ADVANCED DATA STRUCTURES

COP5536 FALL 2017

Instructor: Dr. Sartaj Sahni

Programming Project

(Implementation of Initialize, Insert and Search operations in B+ Tree)

Name: Sandesh Joshi

UFID: 4419-5831

UF Email: sandeshjoshi@ufl.edu

INDEX:

Associated Files

- treesearch.java: This main method resides in this class. All the input operations are read and invoked from this class through scanning the input file.
- bPlusTree.java: This file contains class bPlusTree which implements all the essential B+ Tree methods. The methods includes : Initialize, insertData, searchValue(param), searchValue(param1, param2).
- genericNode.java: A generic Node class with instances containing an arraylist of keys and a boolean to check if it's a leaf node.
- InnerNode.java: This file contains class which extends genericNode. It handles the insertion of keys in the inner node and keeps them in sorted order.
- leaf.java: This file contains the class which implements the leaf nodes. It also extends genericNode and adds another arraylist of values to store values corresponding to keys.
- MyDict.java: This file contains class to implement our own key/node pair data structure.
- makefile: This file helps with compilation of code.

Function Prototypes and Program Structure

The class **bPlusTree.java** includes the following instance and methods needed for the implementation of the given project assignment:

Instance variables:

1. genericNode rootNode
2. int O

Methods:

1. Initialize():
    - Description : Takes order input from the file and initialize that tree.
    - Parameters : None
    - Return value : void

2. insertData(parameter1, parameter2):
    - Description : Inserts value corresponding to a given key. Also works for duplicate keys.
    - Parameters : Double key, String value
    - Return value : void

3. searchValue(parameter):
    - Description : It is a method to find value corresponding to a single key. If the key is not present, it returns a null.
    - Parameters : Double key
    - Return value : String value

4. searchValue(parameter1, parameter2):
    - Description : This method takes two keys as the argument and gives all the key value pairs in that range of keys.
    - Parameters : Double key1, Double key2
    - Return value : ArrayList<String> valueList

5. fetchTree(parameter1, parameter2):
    - Description : Returns a leaf the key is pointing
    - Parameters : genericNode node, Double key
    - Return value : genericNode node

6. fetchChild(parameter1, parameter2, parameter3):

- Description :
- Parameters : genericNode node, myDict pair, myDict newChildpair
- Return value : myDict key/node

7. splitLeaf(parameter1):
    - Description : This method splits a leaf.
    - Parameters: leaf leafNode
    - Return value: myDict key/node

8. splitInnerNode(parameter1):
    - Description : This method splits an inner node.
    - Parameters: innerNode inner
    - Return value: myDict key/node

The file **genericNode.java** includes the following instance and methods needed for the implementation of the given project assignment:

Instance variables:

1. boolean LeafNodeCheck
2. ArrayList<Double> keyList

Methods:

1. OverflowCheck():
    - Description : Checks for node overflow
    - Parameters : None
    - Return value : boolean

The file **leaf.java** includes the following instance and methods needed for the implementation of the given project assignment:

Instance variables:

1. ArrayList<String> valueList
2. leaf leafNext
3. leaf leafPrevious

Methods:

1. InsertAndSort():
   - Description : This method inserts key/value pair into the leaf and keeps it sorted
   - Parameters : Double key, String Value
   - Return value : void

The file **innerNode.java** includes the following instance and methods needed for the implementation of the given project assignment:

Instance variables:

1. boolean LeafNodeCheck
2. ArrayList<Double> keyList

Methods:

1. InsertAndSort(parameter1, parameter2):
   - Description : insert the entry into this node at the specified index so that it still remain sorted.
   - Parameters : myDict dict, int index
   - Return value : void

The file **myDict.java** includes the following instance and methods needed for the implementation of the given project assignment:

Instance variables:

1. Double key
2. genericNode Node

Methods:

1. getKey():
   - Description : Get the key from the myDict object
   - Parameters : None
   - Return value : Double key
2. getNode():
   - Description : Get the Node from the myDict object
   - Parameters : None

