



RAPOR

DERSİN ADI : VERİ YAPILARI VE ALGORİTMALAR

ÖĞRENCİNİN ADI - SOYADI : YUNUS KARATEPE

ÖĞRENCİNİN NUMARASI : 17011051

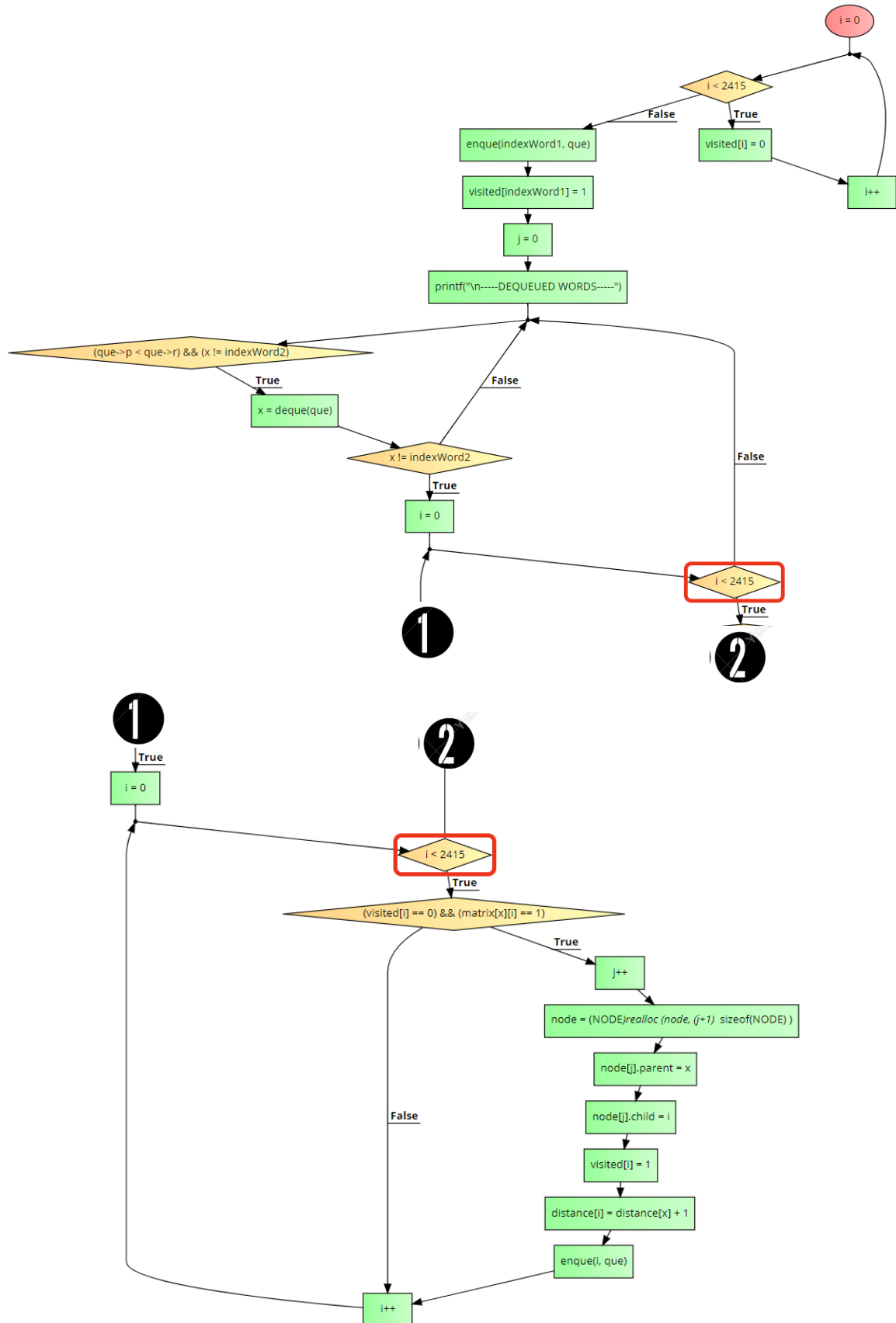
PROJE KONUSU : GRAF İŞLEMLERİ

1 – Yöntem

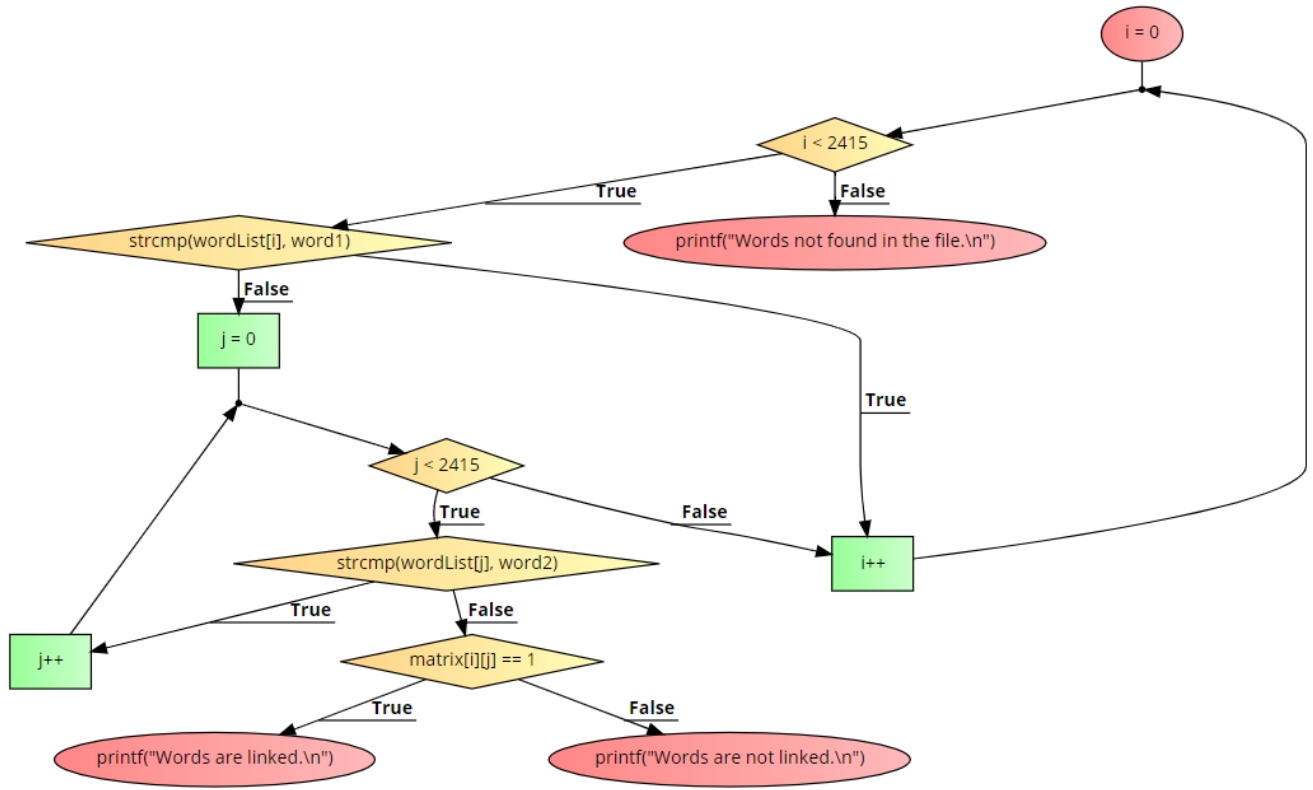
Projede bizden ilk olarak bir “.txt” dosyası içerisindeki kelimeleri okuyup bunlardan bir “adjacency matrix” oluşturmamız istenmiş. Okuma aşamasında ,“%s” ile bir döngü içerisinde, bütün kelimeler “word” isimli dizinin içerisine alınıyor. Daha sonra bu “word” deki tüm elemanları birbirleriyle karşılaştırıp 5 harfinden 4 tanesi aynı olan kelimelerin “adjacencyMatrix” de denk geldikleri komşuluk (bağlantı) bölgeleri 1 yapılmıştır. Bu aralarında bağlantı olduğu anlamına gelmektedir. Aksi takdirde ise 0 yapılmıştır. Böylece iki kelime arasında yalnızca 1 harf değiştirilerek diğerinin elde edilebileceğini 1 ler ile gösteren bir “adjacencyMatrix” oluşturulmuştur. Bağlantının olup olmadığını kontrol eden “isLinked” fonksiyonu da adjacencyMatrix[kelime1][kelime2] ya da adjacencyMatrix[kelime2][kelime1] ,ki ikisinde de aynı değer olmak zorundadır, noktalarına ulaşarak bu değer 1 ise bağlantının olduğunu, 0 ise bağlantının olmadığını bulmaktadır. Dönüşümleri bulan “breadthFirstSearch” fonksiyonu ise “Her kelimedenden her adımda 1 harf değiştirilerek diğer kelime elde edilebilir mi?” sorusuna