**1fd75c3872a94e26ad68c7fa7667bdc82c07dd4ba85f4a6793f7a2b4e943b8fd**

**T.C.**

**SAKARYA ÜNİVERSİTESİ**

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

PROGRAMLAMA DİLLERİNİN PRENSİPLERİ ÖDEV RAPORU

**KOLONİ SAVAŞLARI**

**G201210033 – Sena Nur ERDEM**

**SAKARYA**

**Mayıs, 2023**

Programlama Dillerinin Prensipleri Dersi

KOLONİ SAVAŞLARI

Sena Nur ErdemG201210033-B2A[[1]](#footnote-1)

a Öğrenci numarası ve dersi aldığı grup

Özet

Kullanıcıdan bir kod alınacaktır. Bu kod boşluklar ile ayrılmış olacaktır. Girilen koddaki her sayı farklı bir kolonideki popülasyonu göstermekte ve bir sembol ile temsil edilmektedir. Her koloni taktik yapısından benzetim alan A ya da B taktiğinden rastgele birini kullanmaktadır. Savaşacak olan koloni kullandığı taktik yapısının savaş fonksiyonunu çağırmaktadır. Bu fonksiyonun döndüreceği değere göre hangi koloninin değeri büyükse savaşı o koloni kazanmaktadır. Her koloni üretim yapısından benzetim alan A ya da B üretiminden rastgele birini kullanmaktadır. Her turda döndürülen değerler kolonilerin yemek stoğuna eklenmektedir. Yemek stoğu ya da popülasyonu sıfıra veya eksiye düşen koloninin yaşamı sona erer. Program sadece bir koloni kalana kadar devam eder.

Her koloninin başlangıçta yemek stoğu ilk durumdaki popülasyonun karesi kadardır.

Her turun başında kolonilerin popülasyonu %20 oranında artar ve yemek stoğu (güncel popülasyon x 2 ) oranında azalır. Uretim yapısından gelen değerler yemek stoklarına eklenir.

Her turda bütün koloniler birbiriyle savaşmaktadır.

© 2017 Sakarya Üniversitesi.

Bu rapor benim özgün çalışmamdır. Faydalanmış olduğum kaynakları içeresinde belirttim. Her hangi bir kopya işleminde sorumluluk bana aittir.

Anahtar Kelimeler: koloni, savaş, taktik, üretim

1. GELİŞTİRİLEN YAZILIM

Her koloninin bir adı, popülasyon sayısı, yemek stoğu, kazanma ve kaybetme sayısı bulunur. Popülasyon ve yemek stoğu başlangıçta kullanıcıdan alınan girdilere dayanır. Oyun sınıfı ise kolonileri ve genel oyun durumunu yönetir. Her oyun, bir dizi Koloni yapısını saklar ve tüm koloniler arasındaki etkileşimleri kontrol eder.

**Koloni.h** ve **Koloni.c** dosyalarında bir Koloni yapı (struct) tanımlanmıştır. Bu yapı, bir koloniyi temsil eden bir dizi bilgiyi içerir: kazanma ve kaybetme sayıları, yaşayan popülasyon, toplam popülasyon, yemek stoğu, bir sembol ve taktik ve üretim stratejileri.

**koloni\_olustur** fonksiyonu, bir Koloni yapısını oluşturmak ve başlangıç değerlerini ayarlamak için kullanılır. Bu fonksiyon, belirli bir popülasyon boyutu, sembol, taktik ve üretim stratejisi ile bir koloni oluşturur. Popülasyon boyutunun karesi, başlangıç yemek stoğu olarak kullanılır.

**koloni\_yok\_et** fonksiyonu, bir Koloni yapısının belleğini boşaltmak için kullanılır. Bu, C'deki dinamik bellek yönetiminin bir parçasıdır. Bir Koloni yapısı oluşturulduğunda, ona ayrılan bellek manuel olarak boşaltılmalıdır, aksi takdirde bellek sızıntısına neden olabilir. Bu fonksiyon, hem Koloni yapısının kendisini hem de ona ait olan Taktik ve Uretim yapılarını boşaltır.

