Yapısal Programlama Proje Ödevi

Öğrenci Adı: Burak ERTAN

Öğrenci Numarası: 22011073

Dersin Eğitmeni: Hafize İrem TÜRKMEN

Video Linki: https://drive.google.com/file/d/1B-So0cWOUlgkaNsriMJ6C1YJi7Kp7z_c/view?usp=sharing

1. Kullanılan Veri Yapıları
    - Author = Bir yazarı temsil eder. Ad, soyad ve benzersiz id içerir.
    - AuthorNode = Author bilgilerini tutan, tek yönlü bağlı liste düğümüdür. Yazarlar ID sırasıyla tutulur.
    - IDqueue = Silinen yazarların ID'lerini geri kazanmak için kullanılan **circular queue** yapısıdır.
    - Book = Bir kitabın temel bilgilerini (isim, ISBN, adet) içerir. ISBN 13 haneli sabit uzunluktadır.
    - BookCopy = Her bir kitabın bireysel bir kopyasını temsil eder. etiket değeri ISBN_x formatındadır.
    - BookCopyNode = BookCopy bilgilerini tutan tek yönlü bağlı liste düğümüdür. Her Book'un içinde bulunur.
    - BookNode = Her kitap için Book + ona ait tüm BookCopyNode'ları barındırır. Kitaplar **isme ve ISBN'e göre sıralı** olarak tutulur.
    - BookAuthorLink = Her kitap ile yazar(lar)ı arasındaki ilişkiyi tutar. Bir kitabın birden fazla yazarı olabilir, her ilişki ayrı node'dur.
    - Student = Öğrencilerin adı, soyadı, 8 haneli ID'si, 100 puanlık ceza sistemi puanı ve geçmiş ödünç alma işlemleri bulunur. Öğrenciler **çift yönlü bağlı listede** saklanır.
    - BorrowHistory = Bir öğrencinin kitap ödünç alma veya iade işlemlerinde tutulan kayıt yapısıdır. ISBN, etiket, alış ve iade tarihlerini içerir.
    - BorrowHistoryNode = Yukarıdaki BorrowHistory bilgilerini bağlayan tek yönlü liste düğümüdür. Her öğrenciye özel geçmiş tutulur.

```c
typedef struct Author{
    char *name;
    int id;
    char *surname;
}Author;

typedef struct AuthorNode{
    Author *data;
    struct AuthorNode *next;
}AuthorNode;

typedef struct IDqueue{
    int id;
    struct IDqueue *next;
}IDqueue;

typedef struct Book{
    char *name;
    char isbn[14];
    int adet;
}Book;

typedef struct BookCopy{
    char *etiket;
    char *durum;
}BookCopy;
```

```c
typedef struct BookCopyNode{
    BookCopy *data;
    struct BookCopyNode *next;
}BookCopyNode;

typedef struct BookNode{
    Book *data;
    BookCopyNode *copyHead;
    struct BookNode *next;
}BookNode;

typedef struct BookAuthorLink{
    char isbn[14];
    int authorID;
    struct BookAuthorLink *next;
}BookAuthorLink;

typedef struct Student {
    char *name;
    char *surname;
    int point;
    char id[9];
    struct Student *prev;
    struct Student *next;
    struct BorrowHistoryNode *historyHead;
} Student;
```

```c
typedef struct BorrowHistory {
    char *isbn;
    char *etiket;
    char *alisTarihi;
    char *iadeTarihi;
} BorrowHistory;


typedef struct BorrowHistoryNode {
    BorrowHistory data;
    struct BorrowHistoryNode *next;
} BorrowHistoryNode;
```

2. Temel Fonksiyonlar

Yazar İşlemleri

Yeni yazar eklenirken önce **IDqueue kuyruğunda saklanan silinmiş ID'ler kontrol edilir**. Eğer kuyruk boş değilse, ilk değer çekilir ve bu ID atanır. Kuyruk boşsa, mevcut en büyük ID bir artırılarak yeni bir ID verilir. Yazar, AuthorNode yapısı ile sıralı bağlı listeye eklenir ve yazarlar.csv dosyasına yazılır.

