CSE 222 HOMEWORK #6

Emre

BAYRAM

141044019

TABLE OF CONTENTS

1. Requirements

1.1. Overall Description

Part1

Implementing Encode Method it require 2 parameters which these are message which is going to encode, huffmanTree which is used for create a code using this parameter.

Part2

Implementing iterator class for BinarySearchTree class which is given in source files. This iterator class traverses ascending order.

Part3

Implementing Priority Queue class using Our Queue interface which same as Java's queue interface. It stores any type of data. When creating order uses compartor given when constructing if it is not given it uses one of the elements compareTo Method and orders bigger to less. Lastly it uses arrayList data structure . After implementing that class created Test class written to create Table of benchmark of work time.

1.2. Requirement Definitions

1.2.1. Huffman data when creating Huffman Binary Tree

1.2.2. An entries for Priority Queue

2. Analysis and Solutions Approach

PART 1

Huffman tree is a Binary Tree so generally we use recursion in Binary Tree. It simple to implement. So that I used Recursion thinking . For every character in message traversed tree and one by one created encoded the message.

PART 2

Best way to traverse is using Stack data structure with iterator. So ı used Stack for store the Tree. Again best way to reach elements is recursion technique When Building a Stack. So I used recursion.

PART 3

Priority Queue is like List. There is only one difference between list. This is just like ordered List actually elements has Priority in Queue. So I used ArrayList Data structure when implementing.