**Taktik.h** ve **Taktik.c** dosyaları, genel **Taktik** yapısını tanımlar. **Taktik** bir işaretçi fonksiyonuna sahiptir - **savas**. Bu işaretçi fonksiyon, bir taktiğin belirli bir savunma veya saldırı senaryosunda ne kadar etkili olacağını belirleyen bir değeri hesaplar.

**taktik\_olustur** fonksiyonu, yeni bir **Taktik** yapısı oluşturur ve bu yapıyı, **savas** fonksiyonuna bir işaretçi ile başlatır.

**ATaktik.h** ve **ATaktik.c** dosyaları, belirli bir **ATaktik** taktiğini tanımlar. **ataktik\_savas** fonksiyonu, 0 ile 1000 arasında rastgele bir çift sayı döndürür. Bu, **ATaktik**'in bir savaş durumunda ne kadar etkili olduğunu belirleyebilir.

**BTaktik.h** ve **BTaktik.c** dosyaları, belirli bir **BTaktik** taktiğini tanımlar. **btaktik\_savas** fonksiyonu, 1 ile 999 arasında rastgele bir tek sayı döndürür. Bu, **BTaktik**'in bir savaş durumunda ne kadar etkili olduğunu belirleyebilir.

**Uretim.h** ve **Uretim.c** dosyaları, genel **Uretim** yapılandırmasını tanımlar. **Uretim**, bir işaretçi fonksiyonuna sahiptir - **uret**. Bu işaretçi fonksiyon, bir üretim sürecinin ne kadar başarılı olduğunu belirleyen bir değeri hesaplar.

**uretim\_olustur** fonksiyonu, yeni bir **Uretim** yapısı oluşturur ve **uret** fonksiyonuna bir işaretçi ile başlatır.

**AUretim.h** ve **AUretim.c** dosyaları, belirli bir **AUretim** stratejisini tanımlar. **auretim\_uret** fonksiyonu, 1 ile 10 arasında rastgele bir tek sayı döndürür.

**BUretim.h** ve **BUretim.c** dosyaları, belirli bir **BUretim** stratejisini tanımlar. **buretim\_uret** fonksiyonu, 0 ile 10 arasında rastgele bir çift sayı döndürür.

**Oyun.h** dosyası, **Oyun** yapısını ve ilgili fonksiyonları tanımlar. **Oyun** bir işaretçi dizisine sahiptir - **koloniler**, bir tam sayı - **koloni\_sayisi**, ve bir tam sayı - **tur\_sayisi**.

**Oyun.c** dosyası, **Oyun** yapısını ve ilgili fonksiyonları tanımlar. Bu fonksiyonlar arasında **oyun\_olustur**, **oyun\_baslat**, **oyun\_durumunu\_yazdir** ve **oyun\_sonlandir** bulunur.

**oyun\_olustur** fonksiyonu, yeni bir **Oyun** yapısı oluşturur ve **koloniler**, **koloni\_sayisi** ve **tur\_sayisi** ile başlatır.

**oyun\_baslat** fonksiyonu, oyunun başlamasını ve her turun sonunda kolonilerin durumunun güncellenmesini sağlar.

**oyun\_durumunu\_yazdir** fonksiyonu, oyunun mevcut durumunu (her koloninin popülasyonu, yemek stoğu, kazanma ve kaybetme sayısı) ekrana yazdırır.

Her turda, tüm kolonilerin mevcut durumları (popülasyon, yemek stoğu, kazanma ve kaybetme sayıları) güncellenir ve ekrana yazdırılır. Oyun, sadece bir koloni kaldığında sona erer.

NOT :

Tur Sayısı : 0 durumundaki değerler hiç savaş olmamışken popülasyon ve yemek stoğu bilgilendirmesi yapmak adına ekrana yazdırılmıştır.

