BLG 221
Veri Yapıları

2010 Güz

Proje 2

Son Teslim Tarihi : 27.12.2010

Ad Soyad : Ozan Arkan Can

Numara : 040090573

Sınıf Eğitmeni : Gülşen ERYİĞİT

Asistan: Ahmet Aycan ATAK

CRN: 10950

1-Giriş

Bu projede bir RPG oyunu simule edilecektir. Simülasyon aşağıda belirtilen işleri yapmaktadir.

1)Sınıfı belirlenen karakteri oyun kurallarına bağlı kalarak oluşturur.

2)Kullanıcıdan aldığı boyutlarda rastgele labirent oluşturur ve oyun kurallarınca zorunlu olan nesneleri rastgele yerleştirir.

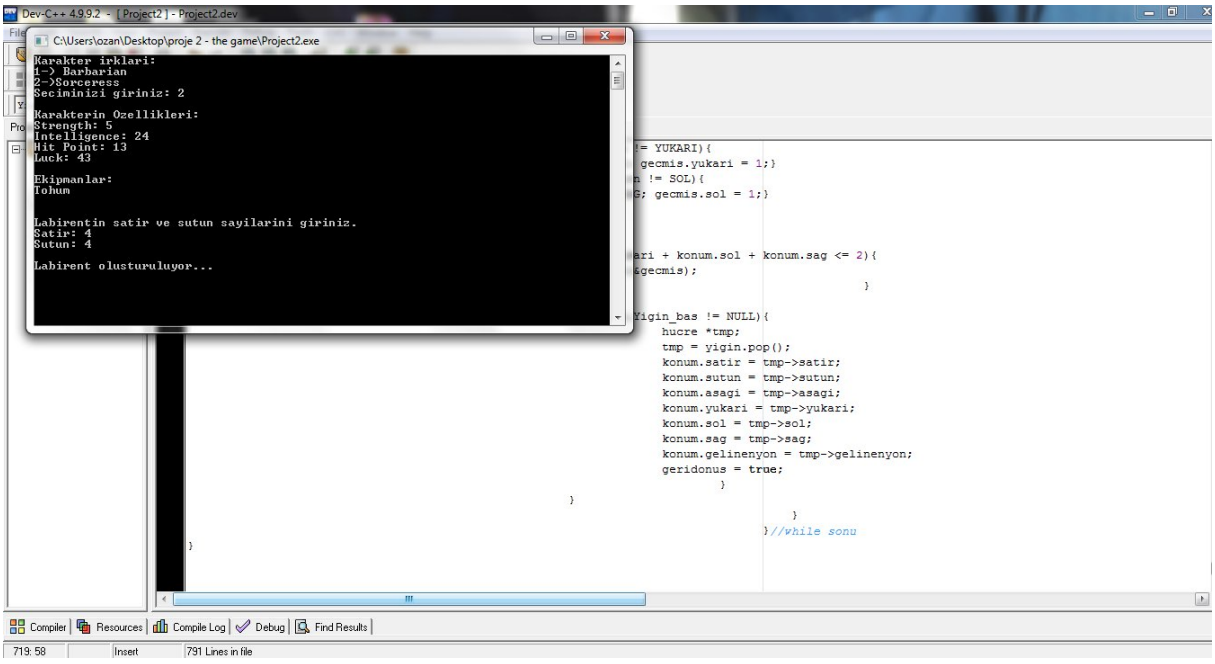
3)Herhangi bir yön bulma algoritmasını kullanarak karakterin labirenti dolaşmasını ve gerekli görevleri yerine getirmesini sağlar.

4)Her bir adımda gerçekleşen iler ekrana bastırılır.

2-Geliştirme ve Çalıştırma Ortamları

MS Windows

Kaynak kodu düzeltmede, derleme aşamasında ve çalıştırmada Dev-C++ kullanılmıştır.



3-Veri Yapısı

Veri Yapısı olarak proje de Kuyruk ve Yığın kullanılmıştır.

Karakter için struct:

```
struct karakter{
    char *irk;  char *saldiri_tipi; int strength; int intelligence; int hit_point;
    int saldiri_gucu; int luck; bool ekipman[4];  int elmas_cantasi; };
```

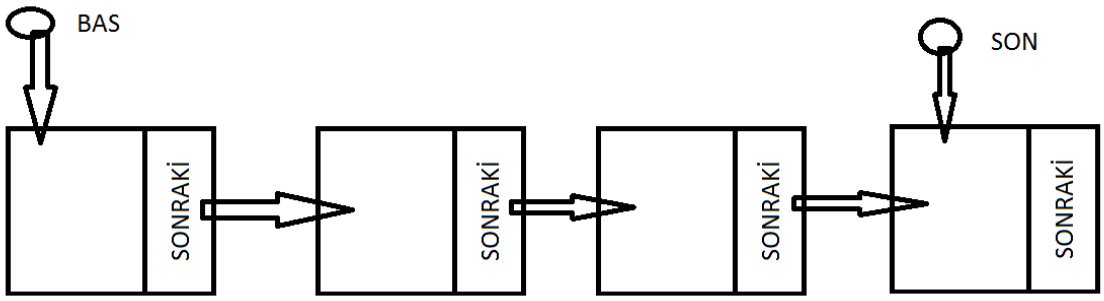
Yaratıklar için struct:

```
struct dusman{
    char *irk; char *saldiri_tipi; int strength; int intelligence;
    int hit_point; int saldiri_gucu; dusman *sonraki; int tur; };
```

