**CSE222\_HW09\_Part\_1**

**ERCAN UCA 091044011**

* **Matrix Graph** Classımda
* Veri tipi double[][];
* Abtract class extend ediliyor.

private double[][] edges;

/\*\*

\* Constructor with 2 parameters

\*

\* @param numV point of matrix

\* @param directed directed control

\*/

public MatrixGraph(int numV, boolean directed) {

super(numV, directed);

...

}

/\*\*

\* This method insert new edge

\* @param edge this edge will add

\*/

@Override

public void insert(Edge edge)

/\*\*

\* This method control edge is an edge, controlling with source and

\* destination

\* @param source source of edge

\* @param dest destination of edge

\* @return True if is an edge, otherwise false.

\*/

@Override

public boolean isEdge(int source, int dest)

/\*\*

\* This method return an edge when know edge info.

\* @param source source of edge

\* @param dest destination of edge

\* @return an edge

\*/

@Override

public Edge getEdge(int source, int dest)

/\*\*

\* This method return Edge Iterator

\* @param source source of edge

\* @return Iterator class

\*/

@Override

public Iterator<Edge> edgeIterator(int source)

/\*\*

\* This method return info about this class

\* @return string informations.

\*/

@Override

public String toString()

/\*\*

\* This inner Class for Edge iterator

\*/

private class IteratorEdge implements Iterator {

...

/\*\*

\* Constructor 1 parameter

\* @param source source of edge

\*/

public IteratorEdge(int source)

/\*\*

\* This Method has next element controlling.

\* @return true if has next element,otherwise false.

\*/

@Override

public boolean hasNext()

/\*\*

\* This method return next edge

\* @return Next edge

\*/

@Override

public Edge next()

/\*\*

\* This method remove an edge exception NoSuchElementException

\*/

@Override

public void remove()