yanıt bulmaktadır. Daha sonrasında ise eğer böyle bir dönüşüm yapılabiliriyorsa “displayPath” fonksiyonu vasıtası ile en kısa yolu ekrana yazdırmaktadır. Bu fonksiyon aynı zamanda “QUE” ismiyle tanımlanmış kuyruk veri yapısını kullanmaktadır. Her adımda kuyruktan en son çekilen ve eğer bu çekilen 2. Kelimeye eşit olmayan kelimenin bütün komşularını adjacencyMatrix yardımıyla bularak kuyruğa yerleştirmektedir. Bu işlem kuyruk bitene (ki kuyruğun bitmesi dönüşümün mümkün olmadığı anlamına gelmektedir) ya da kuyruktan çekilen kelime 2. Kelime olana (bu da dönüşüm mevcut, 2. Kelime elde edilebilir anlamına gelmektedir) kadar aynı işlemleri devam ettirir. Aynı zamanda bir kere kuyruğa atılan kelimenin bir daha kuyruğa girmesine gerek yoktur. Bunun için de “visited” isimli dizi veri yapısı kullanılmıştır. Bu kelimenin visited de denk geldiği indis 1 ise bu kelime daha önce kuyruğa girmiş demektir ve kuyruğa atılmaz. Kuyruğa atmak için “enqueue”, çıkarmak için ise “deque” fonksiyonları yazılmıştır. Ayrıca kuyruğa atılan her kelimenin kim ile bağlantısı olduğu için kuyruğa atıldığının (“parent”) tutulması gereklidir ki en kısa yolu yazdırırken bu “parent” ve “child” ilişkisini kullanarak işimizin kolaylaştırılması hedeflenmiştir. Bunun için ise ayrı bir “NODE” isimli struct yapısı kullanılmıştır. Menu switch-case işlemleri ile oluşturulmuştur.