Labirentin hücreleri için struct:

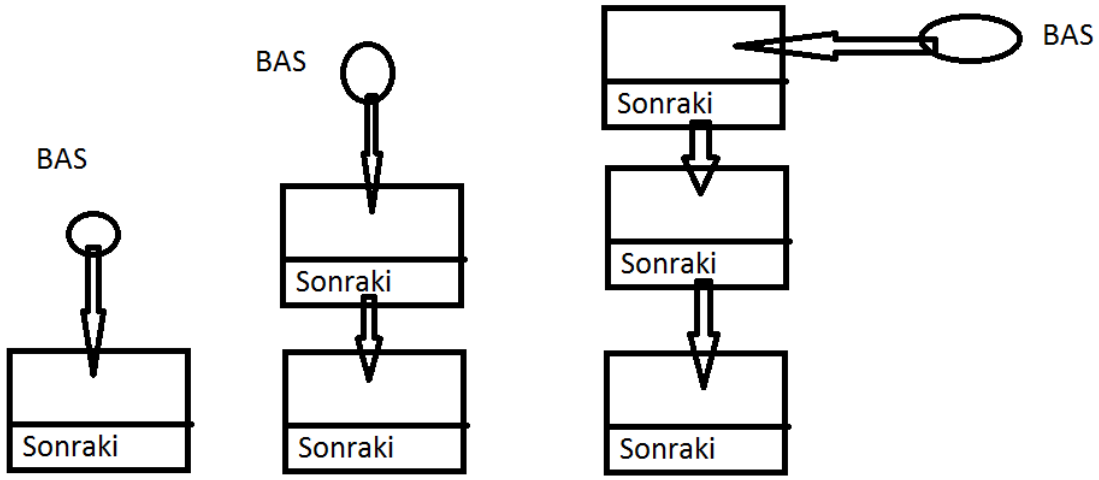
```
struct hucre{
    int yukari; int asagi; int sol; int sag; hucre *sonraki; bool durum;
    char karakter_burda; int satir; int sutun; int gelinenyon; char ozellik; };
```

Simulasyon sırasında öldürülen yaratıklar simulasyon bittikten sonra ekrana bastırılıyor. Bu işlem için öldürülen yaratıklar aşağıdaki kuyruk yapısında tutulmuştur.



Her yeni eklenecek yaratık sona eklenmiştir. Böylece ekrana bastırılırken öldürülüş sırasına uygun bastırılacaktır.

Labirent oluşturma da ve yön bulmada gerekli yerlerde aşağıdaki yığın yapısı kullanılmıştır. Bas işaretçisi her zaman yığının en üstündeki elemanı işaret eder böylece son giren elemana ilk ulaşabiliriz.



4)Kullanılan Algoritmalar

Program içinde gerekli yerlerde zar atışı yapılmıştır. Rastgele sayı üretmek için "Lineer Congruental Algoritma" kullanılmıştır. Algoritmanın basamakları aşağıdaki gibidir.

-Başlangıç kök sayısı bir sabitle çarp

-Başka bir sabitle topla

-Sonucu bir m modüle indirge

Labirent oluşturmak için depth-first search algoritmasının rekürsif backtracking implemantasyonu kullanılmıştır.

-Bulunulan hücreyi işaretle

-Bulunulan hücrenin ziyaret edilmemiş komşusu varsa

.Komşularından rastgele birini seç

.Bulunulan hücreyi yığına at

.Bulunulan hücreyle seçilen komşu arasındaki duvarı yık

.Seçilen komşu hücreyi bulunan hücre yap ve foksiyonu rekürsif olarak onun için çağır

-Bulunulan hücrenin ziyaret edilmemiş komşusu yoksa yığından çek ve fonksiyonu çekilen hücre için çağır

Yığının boş olması algoritmayı sonlandırıcı durumdur

5-Sonuç

Projede yığın ve kuyruk yapılarının programlarda nasıl uygulanacağı öğrenildi. Rekürsif programlama tekniğinin avantajları ve dezavantajları öğrenildi. Bu proje için labirent üretirken kullanılan rekürsif algoritma sebebiyle büyük boyuttaki labirent üretiminde bellek taşması olacağı için olumsuz sonuçlarla karşılaşılabilinir. Labirentte yol bulma algoritması olarak kullanılan algoritma çevreyle karşılaşıldığında sonsuz döngüye girerek labirentin tamamen gezilmesini engellemektedir.