KOCAELI UNIVERSITESI
BILGISAYAR MUHENDISLIGI
-
PROLAB I-3
PRIORITY QUEUE

Muhammed Numan MERCAN
200202098

Mehmet Ali AKDOGAN
20202017

I. BAŞLARKEN

Bu projede oncelikli kuyruk (priority queue) veri yapısı kullanarak hava uçuş kontrol sistemi yapmamız amaçlanmıstır.
Kuyruk veri yapisi adındanda anlaşılacağı üzere ard arda elemanların eklenerek ve ilk eklenen elemanların kuyruktan ilk ayrılması ile sonuclanan (FIFO - Firs in First Out prensibi) bir veri yapısı algoritmasıdır. Bu yapı dizi veya baglı liste yapısı kullanılarak oluşturulabilir. Biz bu projede projenin kapsamı dolayısıyla bağlı liste kullanarak projeyi gerçekleştirdik.
Queue leri anlamak icin kısaca şu adımları sayabiliriz::

• Kuyrugun ilk kısmına (başına) ”FRONT” son kısmına ”REAR” denir
• Kuyruk yapısında iki uctanda işlemler yapılabilir
• Enqueue ile kuyruga eleman eklenir (kuyruk sonuna ya da oncelige göre)
• Dequeue ile kuyruktan eleman cikarilir. Front kısımdaki eleman ilk olarak işlem yapacagı icin ilk olarak o cıkarılır.

Kuyruk yapısında belli başlı fonksiyonlar vardır , bunlardan bazıları :
•peek() : Sıradaki elemani getirir
•add() : Kuyruga eleman ekler
•remove() : Belirli elemani siler
•poll () : Son elemani getirir ve siler
•clear() : Tum elemanlari siler

II. OZET

Projeye queue yapısını tekrar inceleyip iyice ogrenerek basit alıştırmalar yaparak başladık. Basit yapılarda kullanmayı ogrendikten sonra bu kuyrukları oncelik sırası ile çalıştırmayı ve onlar üstünden işlem (cıkar,ekle) yapmayı başardık.
Ardından isterleri nasıl yapabileceğimiz hakkında planlama ve kabaca taslak oluşturma aşamasına geçtik bu aşamada örnek kopleks oncelik yapılarını - projelerini inceleyerek nelerin nasıl yapıldıgı hakkında fikir edindik. Basit bir akış şeması ile süreci gorunur kılmaya çalıştık ve ardından zaten alıştırma yaptıgımız kodlamaya iyice giriştik. Oncelikli kuyrukta oncelikleri tam olarak sıralayabilmek bizi fazlasıyla yordu. Ve bu oncelikleri inis saatleri ile paralel işleyebilmek işimizi en zorlayan yer burası oldu. Yaptığımız sonuları yazdirarak terminalde gosterdik.

Rapor , yaptigimiz bu projeyi her tarafıyla açıklıyor olacaktır.

III. GİRİŞ

Priority queue yapısını öğrenip basit kodlara kurduktan sonra isterlere göre bir kuyruk yapısı olusturmaya koyulduk , eklenen elemanı onceki eleman veya elemanlarla - bu elemanları temp ve prevTemp adlı değişkenlerde saklayarak tüm elemanları kontrol etme amacı ile kullandık- kontrol eden koşul ve döngüler kurduk. Kurdugumuz koşşullu yapılarda bazı goremedigimiz ve bizi

fazlasıyla ugrastıran mantık hataları oldu. Bu hataları deneyerek gormeye ve düzeltmeye çalıştık.
Bu şekilde yaklaşık yapımızı olusturduktan sonra yapılması olası isterleri gostermek ve sectirmek adına secim menusu hazırladık. Bu menu ile manuel secimler ile kuyruk oluşturuluyor ve üzerinde isterlere gore kontroller saglanıyordu.
Tabi projede istenen bu ekleme ve düzenlemeleri el ile değil otomatik olarak ”input.txt” dosyasından cekerek alıp kontrollere sokmaktı. Bu yüzden elimizdeki yapıyı verilerin dosyadan alınacak haliyle düzenlemeye çalıştık.

