ALGORİTMA ANALİZİ 1. ÖDEVİ****

**Oğuz Mert Cerrahoğlu**

11011072

# Yöntem

Gezgin çekirge oyunu basit bir şekilde anlatılmak istenirse verilen bir dizide ilk elemandan başlayarak o indisteki değer kadar (dizi dışına çıkmayacak şekilde) sağa veya sola hareket edilen bir oyundur. Kazanma koşulu dizinin 0 değerini içeren son indisine ulaşmaktır.

Bu oyunu gerçeklerken, hareket sonlanmadıkça veya istenen yere gelmedikçe kendini çağıran bir rekürsif fonksiyon kullanmamız istenmiştir. Bu fonksiyonun initial conditionunu son indise ulaşmak olarak belirledim. Basic operation ise sağa veya sola harekettir. yani fonksiyon sona ulaşmadıysa kendini çağıracaktır.

Not: Bu program verilen değerlerin çekirgeyi her zaman dizi içinde tutacağını varsayarak yazılmıştır. Bu durumun ihtimaller dahilinde olmaması tasarımı değiştirir.

# 

# 

# 

# 

# 

# 

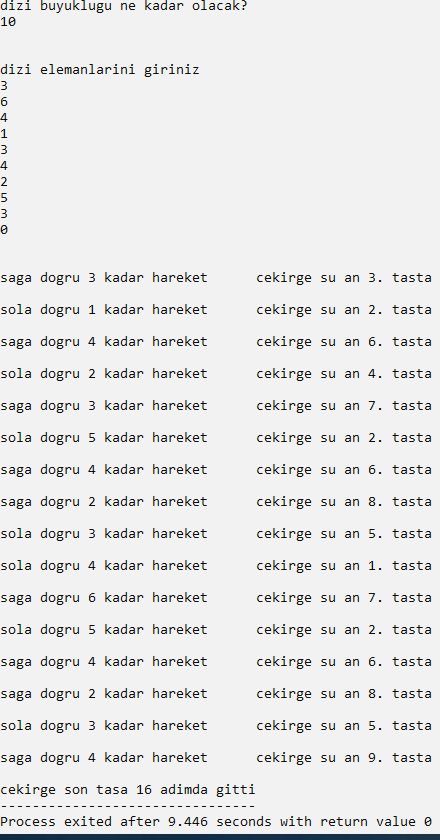
# 

# 

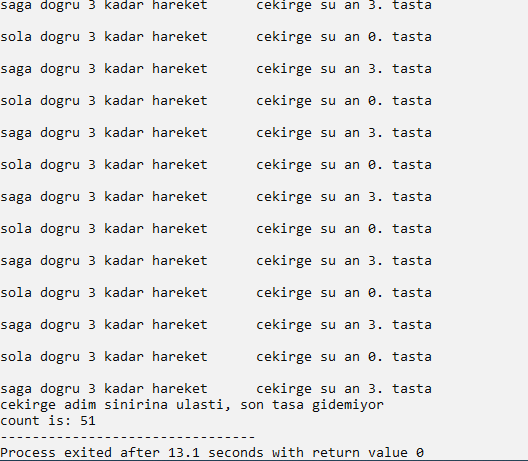
# 

# 2. Uygulama

Başarılı Örnek:



Başarısız Örnek:



# 

# 

# 

# 3. Kod

//oğuz mert cerrahoğlu 11011072

//not: bu program verilen sayıların çekirgeyi dizi dışına çıkarmadığını varsayar

//bir diğer değişle verilen dizler ya sonsuz döngüye girer ya da çekirge son adıma ulaşabilir.

#include<stdio.h>

#include<stdlib.h>

int solvable(int basla,int dizi[],int m );

int count=0;

int maxCount=50; //count fonksiyona kaç kere girildiğini tutar, maxcount ise fonksiyonun döngüye girme ihtimaline

//karşı en fazla kaç adım kere fonksiyona girilebiliri belirler

int main(int argc, char const \*argv[]) {

int n,i; //n dizi büyüklüğü için, i ise döngülerde kullanılmak üzere

printf("dizi buyuklugu ne kadar olacak?\n");

scanf("%d",&n);

int leapCount[n]; //dizi deklarasyonu

printf("\n");

printf("\ndizi elemanlarini giriniz\n");

for(i=0;i<n;i++){

scanf("%d",&leapCount[i]);

}

leapCount[n-1]=0; //son rakam 0'lanıyor (kullanıcı unutabilir vs)

if(solvable(0,leapCount,n)){

printf("\n");

printf("\ncekirge son tasa %d adimda gitti", count);

} else{

printf("\ncekirge adim sinirina ulasti, son tasa gidemiyor");

printf("\ncount is: %d",count);

}

return 0;

}

int solvable (int start,int leapCount[],int n){

if(count>maxCount){ //adım sınırına ulaşılmış mı kontrol

return 0;

}

if(start==(n-1)){ //initial condition

return 1;

} else{

if(start-leapCount[start]<0){ //sola gidersek dizi dışına çıkacağız o zaman sağa hareket

printf("\n");

printf("\nsaga dogru %d kadar hareket",leapCount[start]);

start+=leapCount[start];

printf("\tcekirge su an %d. tasta",start);

count++;

} else{

if(start+leapCount[start]>(n-1)){ // sağa gidersek dizi dışındayız, o zaman sola hareket

printf("\n");

printf("\nsola dogru %d kadar hareket",leapCount[start]);

start-=leapCount[start];

printf("\tcekirge su an %d. tasta",start);

count++;

}else{ //sağ veya sol bizi dizi dışına çıkarmıyor. o zaman rand fonksiyonundan gelen 0 veya 1'e göre hareket edelim

if((rand() % 2)==0){ //eğer 0 ise sağ, 1 ise sol

printf("\n");

printf("\nsaga dogru %d kadar hareket",leapCount[start]);

start+=leapCount[start];

printf("\tcekirge su an %d. tasta",start);

count++;

} else {

printf("\n");

printf("\nsola dogru %d kadar hareket",leapCount[start]);

start-=leapCount[start];

printf("\tcekirge su an %d. tasta",start);

count++;

}

}

}

} return (solvable(start,leapCount,n)); //basic operation

}