* **List Graph Class;**
  + **Veri tip edge listesi**
  + **List<Edge>[]**

**/\*\***

**\* This Class extends Abstract class and**

**\* Data Filed is List edges.**

**\* @author ercan**

**\*/**

**public class ListGraph**

**extends AbstractGraph {**

**/\*\***

**\* Construct a graph with the specified number of vertices and**

**\* directionality.**

**\* @param numV The number of vertices**

**\* @param directed The directionality flag**

**\*/**

**public ListGraph(int numV, boolean directed) {**

**super(numV, directed);**

**...**

**}**

**/\*\***

**\* Determine whether an edge exists.**

**\* @param source The source vertex**

**\* @param dest The destination vertex**

**\* @return true if there is an edge from source to dest**

**\*/**

**@Override**

**public boolean isEdge(int source, int dest)**

**/\*\***

**\* Insert a new edge into the graph.**

**\* @param edge The new edge**

**\*/**

**@Override**

**public void insert(Edge edge)**

**/\*\***

**\* This method iterator of list Graph**

**\* @param source source of edge**

**\* @return iterator by using source**

**\*/**

**@Override**

**public Iterator< Edge> edgeIterator(int source)**

**/\*\***

**\* Get the edge between two vertices. If an edge does not exist, an Edge**

**\* with a weight of Double.POSITIVE\_INFINITY is returned.**

**\* @param source The source**

**\* @param dest The destination**

**\* @return the edge between these two vertices**

**\*/**

**@Override**

**public Edge getEdge(int source, int dest)**

**/\*\***

**\* This method return info about this class**

**\* @return string informations.**

**\*/**

**@Override**

**public String toString()**

**}**

**Matrix ve list graphları için;**

**-> Abtract classta tanımladığım loadfull graph ile .xml dosyalarını yükledim**

**-> Her satırı okumak için loadfile ile satırların içindeki değerlere ulaşıp,**

**-> insert methodu ile edgeleri ekledim.**

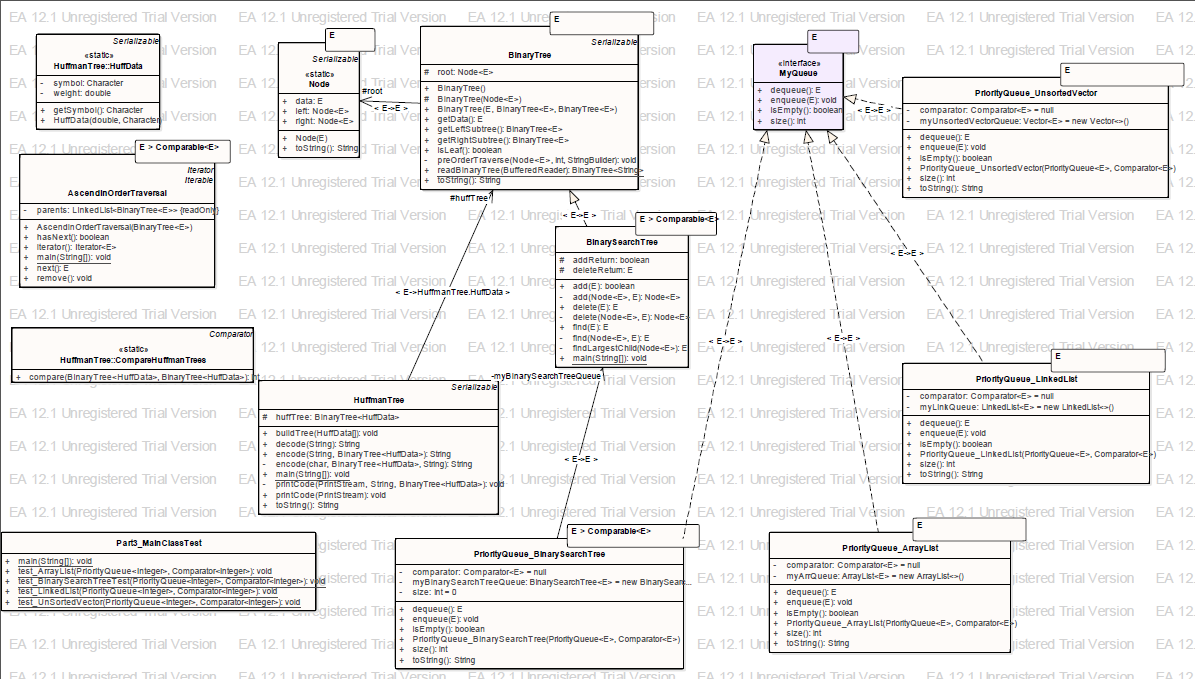
**-> Oluşturulan graphları DijkstraAlgorithm ve prim’s algoritmalari kullanarak.**

**-> sonuçları main de test ettim.**

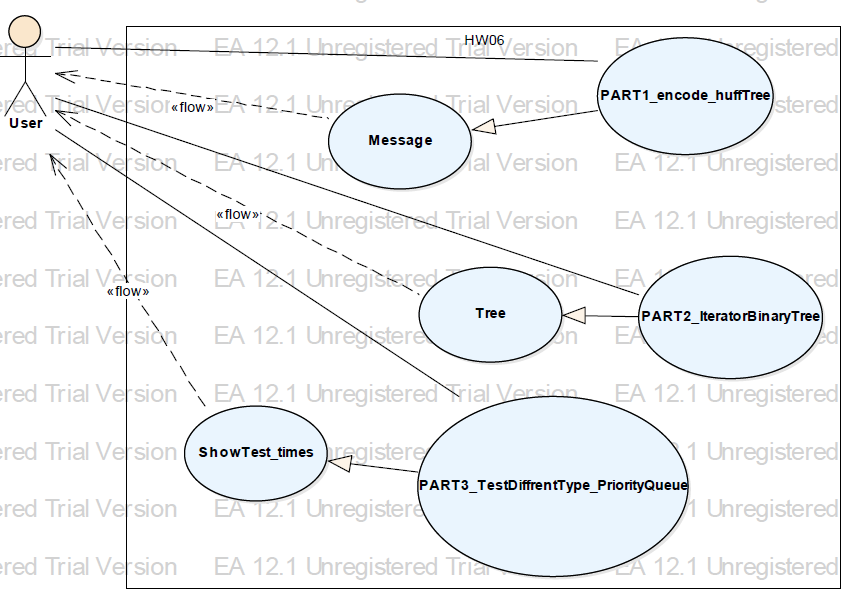
**-> Bazı hatalar söz konusu oldu bazı .xml dosylarıdaki değerlerin farklılığından o yüzden az test edilmiş oabilir.**

