Analiz

Program üç farklı ekrandan oluşması gerekiyor. Bunlar, ana menü, sıralama ekranı ve sıcak patates oyununun oynandığı ekran. Ana menü ekranı kullanıcıya ilk girişte gösterilmeli ve sıralama veya sıcak patates ekranını seçene kadar silinememelidir. Ayrıca diğer iki ekranın sonunda “Ana menüye dönmek istiyor musunuz?” sorusu ile kullanıcı, programı tekrar başlatmadan ana menüye dönerek program içinde gezmeye devam edebilir.

Sıralama ekranına ilişkin olarak, stack yapılarının her zaman gösterilmesi – güncel olarak - için ekran sürekli silinip tekrar yazılmalıdır. Sıralama iki stack üzerinden yapılacağından, ikinci stack tampon olarak kullanılmalı. Yani, yeni gelen sayı, birinci stackteki sayıdan büyük olduğu sürece, birinci stackten kaldırılıp, ikinci stack’e aktarılmalı. Bu döngünün sonunda ise, yeni gelen sayı birinci stacke konulup, ikinci stackteki (tampon) sayılar tekrar birinci stacke alınıp, tampon boşaltılmalıdır. Bu işlem belirlenen bir sayı kadar devam ettirilmelidir. Bu sayının daha sonra yeniden düzenlenmesi için sabit olarak tanımlanmasında fayda vardır. Ayrıca, tüm ekleme işlemleri bittikten sonra kullanıcıya “Sıralama yönünü değiştirmek ister misiniz” sorusu yöneltilmeli, eğer evet der ise, birinci stackteki verilerden alıp, tampona koymak, sıralama yönünü değiştirir.

Sıcak patates oyununda ise dairesel bir list kullanmak gereklidir. Oyunun müzik süresi, elde tutma süresi rastgele olarak seçilmelidir. Ekranda bu süreler ve listin durumu, patatesi elinde bulunduran kişi ve vereceği kişi gözükmelidir. Elde durma süresi bitince listin başındaki kişi kaldırılıp, sonuna geçmelidir. Gerekli ilişkiler kurularak oyuna devam etmeli, müzik süresi bittiğinde ise patatesi elinde bulunduran kişi oyundan elenmiş olmalıdır. Oyun elenen kişi olmadan devam eder. Bu oyunun sonunda da her ekranda olduğu gibi “ana menüye dönmek ister misiniz” sorusu sorulmalıdır.

Sıralama Ekranı Tasarımı

>> Initialize *MAX\_NONCE* to 15

>> Initialize *AStack* and *BStack* from *Struct Stack*

>> Initialize *Nonce* to zero

>> While *Nonce* is not equal to *MAX\_NONCE*

>> Print AStack Items

>> Initialize *number*

>> Input the *number*

>> Initialize *lastInsert* to *AStack’s top of value*

>> While *number* is greater than *lastIntert* and *count of AStack* is not equal to *zero*

>> Push *lastInsert* to *Stack StackB*

>> Set *lastInsert* to popped value of *AStack*

>> Push *number* to *AStack*

>> While *count of BStack* is not equal to *zero*

>> Push popped value of BStack to AStack

>> Set *nonce* to *nonce plus one*

>> Print “Sıralamayı büyükten küçüğe değiştirmek istiyor musunuz?”

>> Input *choose*

>> If *choose* is equal to 1

>> Set value to popped value of AStack

>> While *AStack* is not equal to *Empty*

>> Push *Bstack* to *popped value of AStack*

>> Set *value* to *popped value of AStack*

>> Call askMainMenu function to return main menu.