Silme işleminde, ilgili yazar listeden çıkarılır ve ID'si IDqueue içine tekrar eklenir. Böylece sistem gereksiz ID şişmesi olmadan çalışmaya devam eder. Ayrıca KitapYazar.csv dosyasında bu ID yerine -1 yazılarak bağlantı koparılır. Güncellemede yazarın adı ve soyadı değiştirilir.

Kitap İşlemleri

Yeni kitap eklenirken ISBN numarası kontrol edilir. Daha önce eklenmişse sadece yeni kopyalar oluşturulur. ISBN yoksa kitap sıfırdan tanımlanır ve her kopya "RAFTA" durumuyla etiketlenir (ISBN_1, ISBN_2 gibi). Kopyalar BookCopyNode zinciriyle bağlanır.

Kitap silme işleminde yalnızca tüm kopyalar raftaysa silme yapılır. İlgili BookAuthorLink bağlantıları da kaldırılır. Güncellemede kitap adı değiştirilebilir veya yazar listesi baştan belirlenebilir. Bu durumda eski bağlantılar silinir ve yeni eşleştirmeler yapılır.

Öğrenci İşlemleri

Öğrenciler çift yönlü bağlı liste yapısıyla saklanır. Yeni eklenen öğrenciye 8 haneli bir ID, isim ve soyisim verilir; başlangıç puanı 100'dür. Güncelleme işleminde öğrenci ismi veya soyismi değiştirilebilir. Silme işleminde öğrencinin tüm bilgileri ve ödünç alma geçmişi temizlenir.

Öğrenci bilgisi sorgulandığında kimlik bilgileriyle birlikte kitap alışveriş geçmişi listelenebilir. Ayrıca rafta olmayan kitaplar ve gecikmiş iadeler kolaylıkla tespit edilebilir.

Ödünç ve İade Sistemi

Bir öğrenci kitap almak istediğinde önce puanı kontrol edilir. Kitap kopyası etiket ile belirlenir. Durumu "RAFTA" olan kopyalar ödünç verilebilir. Bu durumda, kitap durumu öğrenci numarası olarak değiştirilir ve işlem OduncIslemleri.csv dosyasına yazılır.

İade sırasında yine etiket üzerinden işlem yapılır. Gecikme varsa 15 günü geçen iadelerde öğrenciden 10 puan kesilir. Kitap tekrar "RAFTA" durumuna döner ve işlem geçmişe işlenir.

Dosya Senkronizasyonu

Program açılışında tüm .csv dosyaları belleğe yüklenir. Bu sayede uygulama bellekte hızlı çalışır. Her işlem sonrası dosyalar güncellenerek veri kaybı engellenir. readAuthorsFromCSV fonksiyonu, kullanılan ID'leri analiz eder; **kullanılmayan ID'leri kuyruk yapısına (IDqueue) ekleyerek sistemin sürdürülebilirliğini sağlar.**