- Return value : genericNode Node

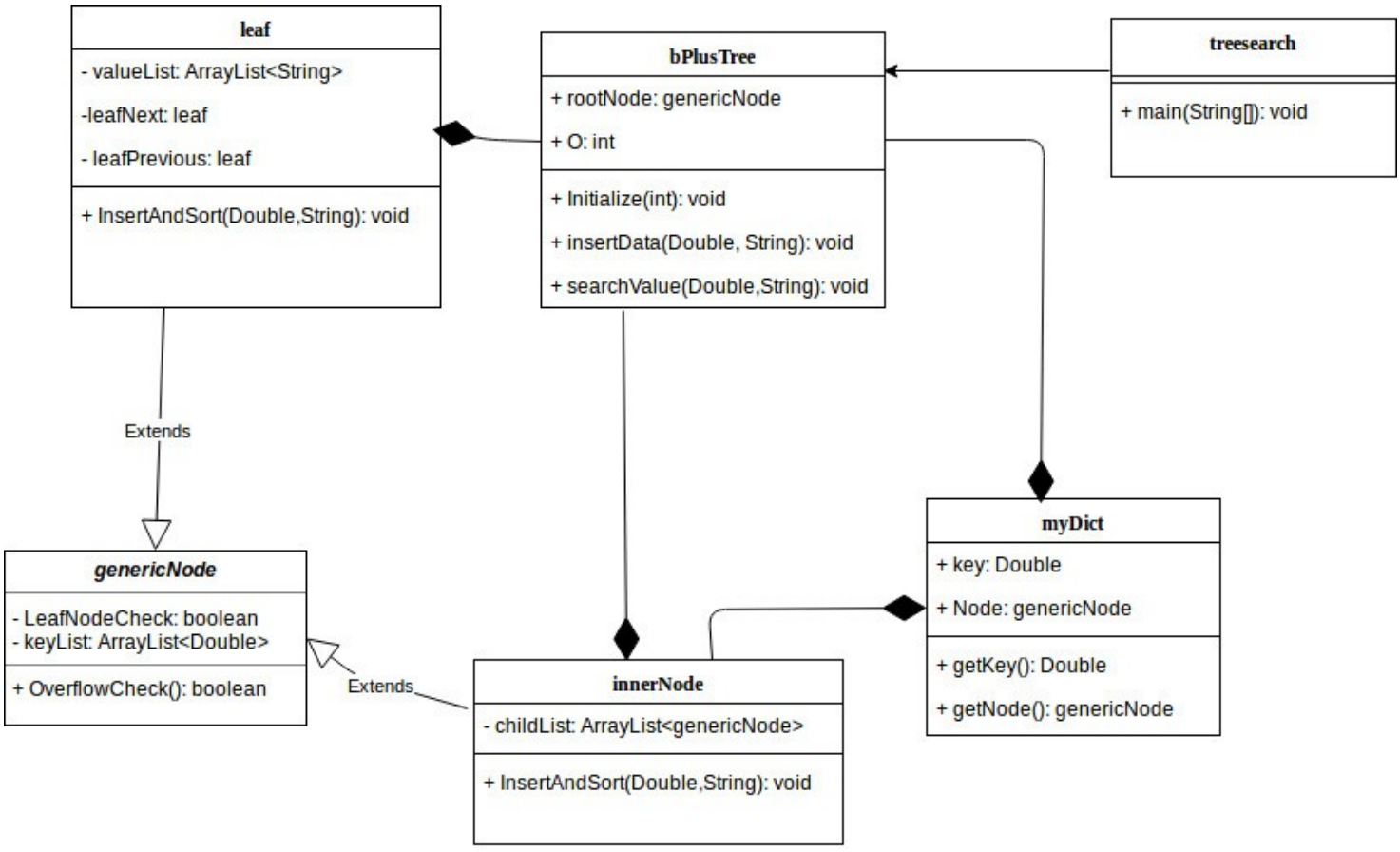
The file **treesearch.java** includes the following instance and methods needed for the implementation of the given project assignment:

Instance variables: None

Methods:

1. main():
   - Description : Main method that invokes all the methods from input file.
   - Parameters : String[] args
   - Return value : void

Program Structure:



Running the Program

The project has been compiled and tested on the Ubuntu 16.04 platform using the javac compiler.

It has been run on the thunder.cise.ufl.edu remote server to test its readiness before final submission.