A. *YONTEM ve İLERLEYİŞ*

Biz öncelikle bir çok projede yaptıgımız gibi projenin temel yapısındaki algoritmayı , ardından o algoritma dogrultusunda bizden istenenleri ve bizim ilk olarak yapabileceklerimizi araştırmamız ile ilgili kücük bir yol haritası ve taslak oluşturma ile başlıyoruz. Bu planlama sonrası açık bir anlayış kazandıysak projeye karsı tam olarak kodlamaya başlıyor ve ilerliyorduk. Bu projede kuyruk yapısı hakkında fazla eksiğimiz ve az tecrübemiz olduğu için yapabileceklerimiz ve onları nasıl gerçekleştireceğimiz konusunda aksaklıklar yaşadık. Bu durum haliyle projenin tam anlayısla değil anlayarak kodlama yoluyla olusmasını sagladı.

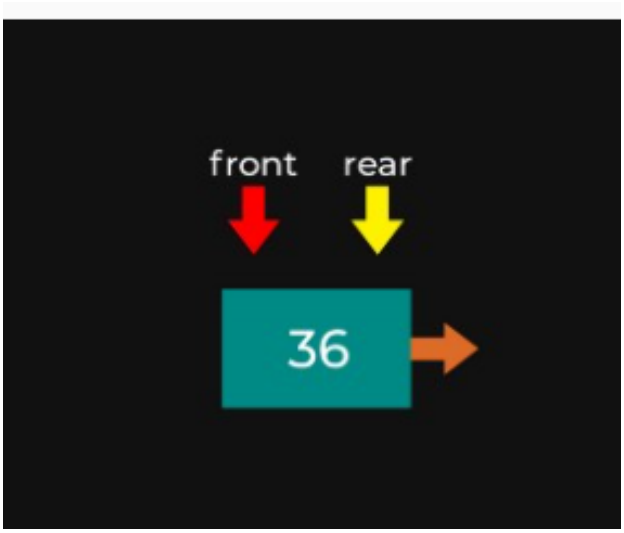

Fig. 1. Tek elemanli bir kuyruk ornegi
Bu yapi kuyrukta tel eleman oldugunda front ve rear yapisinin nerede oldugunu acikliyor , tek eleman kaldiginda olusabilecek mantık ve referans hatalarini gidermek icin boyle bir tanımlama yapmamiz gerekecek.

Oncelik	UcakID	Saat
1	1	14
2	2	13
2	3	11
3	4	10
4	5	16
2	6	10
3	7	2
3	8	14
1	9	17
4	10	13
1	11	1
1	12	1

Fig. 2. Dosayadan okunan sirasiz veriler
Burada ”input.txt” dosyasindan okunan degerlerin oncelik , ucakID , saat verileri ile okunmus hali bulunuyor.

GUNCEL KUYRUK

ONCELIK_ID	UCAK_ID	SAAT
1	11	1
1	12	1
3	17	1
4	22	1
3	7	2
2	23	2
2	27	2
4	28	3
3	24	8
2	13	9
3	4	10

Fig. 3. Verilerin kuyruga eklenmis hali
Burada okunan veriler saat ve oncelik baz alınarak kuyruga eklenmis durumdalar.

IV. SONUCLAR

Projede sürecin sonuna yaklasirken elimizde pek elle tutulur bir program yoktu fakat gozden kacirdigimiz kücük ama etkisi büyük bir problemi farketmemeiz sonucunda o problemi cozerek cok hızlı bir sekilde sonuca ve neredeyse tüm isterleri karşılar hale ulaştık. Bize başta kuyruk veri yapısını ve en önemlisi de bazen resme biraz daha geriden bakma tecrubesini edinmeyi saglaması projenin en buyuk getirilerinden oldu. Bu sayede hem teknik hemde hayat tecrübesi açısından güzel bir deneyim edinmiş olduk. Bu proje ile birlikte planlamayı , sürec yonetmedeki hataları kodlama ve planlama arasındaki baglantiyi ve problemi en basit haliyle cozebilmenin avantajlarını ogrenmiş ve ileriki deneyimlerimizde kullanmak üzere almış olduk.

```
28 nolu ucak havaalanindan ayrildi.
Ucagin kalkis yaptigi saat : 3
----------
24 nolu ucak havaalanindan ayrildi.
Ucagin kalkis yaptigi saat : 8
----------
13 nolu ucak havaalanindan ayrildi.
Ucagin kalkis yaptigi saat : 9
----------
```