3. Class Diagrams

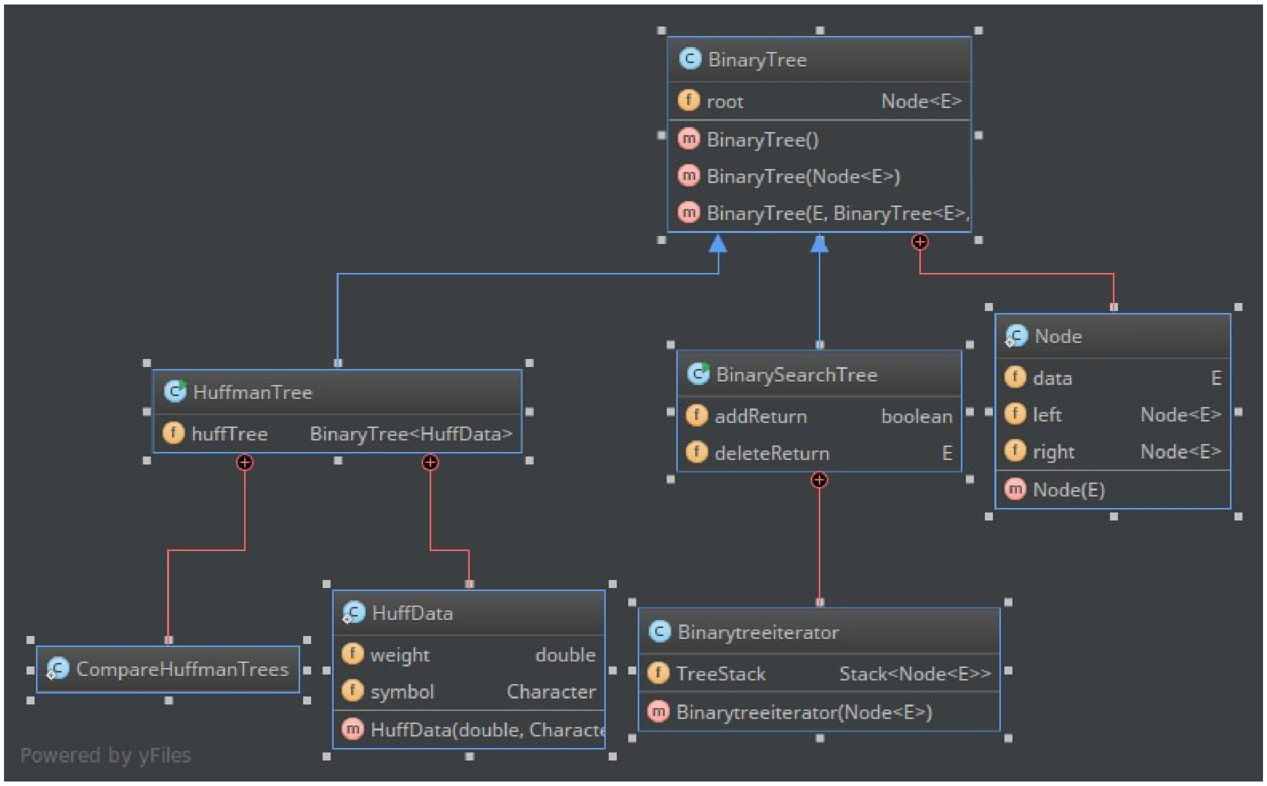


Figure 1 Binary Tree and Huffman Tree Class Diagrams

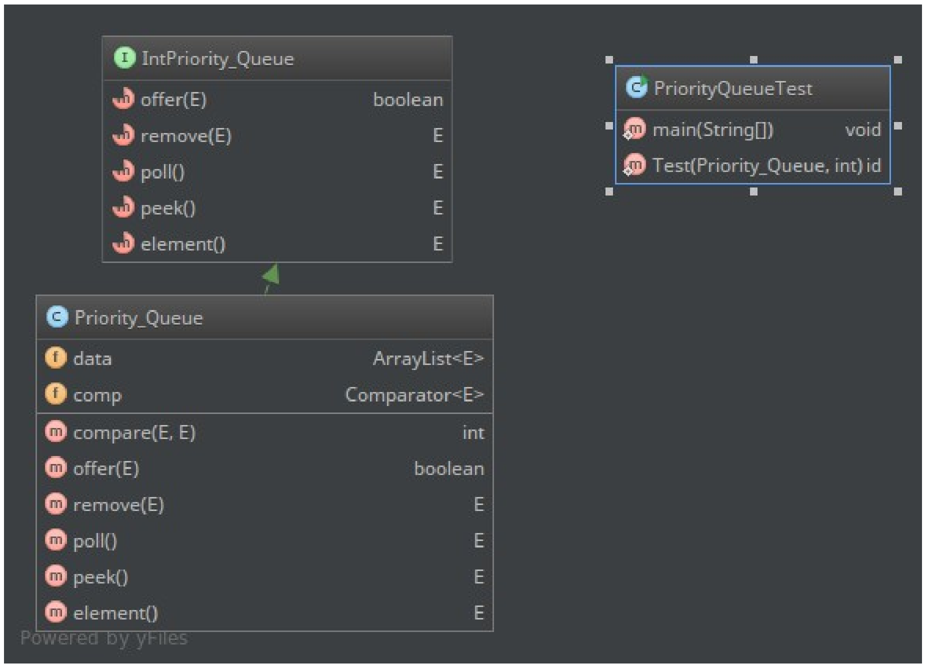
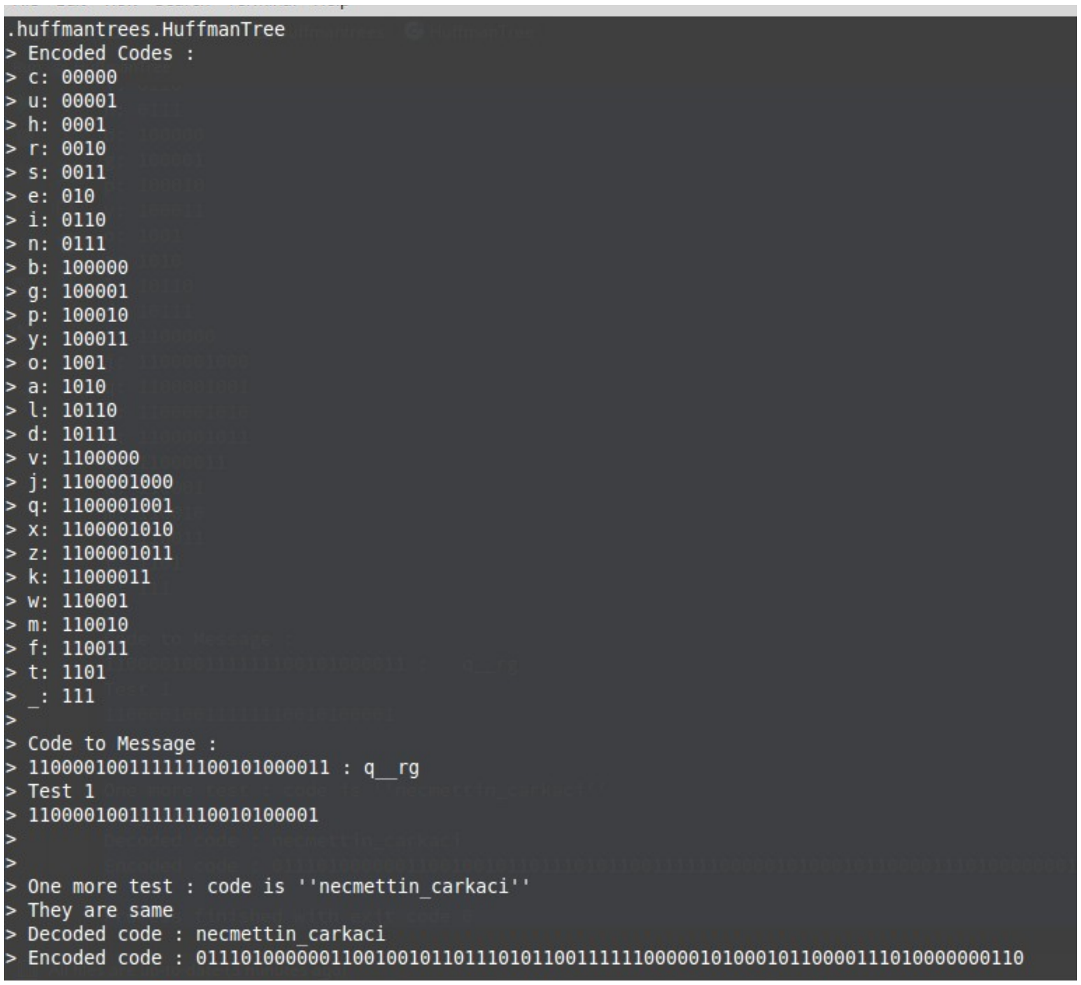


