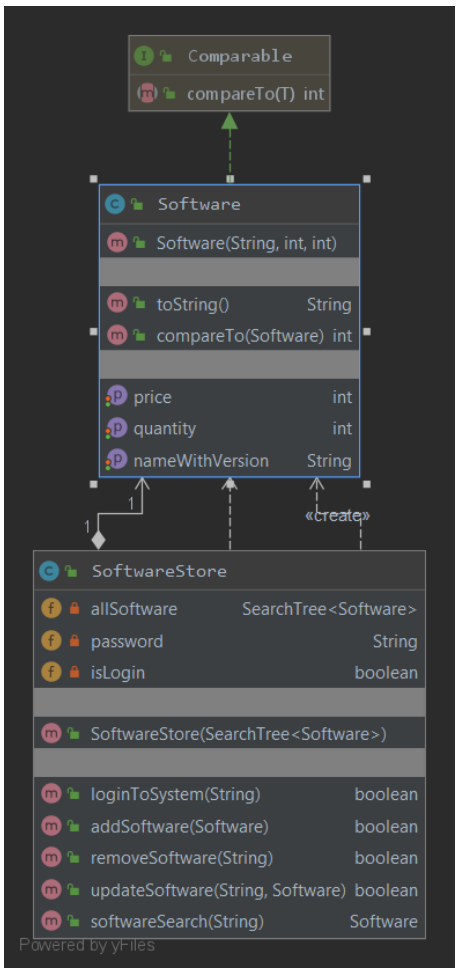
Class Diagram-Q2



Class Diagram-Q3



Class Diagram-Q4



Problem Solution Approach-Q2

I have just implemented the add method. This add method inserts the data properly . This data structure grows from end so any node can not be smaller than the min value. If the node is full then this method try to insert to next node.

Problem Solution Approach-Q3

We have compared performance of some data structures. I have implemeted the these data structures as described in book. but there was some missing methods like remove so I had to implement these methods. After implementation I have tested these method as described in homework.

Problem Solution Approach-Q4

We have implemented a simple software store system. I have software class to be able to hold the datas of a software and I have SofwareStore class to handle some opeartions like adding, removing, updating. I have kept the data in a search tree . I have taken a search tree as parameter of constructor because I can not initialize a interface. I have added the datas by comparing its name so there is not be able to two software which have same name.

Test Cases-Q2

Test ID	Scenario	Test Data	Expected Result	Actual Result	Pass /Fail
1	Add some random data and print it.	Max = 4 Min = 2 Data = 100,50,150,125,25,15,10,-1,30,130,151,192,-34,123,152,-152	All datas are added to list and printed properly.	As expected	Pass

Test Cases-Q3

Test ID	Scenario	Test Data	Expected Result	Actual Result	Pass /Fail
1	Test the regular binary search tree. 1)Add some roandom data 2) add extra 10 random data and measure the run time. 3) perform 10 random successfull deletion operation and measure the run time. 4) repeat this scenario ten times.	10000 random data 20000 random data 40000 random data 80000 random data	All elements are added to tree. 10 random successfull deletion operarion is performed.	As expected	Pass
2	Test the red black tree implementation in book. 1)Add some roandom data 2) add extra 10 random data and measure the run time. 3) perform 10 random successfull deletion operation and measure the run time. 4) repeat this scenario ten times.	10000 random data 20000 random data 40000 random data 80000 random data	All elements are added to tree. 10 random successfull deletion operarion is performed.	There is no remove implementation so deletion operations are not successfull.	Fail
3	Test the red black tree implementation in java(TreeSet). 1)Add some roandom data 2) add extra 10 random data and measure the run time. 3) perform 10 random successfull deletion operation and measure the run time.	10000 random data 20000 random data 40000 random data 80000 random data	All elements are added to tree. 10 random successfull deletion operarion is performed.	As expected	Pass

	4) repeat this scenario ten times.				
4	Test the BTree implementation in the book. 1)Add some roandom data 2) add extra 10 random data and measure the run time. 3) perform 10 random successfull deletion operation and measure the run time. 4) repeat this scenario ten times.	10000 random data 20000 random data 40000 random data 80000 random data	All elements are added to tree. 10 random successfull deletion operarion is performed.	There is no remove implementation so deletion operations are not successfull	Fail
5	Test the SkipList implementation in the book. 1)Add some roandom data 2) add extra 10 random data and measure the run time. 3) perform 10 random successfull deletion operation and measure the run time. 4) repeat this scenario ten times.	10000 random data 20000 random data 40000 random data 80000 random data	All elements are added to tree. 10 random successfull deletion operarion is performed.	As expected	Pass
6	Test the SkipList implementation in java (concurrentSkipList). 1)Add some roandom data 2) add extra 10 random data and measure the run time. 3) perform 10 random successfull deletion operation	10000 random data 20000 random data 40000 random data 80000 random data	All elements are added to tree. 10 random successfull deletion operarion is performed.	As expected	Pass

	and measure the run time. 4) repeat this scenario ten times.				
7	Test the SkipList implementation in question-2 1)Add some roandom data 2) add extra 10 random data and measure the run time. 3) perform 10 random successfull deletion operation and measure the run time. 4) repeat this scenario ten times.	10000 random data 20000 random data 40000 random data 80000 random data	All elements are added to tree. 10 random successfull deletion operarion is performed.	There is no remove implementation so deletion operations are not successfull. Add operation takes too time.	Fail