Fig. 4. Sirasi geldiginde kalkis yapan ucaklar

Burada kuyruga girmis ve sirasi gelmis ucaklarin kalkis yapmasi ve kalkis ile ilgili bazi bilgileri veriliyor

Proje için planlama yaparken oluşturdugumuz ve problemin çözümünü anlaşılır kılan ve kodlamada işimize yarayan akış şemasını raporun ilerleyen kısımlarında göreceksiniz.Bu şemaları ayrı ayrı yapmakta fayda gördük çünkü birleşik bir şema fazlasıyla kompleks ve büyük duruyor olacaktı.

Bu raporda projemizi sürecin başından sonuna kadar hem problemin hemde problem cözümünün aşamalarını açıklamak ve anlaşılır kılmak için bir rehber niteliği görmektedir.

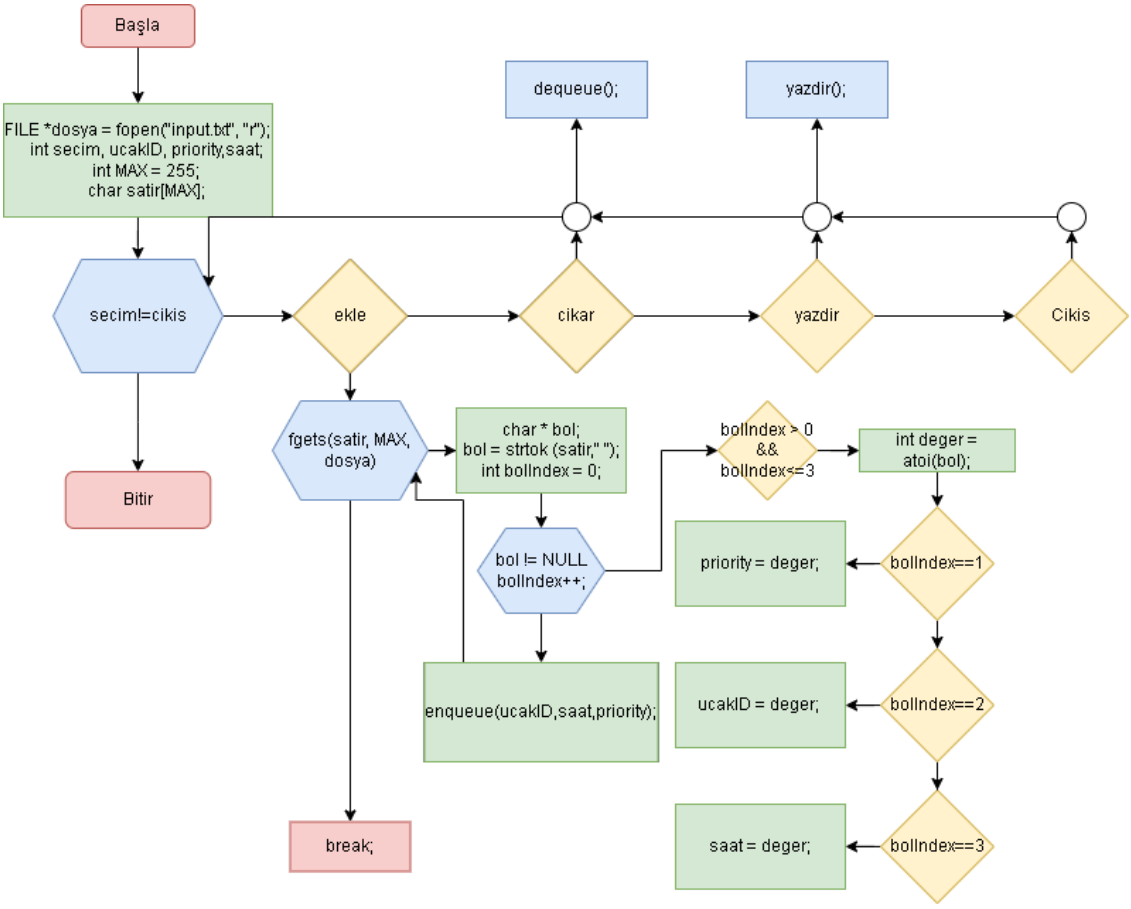

Fig. 5. Akis Semasi Ana Govde

Burada main fonksiyonu icinde yaptigimiz islem akışını akış şemasi ile gosterdik , icerde kullandıgımız diger fonksiyonların algoritma akış şemaları bu kısımda karışık ve büyük duracağından dolayı onlarıda ayrı bir şema ile gosterdik.