**->Son olarak list ve matrix için toplam geçen süreleri ekranda gösterdim.**

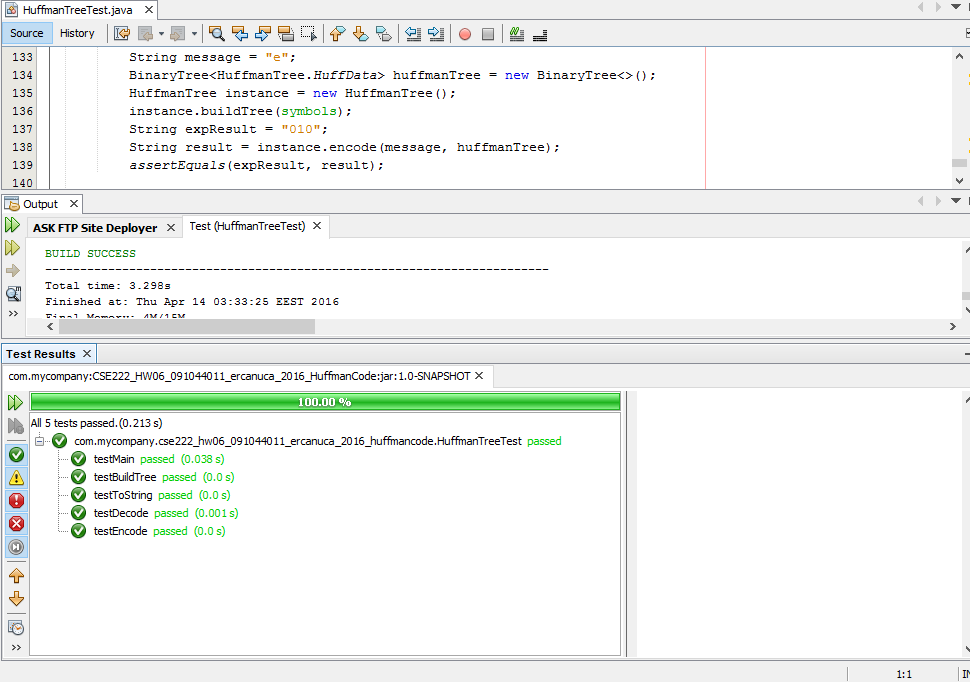
**Class Diagramı**

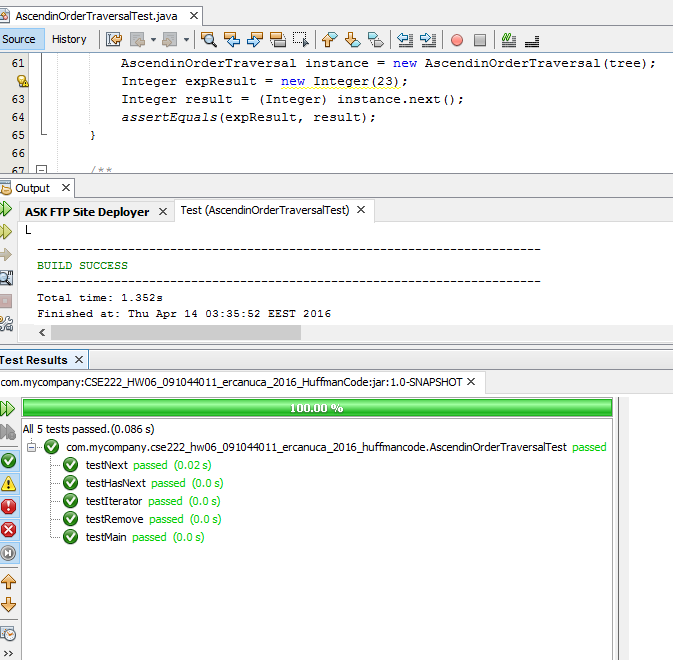


**Use Case Diagrams**



**Tests**





**Main tests**

**-----------------------PART\_1-----------------------**

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

space: 111

Code to Message :

110000100111111100101000011 : q rg

Message to code :

merhaba ben ercan uca : 110010010001000011010100000101011110000001001111110100010000001010011111100001000001010