Steps taken to execute the project after gaining access to the remote server are as follows:

1. Move all the files into a directory on your server.

2. Use ‘cd /path’ to change directory to the files location.

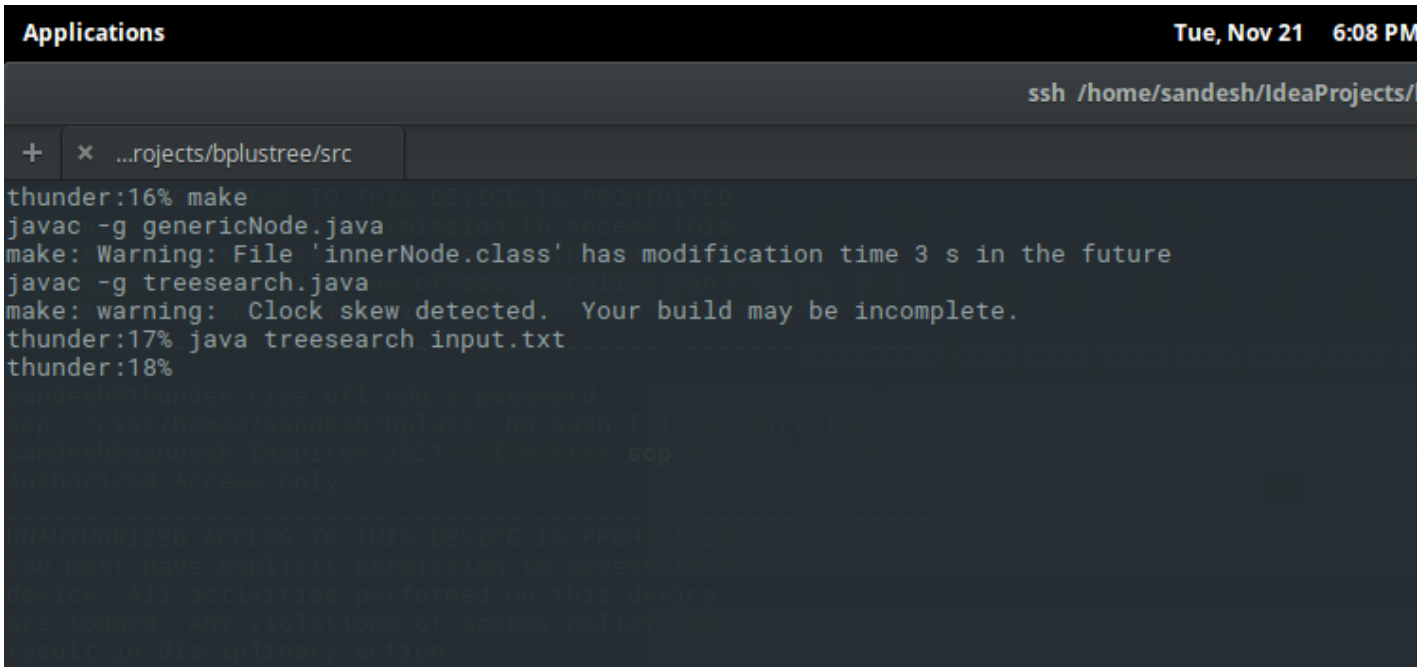
3. Execute the ‘make’ command to compile all the files.