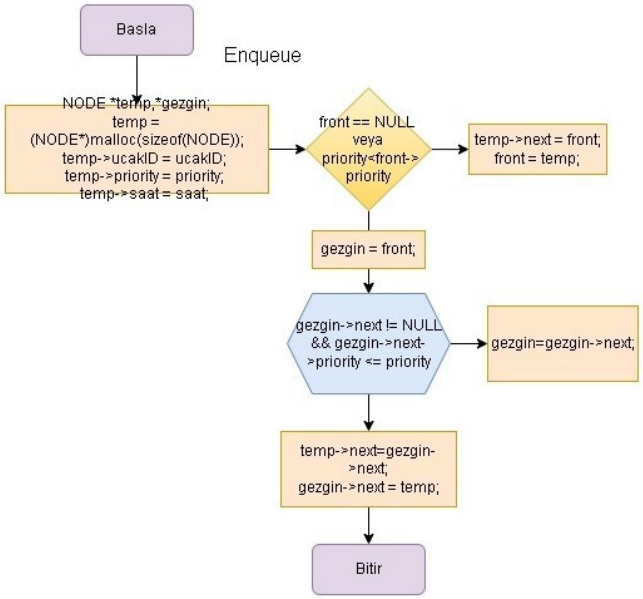

Fig. 6. Enqueue fonksiyon FC.

Kuyruga eleman ekleme fonksiyonlari bu kısımda bulunuyor.

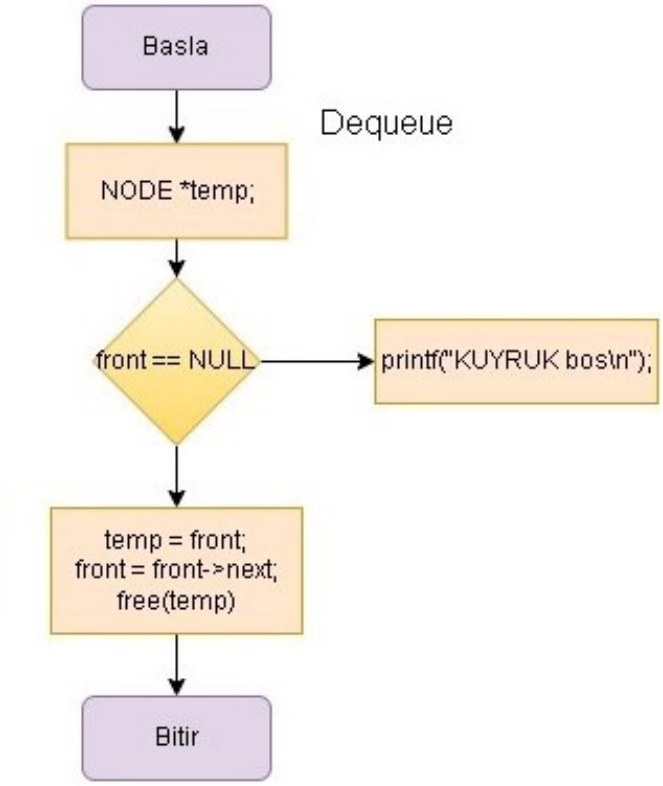

Fig. 7. Dequeue fonksiyon FC.

Kuyrukta siradaki elemani cikarma fonksiyonlari bu kısımda bulunuyor.

V. KAYNAKCA

- https://medium.com/@tolgahan.cepel
  Veri Yapıları : Queue
- http://www.belgeler.org/glibc/glibc-Finding-Tokens-in-a-String.html
- https://www.javatpoint.com/priority-queue-using-linked-list
- http://www.baskent.edu.tr/
- Bilgisayar Kavramlari : Veri Yapilari - Queue
  https://bilgisayarkavramlari.com/
- https://www.geeksforgeeks.org/priority-queue-using-linked-list/
- https://en.wikipedia.org/wiki/
  Priority Queue