Genel anlamda Breadth First Search algoritmasının anlaşılması açısından verimli bir proje olmuştur. Daha fazla açıklama source kodun içerisinde bulunabilir.

breadthFirstSearch() fonksiyonu algoritması



isLinked() fonksiyonu algoritması



2 – Uygulama

```
C:\Users\yunus\OneDrive\Masaüstü\VeriYapılar\Proje\main.exe
ENTER A OPERATION NUMBER :
Show all words           : 1
Is linked                : 2
Find a transition between 2 words : 3
Quit                     : 0
3
Enter 2 words :
amuse
agave

-----DEQUEUED WORDS-----
Dequeued word : amuse
Dequeued word : abuse
Dequeued word : abase
Dequeued word : abash
Dequeued word : abate
Dequeued word : awash
Dequeued word : agate
Dequeued word : agave
-----SHORTEST PATH-----
amuse->abuse->abase->abate->agate->agave
-----NUM OF STEPS-----
There is transition. Number of steps = 5
ENTER A OPERATION NUMBER :
Show all words           : 1
Is linked                : 2
Find a transition between 2 words : 3
Quit                     : 0
```

Adım 1 : “amuse” kelimesi kuyruğa atılır, daha sonra kuyruktan çekilir ve “amuse” ye komşu olan bütün kelimeler (yalnızca “abuse”) kuyruğa atılır.