---------------------PART\_1\_END---------------------

-----------------------PART\_2-----------------------

36

* + - * / \
      * / \
      * / \
      * / \
      * 25 40
      * / \ / \
      * / \ / \
      * / \ / \
      * 10 27 35 45
      * / \ / \ / \ / \
      * / \ / \ / \ / \
      * 5 15 26 28 32 37 43 50
      * /\ /\ /\ /\ /\ /\ /\ /\
      * null null null null null null

36

25

10

5

null

null

15

null

null

27

26

null

null

28

null

null

40

35

32

null

null

37

null

null

45

43

null

null

50

null

null

5

10

15

25

26

27

28

36

32

35

37

40

43

45

50

---------------------PART\_2\_END---------------------

-----------------------PART\_3-----------------------

--------------ARRAYLIST\_QUEUE\_TEST------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 1 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 1 element\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 18566 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 1 element\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 18112 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 1 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 10 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 10 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 27169 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 10 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 11320 ns

\*\*\*\*\*\*\*\*\*\*\*\*End for 10 numbers\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 100 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*Enqueu 100 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 153505 ns

\*\*\*\*\*\*\*\*\*\*\*Dequeu 100 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 68828 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 100 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 1000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 1000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 1336715 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 1000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 950463 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 1000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 10000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 10000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 4 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 10000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 14 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 10000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 100000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 100000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 21 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 100000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 1923 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 100000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 1000000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 1000000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 265 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 1000000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 274361 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 1000000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*

----------------END\_ARRAYLIST\_QUEUE\_TEST------------

--------------LINKEDLIST\_QUEUE\_TEST------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 1 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 1 element\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 20377 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 1 element\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 26716 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 1 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 10 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 10 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 6793 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 10 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 14943 ns

\*\*\*\*\*\*\*\*\*\*\*\*End for 10 numbers\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 100 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*Enqueu 100 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 62941 ns

\*\*\*\*\*\*\*\*\*\*\*Dequeu 100 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 64753 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 100 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 1000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 1000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 573266 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 1000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 499910 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 1000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 10000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 10000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 2 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 10000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 1 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 10000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 100000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 100000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 15 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 100000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 5 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 100000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 1000000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 1000000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 267 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 1000000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 22 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 1000000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*

----------------END\_LINKEDLIST\_QUEUE\_TEST------------

--------------UNSORTED\_VECTOR\_QUEUE\_TEST------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 1 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 1 element\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 13131 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 1 element\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 12226 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 1 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 10 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 10 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 8151 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 10 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 9510 ns

\*\*\*\*\*\*\*\*\*\*\*\*End for 10 numbers\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 100 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*Enqueu 100 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 81960 ns

\*\*\*\*\*\*\*\*\*\*\*Dequeu 100 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 57960 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 100 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 1000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 1000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 662471 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 1000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 935067 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 1000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 10000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 10000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 3 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 10000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 12 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 10000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 100000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 100000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 15 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 100000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 1286 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 100000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 1000000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 1000000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 171 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 1000000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 261435 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 1000000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*

----------------END\_UNSORTED\_VECTOR\_QUEUE\_TEST------------

---------BINARY\_SEARCH\_TREE\_QUEUE\_TEST------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 1 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 1 element\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 6792 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 1 element\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 1358 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 1 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 10 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 10 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 3169 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 10 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 2265 ns

\*\*\*\*\*\*\*\*\*\*\*\*End for 10 numbers\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 100 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*Enqueu 100 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 31697 ns

\*\*\*\*\*\*\*\*\*\*\*Dequeu 100 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 9056 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 100 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 1000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 1000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 156675 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 1000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 155316 ns

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 1000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 10000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 10000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 1 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 10000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 10 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 10000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 100000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 100000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 23 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 100000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 1348 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 100000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*For 1000000 integer\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enqueu 1000000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 172 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*Dequeu 1000000 elements\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Took approximately 287336 ms

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End for 1000000 number\*\*\*\*\*\*\*\*\*\*\*\*\*\*

----------END\_BINARY\_SEARCH\_TREE\_QUEUE\_TEST---------

-----------------PART\_3\_END-------------------------

Ödev githup linki

<https://github.com/erccanuca/cse222_hw06_BinarySeachTree_HuffmanTree_encoding.git>

(Ödev teslim süresi geçince public yapılacak.)