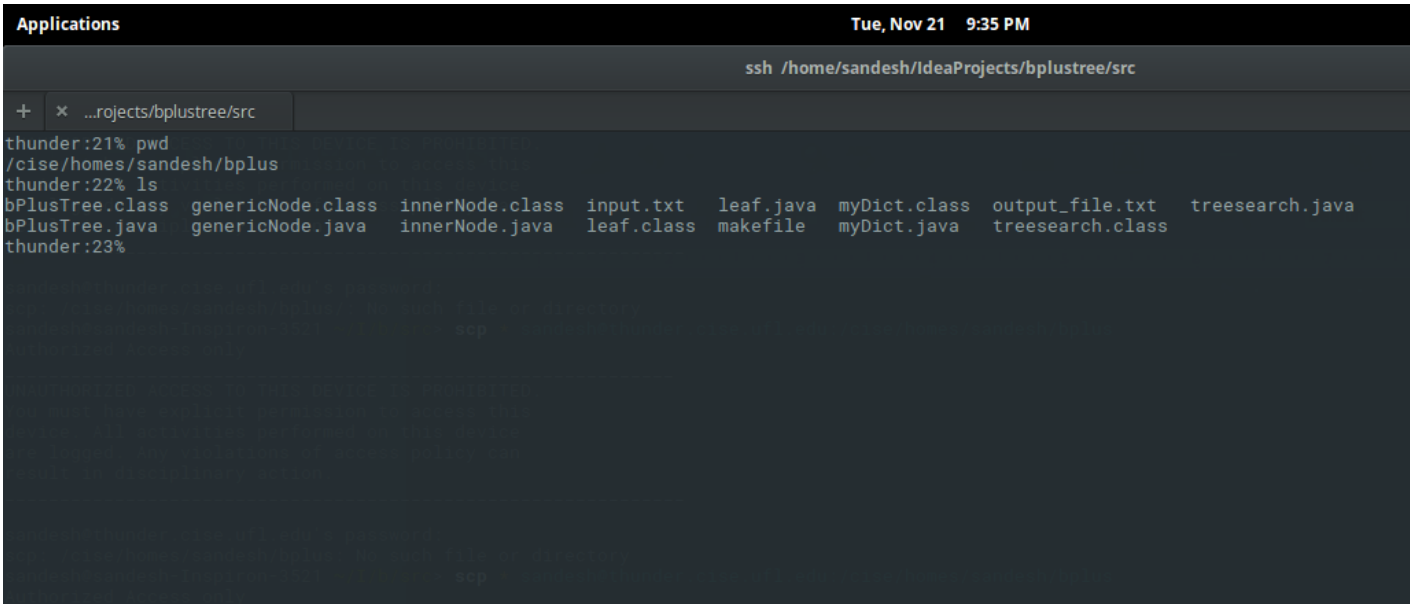
4. Run java treesearch input_filename.txt

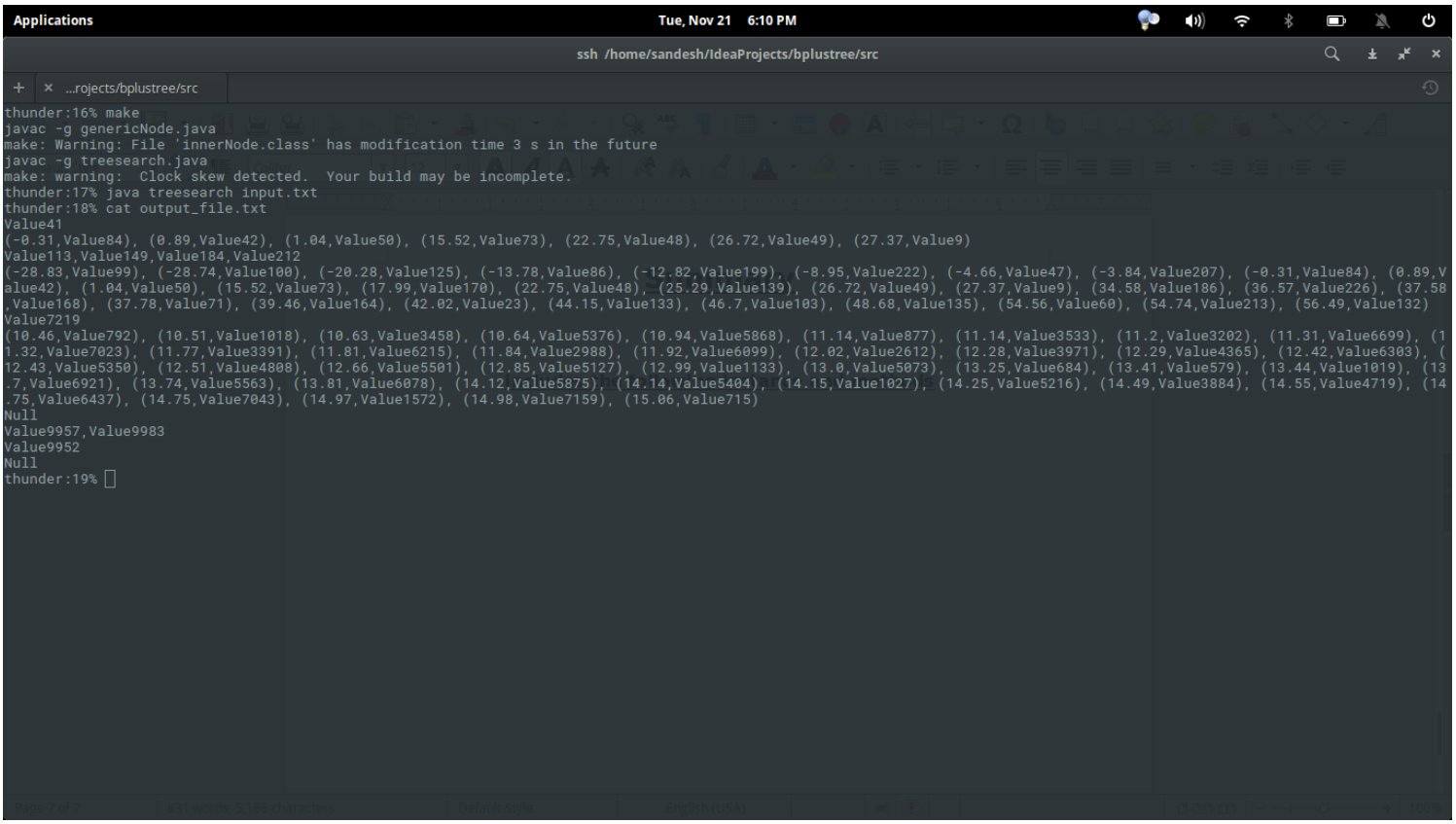
5. The output is generated as the ‘output_file.txt’ file