Adım 2 : Kuyruktaki en üstteki kelime “abuse” kuyruktan çekilir ve komşusu olan “abase” kuyruğa atılır.

Adım 3 : “abase” çekilir ve komşusu olan “abash” ve “abate” kuyruğa atılır.

Adım 4 : “abash” ın komşusuz olan “awash” kuyruğa atılır.

Adım 5 : Kuyrukta en üstte “abate” vardır ve kuyruktan çekilir ve yerine “agate” atılır.

Adım 6 : “awash” ın daha önceden kuyruğa girmeyen herhangi bir komşusu yoktur sadece “awash” kuyruktan çekilir.

Adım 7 : “agate” kuyruktan çekilerek önceden kuyruğa girmemiş olan komşusu “agave” kuyruğa atılır.

Adım 8 : “agave” kuyruktan çekilir ve çekilen kelime aradığımız kelime olduğu için döngü sonlanır.

SHORTEST PATH : Path oluşturulurken ise bir parentin (o döngü anında deque edilmiş olan kelime) kuyruğa atılan tüm childları (komşuları) bir 2 li struct yapısında tutulduğu için en son kelimeye ulaşıldığında o kelimeden başlayarak sürekli parentine giderek, ilk kelimeye ulaşılabilir. Bu vasıta ile ekrana path yazdırılır.

Tamamen aynı mantık ile çalışan bir başka örnek.

```
C:\Users\yunus\OneDrive\Masaüstü\VeriYapılar2Proje\main.exe
Is linked : 2
Find a transition between 2 words : 3
Quit : 0
3
Enter 2 words :
sonic
tunic

-----DEQUEUED WORDS-----
Dequeued word : sonic
Dequeued word : conic
Dequeued word : ionic
Dequeued word : monic
Dequeued word : tonic
Dequeued word : comic
Dequeued word : cynic
Dequeued word : manic
Dequeued word : topic
Dequeued word : toxic
Dequeued word : tunic
-----SHORTEST PATH-----
sonic->tonic->tunic
-----NUM OF STEPS-----
There is transition. Number of steps = 2
ENTER A OPERATION NUMBER :
Show all words : 1
Is linked : 2
Find a transition between 2 words : 3
Quit : 0
```

- “runic” ve “study” kelimelerinin dönüşümünü arattığımızda

```
C:\Users\yunus\OneDrive\Masaüstü\VeriYapıları\Proje\main.exe
Dequeued word : scrim
Dequeued word : splay
Dequeued word : vault
Dequeued word : flail
Dequeued word : brawl
Dequeued word : crawl
Dequeued word : bluet
Dequeued word : blurt
Dequeued word : fruit
Dequeued word : craft
Dequeued word : draft
Dequeued word : kraft
Dequeued word : droll
Dequeued word : twirl
Dequeued word : barre
Dequeued word : large
Dequeued word : stead
Dequeued word : steal
Dequeued word : steam
Dequeued word : quiet
Dequeued word : quill
Dequeued word : quint
Dequeued word : quirt
Dequeued word : splat
Dequeued word : fault
Dequeued word : flair
Dequeued word : blurb
Dequeued word : croft
Dequeued word : drift
Dequeued word : kraut
Dequeued word : drool
Dequeued word : swirl
Dequeued word : quell
Dequeued word : quart
Dequeued word : quirk
Dequeued word : split
Dequeued word : droop
Dequeued word : quark
Dequeued word : quick
Dequeued word : troop
Dequeued word : quack
-----NUM OF STEP-----
There is no transition from one word to another.
ENTER A OPERATION NUMBER :
Show all words          : 1
Is linked               : 2
Find a transition between 2 words : 3
Quit                   : 0
```

Uzun süren bir BST sonrasında dönüşümün olmadığı görülür.