Test Cases-Q4

Test ID	Scenario	Test Data	Expected Result	Actual Result	Pass /Fail
1	Search software with its name	DataName = “Norton5.5”	Prints all information about the book.	As expected	Pass
2	Login to system as admin with wrong password	Password = “123” Actual password=”171044066”	Prints “login fail”	As expected	pass
3	Login to system as adimin with correct password	Password = “171044066”	Admin panel is opend.	As expected	Pass
4	1)remove a book 2)search the removed bokk.	“norton5.5”	The book is removed and search returned null.	As expected	Pass
5	1)add a new book 2)search the new book	Name = “myStore” Price = 1905	The book is added. The all information about book is printed.	As expected	Pass

Running command and results-Q2

[-152, -34, -1, 10] [15, 25, 30, 50] [100, 123, 125, 130] [150, 151, 152, 192]

[-152, -34, -1, 10] [100, 123, 125, 130]

Running command and results-Q3

(Regular Binary Search Tree)

problem size : 10000

Average running time (insertion): 14090ns

Average running time (remove): 94670ns

(Regular Binary Search Tree)

problem size : 20000

Average running time (insertion): 15540ns

Average running time (remove): 44220ns

(Regular Binary Search Tree)

problem size : 40000

Average running time (insertion): 8870ns

Average running time (remove): 50990ns

(Regular Binary Search Tree)

problem size : 80000

Average running time (insertion): 9640ns

Average running time (remove): 31310ns

(Red black tree implementation in the book)

problem size : 10000

Average running time (insertion): 12220ns

(Red black tree implementation in the book)

problem size : 20000

Average running time (insertion): 11880ns

(Red black tree implementation in the book)

problem size : 40000

Average running time (insertion): 26900ns

(Red black tree implementation in the book)

problem size : 80000

Average running time (insertion): 14580ns

(Tree set(Red black tree implementation in java))

problem size : 10000

Average running time (insertion): 16710ns

Average running time (remove): 78970ns

(Tree set(Red black tree implementation in java))

problem size : 20000

Average running time (insertion): 27830ns

Average running time (remove): 76490ns

(Tree set(Red black tree implementation in java))

problem size : 40000

Average running time (insertion): 8070ns

Average running time (remove): 76690ns

(Tree set(Red black tree implementation in java))

problem size : 80000

Average running time (insertion): 17000ns

Average running time (remove): 60790ns

BTree

problem size : 10000

Average running time (insertion): 27280ns

BTree

problem size : 20000

Average running time (insertion): 25340ns

BTree

problem size : 40000

Average running time (insertion): 28350ns

BTree

problem size : 80000

Average running time (insertion): 32630ns

Skip List

problem size : 10000

Average running time (insertion): 21900ns

Average running time (remove): 46020ns

Skip List

problem size : 20000

Average running time (insertion): 22980ns

Average running time (remove): 54730ns

Skip List

problem size : 40000

Average running time (insertion): 25660ns

Average running time (remove): 45810ns

Skip List

problem size : 80000

Average running time (insertion): 26710ns

Average running time (remove): 37220ns

concurrentSkipListSet(skip list implementation in java)

problem size : 10000

Average running time (insertion): 38240ns

Average running time (remove): 255820ns

concurrentSkipListSet(skip list implementation in java)

problem size : 20000

Average running time (insertion): 20150ns

Average running time (remove): 152660ns

concurrentSkipListSet(skip list implementation in java)

problem size : 40000

Average running time (insertion): 23910ns

Average running time (remove): 166440ns

concurrentSkipListSet(skip list implementation in java)

problem size : 80000

Average running time (insertion): 17510ns

Average running time (remove): 63520ns

Process finished with exit code 0

Running command and results-Q4

Software Store System

1) Login to system as admin

2) Search Book by name

0) Quit

2

Enter the name of the software

Norton5.5

Name : Norton5.5

quantity : 1

price : 103

Software Store System

1) Login to system as admin

2) Search Book by name

0) Quit

1

Password : 123

login fail

Software Store System

1) Login to system as admin

2) Search Book by name

0) Quit

1

Password : 171044066

Login success

Admin Interface

1) add software

2) delete software

3) update software

2

Enter the name of the new software to be removed

Norton5.5

Software Store System

1) Login to system as admin

2) Search Book by name

0) Quit

2

Enter the name of the software

Norton5.5

null

Software Store System

1) Login to system as admin

2) Search Book by name

0) Quit

1

Password : 171044066

Login success

Admin Interface

1) add software

2) delete software

3) update software

1

Enter the name of the new software to be added

myStore

Enter the price of the new software to be added

1905

Software Store System

1) Login to system as admin

2) Search Book by name

0) Quit

2

Enter the name of the software

myStore

Name : myStore

quantity : 1

price : 1905

Software Store System

1) Login to system as admin

2) Search Book by name

0) Quit

0

Process finished with exit code 0