Summary

The program is successfully compiled and generates the desired output for the given input file






Applications
Tue, Nov 21 6:10 PM
ssh /home/sandesh/IdeaProjects/bplustree/src
...rojects/bplustree/src
thunder:16% make
javac -g genericNode.java
make: Warning: File 'innerNode.class' has modification time 3 s in the future
javac -g treesearch.java
make: warning: Clock skew detected. Your build may be incomplete.
thunder:17% java treesearch input.txt
thunder:18% cat output_file.txt
Value41
(-0.31,Value84), (0.89,Value42), (1.04,Value50), (15.52,Value73), (22.75,Value48), (26.72,Value49), (27.37,Value9)
Value113,Value149,Value184,Value212
(-28.83,Value99), (-28.74,Value100), (-20.28,Value125), (-13.78,Value86), (-12.82,Value199), (-8.95,Value222), (-4.66,Value47), (-3.84,Value207), (-0.31,Value84), (0.89,Value42), (1.04,Value50), (15.52,Value73), (17.99,Value170), (22.75,Value48), (25.29,Value139), (26.72,Value49), (27.37,Value9), (34.58,Value186), (36.57,Value226), (37.58,Value168), (37.78,Value71), (39.46,Value164), (42.02,Value23), (44.15,Value133), (46.7,Value103), (48.68,Value135), (54.56,Value60), (54.74,Value213), (56.49,Value132)
Value7219
(10.46,Value792), (10.51,Value1018), (10.63,Value3458), (10.64,Value5376), (10.94,Value5868), (11.14,Value877), (11.14,Value3533), (11.2,Value3202), (11.31,Value6699), (11.32,Value7023), (11.77,Value3391), (11.81,Value6215), (11.84,Value2988), (11.92,Value6099), (12.02,Value2612), (12.28,Value3971), (12.29,Value4365), (12.42,Value6303), (12.43,Value5350), (12.51,Value4808), (12.66,Value5501), (12.85,Value5127), (12.99,Value1133), (13.0,Value5073), (13.25,Value684), (13.41,Value579), (13.44,Value1019), (13.7,Value6921), (13.74,Value5563), (13.81,Value6078), (14.12,Value5875), (14.14,Value5404), (14.15,Value1027), (14.25,Value5216), (14.49,Value3884), (14.55,Value4719), (14.75,Value6437), (14.75,Value7043), (14.97,Value1572), (14.98,Value7159), (15.06,Value715)
Null
Value9957,Value9983
Value9952
Null
thunder:19%