**Kodlar bu bölümde her veri yapısı için çok benzerdir. Bu yüzden sadece Yazar veri yapısına ait olanları rapora ekledim

```c
Author* createAuthor(char *name,int id,char *surname){
    Author *newAuthor  = (Author*)malloc(sizeof(Author));
    if(newAuthor == NULL){
        return NULL;
    }
    newAuthor->name = (char*)malloc(sizeof(char) * strlen(name)+1);
    newAuthor->surname = (char*)malloc(sizeof(char) * strlen(surname)+1);

    if (newAuthor->name == NULL || newAuthor->surname == NULL) {
        free(newAuthor->name);
        free(newAuthor->surname);
        free(newAuthor);
        return NULL;
    }
    strcpy(newAuthor->name,name);
    strcpy(newAuthor->surname,surname);
    newAuthor->id = id;
    return newAuthor;
}
int addAuthor(AuthorNode **head,IDqueue **rear,char *name,char *surname,int *currentID){
    int newID = createID(rear, currentID);
    Author *newAuthor = createAuthor(name,newID,surname);
    if (newAuthor == NULL) return -1;

    AuthorNode *newNode = (AuthorNode*)malloc(sizeof(AuthorNode));
    if (newNode == NULL) return -1;
    newNode->data = newAuthor;
    newNode->next = NULL;
    //Liste Boş veya başa eklemem gereken durum
    if(*head == NULL || (*head)->data->id > newNode->data->id){
        newNode->next = *head;
        *head = newNode;
    }
    //Arada bir yere ekleme veya sona ekleme durumu
    else{
        AuthorNode *temp = *head;
        while(temp->next != NULL && temp->next->data->id < newNode->data->id){
            temp = temp->next;
        }
        newNode->next = temp->next;
        temp->next = newNode;
    }
    return newID;
}
```

```c
int createID(IDqueue **rear, int *currentID){
    int id = dequeueID(rear);
    if(id != -1){
        printf("Geri ID kullanıldı: %d\n", id);
        return id;
    }
    printf("Yeni ID verildi: %d\n", *currentID + 1);
    return ++(*currentID);
}
```

```c
 void enqueueID(IDqueue **rear,int value){
     IDqueue *newNode = (IDqueue*)malloc(sizeof(IDqueue));
     newNode->id = value;
     printf("ID kuyrukta saklanıyor: %d\n", value);

     if(*rear == NULL){
         newNode->next = newNode;
         *rear = newNode;
     } else {
         newNode->next = (*rear)->next;
         (*rear)->next = newNode;
         *rear = newNode;
     }
 }
```

```c
int dequeueID(IDqueue **rear){
    if (*rear == NULL)
        return -1;

    IDqueue *front = (*rear)->next;
    int id = front->id;

    if (front == *rear) {
        *rear = NULL;
    } else {
        (*rear)->next = front->next;
    }
    printf("ID kuyruktan alındı: %d\n", id);
    free(front);
    return id;
}
```

```c
int deleteAuthor(AuthorNode **head, IDqueue **recycledIDs, int id, const char *authorCSV, const char *bookAuthorCSV){
    if (*head == NULL) {
        printf("Yazar listesi boş.\n");
        return 0;
    }

    AuthorNode *temp = *head;
    AuthorNode *prev = NULL;

    if (temp->data->id == id) {
        *head = temp->next;
    } else {
        while (temp != NULL && temp->data->id != id) {
            prev = temp;
            temp = temp->next;
        }

        if (temp == NULL) {
            printf("Yazar ID %d bulunamadı.\n", id);
            return 0;
        }
        prev->next = temp->next;
    }

    enqueueID(recycledIDs, temp->data->id);

    free(temp->data->name);
    free(temp->data->surname);
    free(temp->data);
```

```c
void refreshAuthorsCSV(AuthorNode *head, const char *filename) {
    FILE *fp = fopen(filename, "w");
    if (fp == NULL) {
        printf("Dosya yazma hatası!\n");
        return;
    }

    AuthorNode *temp = head;
    while (temp != NULL) {
        fprintf(fp, "%d,%s,%s\n", temp->data->id, temp->data->name, temp->data->surname);
        temp = temp->next;
    }

    fclose(fp);
}

int updateAuthor(AuthorNode *head, int id, const char *newName, const char *newSurname, const char *authorCSV) {
    AuthorNode *temp = head;
    while (temp != NULL) {
        if (temp->data->id == id) {
            free(temp->data->name);
            free(temp->data->surname);

            temp->data->name = (char*)malloc(strlen(newName) + 1);
            temp->data->surname = (char*)malloc(strlen(newSurname) + 1);

            if (temp->data->name == NULL || temp->data->surname == NULL) {
                printf("Bellek hatası!\n");
                return 0;
            }

            strcpy(temp->data->name, newName);
            strcpy(temp->data->surname, newSurname);

            refreshAuthorsCSV(head, authorCSV);

            return 1;
        }
        temp = temp->next;
    }

    printf("ID %d ile yazar bulunamadı.\n", id);
    return 0;
}
```

3. CSV Dosyaları
    - Yazarlar.csv
    - Ogrenciler.csv
    - Kitaplar.csv
    - KitapYazar.csv
    - KitapKopya.csv
    - OduncIslemleri.csv

Yazarlar.csv (id,isim,soyisim)

```
1,Ahmet,Yılmaz
2,Ayşe,Demir
3,Mehmet,Kara
```

Kitaplar.csv(isbn,adı,adeti)

```
9789876543210,Algoritmalar,3
9781234567890,Yapay Zeka 101,2
```