1. ÇIKTILAR

Örnek bir çıktı aşağidaki gibidir . Ancak unutulmamalıdır ki aynı değerler için tur sayısı farklı olabilir. Çünkü savaş değerleri rastgele gelmektedir.

senanurerdem@192 ColonyClash % make

gcc -I ./include/ -o ./lib/Taktik.o -c ./src/Taktik.c

gcc -I ./include/ -o ./lib/Koloni.o -c ./src/Koloni.c

gcc -I ./include/ -o ./lib/Oyun.o -c ./src/Oyun.c

gcc -I ./include/ -o ./lib/Uretim.o -c ./src/Uretim.c

gcc -I ./include/ -o ./lib/AUretim.o -c ./src/AUretim.c

gcc -I ./include/ -o ./lib/BUretim.o -c ./src/BUretim.c

gcc -I ./include/ -o ./lib/ATaktik.o -c ./src/ATaktik.c

gcc -I ./include/ -o ./lib/BTaktik.o -c ./src/BTaktik.c

gcc -I ./include/ -o ./bin/Test ./lib/Taktik.o ./lib/Koloni.o ./lib/Oyun.o ./lib/Uretim.o ./lib/AUretim.o ./lib/BUretim.o ./lib/ATaktik.o ./lib/BTaktik.o ./src/Test.c

./bin/Test

Popülasyonları girin (boşlukla ayrılmış şekilde): 10 20 30

Tur Sayısı: 0

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ 10 100 0 0

% 20 400 0 0

& 30 900 0 0

-------------------------------------------------------------

Tur Sayısı: 1

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ 6 36 0 2

% 24 454 2 0

& 33 780 1 1

-------------------------------------------------------------

Tur Sayısı: 2

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ 3 385 1 3

% 27 412 3 1

& 18 335 2 2

-------------------------------------------------------------

Tur Sayısı: 3

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ 1 194 2 4

% 8 90 3 3

& 21 748 4 2

-------------------------------------------------------------

Tur Sayısı: 4

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 9 144 5 3

& 15 426 4 4

-------------------------------------------------------------

Tur Sayısı: 5

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 9 112 5 4

& 18 415 5 4

-------------------------------------------------------------

Tur Sayısı: 6

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 10 219 6 4

& 15 265 5 5

-------------------------------------------------------------

Tur Sayısı: 7

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 11 185 6 5

& 18 254 6 5

-------------------------------------------------------------

Tur Sayısı: 8

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 13 270 7 5

& 11 112 6 6

-------------------------------------------------------------

Tur Sayısı: 9

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 12 197 7 6

& 13 142 7 6

-------------------------------------------------------------

Tur Sayısı: 10

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 10 120 7 7

& 15 174 8 6

-------------------------------------------------------------

Tur Sayısı: 11

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 12 143 8 7

& 13 110 8 7

-------------------------------------------------------------

Tur Sayısı: 12

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 11 94 8 8

& 15 112 9 7

-------------------------------------------------------------

Tur Sayısı: 13

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 4 24 8 9

& 18 131 10 7

-------------------------------------------------------------

Tur Sayısı: 14

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 4 44 9 9

& 16 74 10 8

-------------------------------------------------------------

Tur Sayısı: 15

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 4 60 10 9

& 11 23 10 9

-------------------------------------------------------------

Tur Sayısı: 16

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 3 46 10 10

& 13 12 11 9

-------------------------------------------------------------

Tur Sayısı: 17

Koloni Popülasyon Yemek Stoğu Kazanma Kaybetme

+ — — — —

% 3 39 11 10

& — — — —

-------------------------------------------------------------

1. SONUÇ

C dilinde nesne benzetimini ve C dili hakkında daha pek çok şey öğrendiğim bir proje oldu.

Referanslar

[1] https://www.youtube.com/watch?v=yMHGt2Bcsts&feature=youtu.be&themeRefresh=1

[2] https://www.youtube.com/watch?v=ov\_GudbL1qo

1. \* Ödev Sorumlusu. Sorumlu ad soyad, öğrenci no,

   *Mail Adresi:* sena.erdem1@ogr.sakarya.edu.tr [↑](#footnote-ref-1)