Sıcak Patates Ekranı Tasarımı

>> Initialize *Users* from *Queue*

>> Initialize *HOT\_POTATO\_USER\_COUNT*

>> Set *HOT\_POTATO\_USER\_COUNT* to *ten*

>> For i is less than *HOT\_POTATO\_USER\_COUNT*

>> Push *Users* to i

>> Print *Queue*

>> Initiliaze and Set *gameDuration* to Call *randRange(10,25)*

>> Initiliaze and Set *keepDuration* to Call *randRange(1,3)*

>> Initiliaze and Set *seconds t*o 1

>> While *Infinite Loop*

>> Clear Screen

>> Print *Queue*

>> Print “*Patates Users->front->value*”

>> Print “*Sıradaki Users->front->next->value*”

>> Print *“|MUZIK SURESI: gameDuration| ELDE TUTMA SURESI: keepDuration | GECEN SURE: seconds |”*

>> If seconds mod keepDuration is equal to zero

>> Set *keepDuration* to Call randRange(1,3)

>> Push *Users to popped value* of Users

>> Sleep one seconds.

>> Set *seconds* to seconds plus one

>> If *gameDuration* is equal to seconds

>> If *Users->front* is equal to NULL

>> Break While

>> Print *“| YANAN KISI: Users->front->value |”*

>> Pop from *Users*

>> Sleep two seconds

>> Set *seconds* to zero

>> Set *gameDuration* to Call *randRange(10,25)*

Programcı Kataloğu

Sabitler

Yazılımda bazı şeyleri daha hızlı düzenleyebilmek için bazı sabitler bulunmaktadır.

* *int* MAX\_NONCE

Sıralama ekranında en fazla kaç sayı alınacağını belirler.

* *int* HOT\_POTATO\_USER\_COUNT

Sıcak patates oyununda kaç tane oyuncu olacağını belirler.

* *char[]* *Names*

Sıcak patates oyunundaki isimlerin belirlendiği dizidir.

Fonksiyonlar

Yazılımda işlemler fonksiyoneldir.

|  |  |
| --- | --- |
| **void clear()** | |
| Ekranı silmeye yarar. Herhangi bir değer almaz. | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | Bilinmiyor |

|  |  |
| --- | --- |
| **int randRange(int min, int max)** | |
| Belirlenen iki değer arasında rassal değer üretir. | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | 2 \* 4 byte |

|  |  |
| --- | --- |
| **void mainMenu()** | |
| Ekrana ana menüyü çizer, kullanıcının isteğine göre farklı fonksiyonları çalıştırır. | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | 2 byte |

|  |  |
| --- | --- |
| **void askMainMenu()** | |
| Ekrana “Ana menüye gitmek ister misiniz” yazısını basar ve kullanıcının isteğine bağlı olarak, main menüyü çizer. | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | 2 byte |

|  |  |
| --- | --- |
| **QUEUE\_POINTER createQueue()** | |
| Yeni bir kuyruk yaratır. | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | sizeof(struct Queue) |

|  |  |
| --- | --- |
| **STACK\_POINTER createStack()** | |
| Yeni bir yığıt yaratır. | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | sizeof(struct Stack) |

|  |  |
| --- | --- |
| **NODE\_POINTER createNode()** | |
| Kuyruk ve yığıtta kullanılmak üzere yeni bir düğüm oluşturur. | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | sizeof(struct Node) |

|  |  |
| --- | --- |
| **void pushStack(STACK\_POINTER stack, int value)** | |
| Bir yığıta belirtilen değerde bir düğüm ekler. | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | sizeof(struct Node) + sizeof(struct Stack) |

|  |  |
| --- | --- |
| **void pushQueue(QUEUE\_POINTER queue,int value)** | |
| Bir kuyruğa belirtilen değerde bir düğüm ekler. | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | sizeof(struct Node) + sizeof(struct Queue) |

|  |  |
| --- | --- |
| **void printStack(STACK\_POINTER stack)** | |
| Bir yığıtı ekrana basar | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | sizeof(struct Stack) |

|  |  |
| --- | --- |
| **void printQueue(QUEUE\_POINTER queue)** | |
| Bir kuyruğu ekrana basar | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | sizeof(struct Queue) |

|  |  |
| --- | --- |
| **int popStack(STACK\_POINTER stack)** | |
| Bir yığıttan eleman çıkarır. | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | sizeof(struct Stack) |

|  |  |
| --- | --- |
| **int popQueue(QUEUE \_POINTER queue)** | |
| Bir kuyruktan eleman çıkarır. | |
| **Karmaşıklık** | **Bellek Gereksinimi** |
| Sabit | sizeof(struct Queue) |