Ogrenciler.csv(numara,isim,soyisim,puan)

```
20230001,Zeynep,Yıldız,100
20230002,Emre,Aslan,100
12345678,Burak,Ertan,100
```

KitapYazar.csv(isbn,yazarid)

```
9781234567890,1
9781234567890,2
9789876543210,3
```

KitapKopya.csv(etiket numarası(isbn_n), durumu)

```
9789876543210_1,RAFTA
9789876543210_2,RAFTA
9789876543210_3,RAFTA
9781234567890_1,RAFTA
9781234567890_2,RAFTA
```

oduncIslemleri.csv(numara,etiket,tarih)

```
20230001,9781234567890_1,0,2024-12-01
20230001,9781234567890_1,1,2024-12-15
20230002,9789876543210_1,0,2024-12-0512345678,
12345678,9781234567890_1,1,2025-05-27
```

4. Programla İlgili Çeşitli Çıktılar

```
 ==== KÜTÜPHANE ANA MENÜ ====
 1  - Öğrenci Ekle
 2  - Öğrenci Güncelle
 3  - Öğrenci Sil
 4  - Öğrenci Bilgisi Görüntüle
 5  - Teslim Etmeyen Öğrencileri Listele
 6  - Cezalı Öğrencileri Listele
 7  - Öğrenci Listesini Göster
 8  - Kitap Ödünç Ver
 9  - Kitap İade Al
 10 - Kitap Ekle
 11 - Kitap Güncelle
 12 - Kitap Sil
 13 - Kitap Ara ve Göster
 14 - Raftaki Kitapları Listele
 15 - Gecikmeli İadeleri Listele
 16 - Yazar Ekle
 17 - Yazar Güncelle
 18 - Yazar Sil
 19 - Yazarları Listele
 20 - Kitap-Yazar Eşleşmelerini Göster
 21 - Tüm Verileri Yeniden Yükle (CSV'den)
 22 - Geri Kazanılabilir Yazar ID'lerini Göster
 0  - Çıkış
 Seçiminiz:
Seçiminiz: 5

=== Teslim Etmemiş Öğrenciler ===
Öğrenci: Emre Aslan | Numara: 20230002

==== KÜTÜPHANE ANA MENÜ ====
1  - Öğrenci Ekle
2  - Öğrenci Güncelle
3  - Öğrenci Sil
4  - Öğrenci Bilgisi Görüntüle
5  - Teslim Etmeyen Öğrencileri Listele
6  - Cezalı Öğrencileri Listele
7  - Öğrenci Listesini Göster
8  - Kitap Ödünç Ver
9  - Kitap İade Al
10 - Kitap Ekle
11 - Kitap Güncelle
12 - Kitap Sil
13 - Kitap Ara ve Göster
14 - Raftaki Kitapları Listele
15 - Gecikmeli İadeleri Listele
16 - Yazar Ekle
17 - Yazar Güncelle
18 - Yazar Sil
19 - Yazarları Listele
20 - Kitap-Yazar Eşleşmelerini Göster
21 - Tüm Verileri Yeniden Yükle (CSV'den)
22 - Geri Kazanılabilir Yazar ID'lerini Göster
0  - Çıkış
Seçiminiz:
```

```
Seçiminiz: 13
Aranacak kitap adını girin: Algoritmalar

Kitap: Algoritmalar | ISBN: 9789876543210 | Adet: 3
  - Etiket: 9789876543210_1 | Durum: RAFTA
  - Etiket: 9789876543210_2 | Durum: RAFTA
  - Etiket: 9789876543210_3 | Durum: RAFTA
```

```
Seçiminiz: 12
Silinecek kitabın ISBN numarası: 9789876543210
Kitap başarıyla silindi.
```

Kitaplar.csv

```
1    9781234567890,Yapay Zeka 101,2
2
```

```
Seçiminiz: 4
Öğrenci ID veya Ad-Soyad girin: 12345678

--- Öğrenci Bilgisi ---
Ad: Burak
Soyad: Ertan
ID: 12345678
Puan: 100

--- Kitap Hareketleri ---
Hiç kitap ödünç almamış.
```