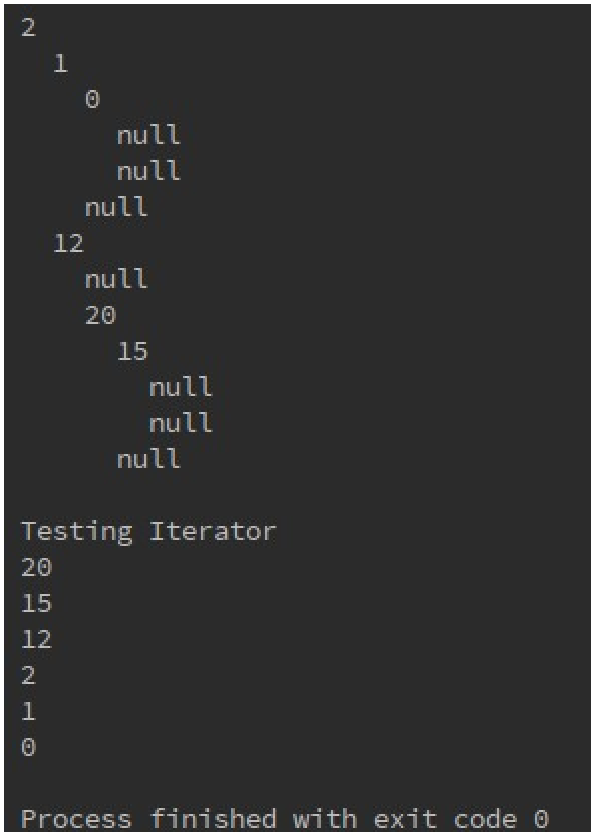
Figure 2 Priority Queue Class Diagram

4. Tests



```
.huffmantrees.HuffmanTree
> Encoded Codes :
> c: 00000
> u: 00001
> h: 0001
> r: 0010
> s: 0011
> e: 010
> i: 0110
> n: 0111
> b: 100000
> g: 100001
> p: 100010
> y: 100011
> o: 1001
> a: 1010
> l: 10110
> d: 10111
> v: 1100000
> j: 1100001000
> q: 1100001001
> x: 1100001010
> z: 1100001011
> k: 11000011
> w: 110001
> m: 110010
> f: 110011
> t: 1101
> _: 111
>
> Code to Message :
> 11000010011111100101000011 : q__rg
> Test 1
> 1100001001111110010100001
>
>
> One more test : code is ''necmettin_carkaci''
> They are same
> Decoded code : necmettin_carkaci
> Encoded code : 0111010000001100100101101110101100111111000001010001011000011101000000110
```

Part 1 Test Results.
 Everything is ok.



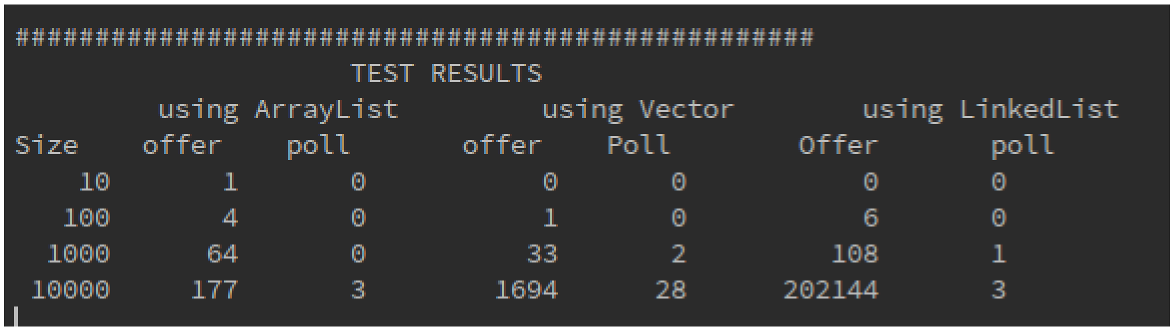
```
2
  1
    0
      null
      null
    null
  12
    null
    20
      15
        null
        null
      null

Testing Iterator
20
15
12
2
1
0

Process finished with exit code 0
```

Part 2 Test Result

Everything is ok.



```
##################################################
                     TEST RESULTS
         using ArrayList        using Vector       using LinkedList
Size    offer    poll        offer     Poll         Offer       poll
    10       1       0           0        0            0        0
   100       4       0           1        0            6        0
  1000      64       0          33        2          108        1
 10000     177       3        1694       28       202144        3
```

Part 3 Test Results

Everything is ok.

5.Result

As a result I challenged part1. And I couldn't implement the Priority Queue with Binary Tree Because of my other Homework. This homework improve my recursion ability. Linked List class in The test squence take too much time with 100000 test size so I cut it. Vector and arrayList nearly same but linkedList takes too much time.