2019-2020 Güz Yarıyılı
Algoritma Analizi Final Projesi
Grup 2

Graf Üzerinde Arama İşlemi

Ahmet Onur Akman
16011059

Ödevin Konusu

Ödevde, bize verilen oyuncu ve film adları içeren text dosyasını okumamız, içeriğindeki elementleri ayırmamız, elementler arası bağlantılar kurmamız ve bu elementler arasında BFS metoduyla arama yaparak istenen aktörler arası path oluşturmamız ve uzaklıklarını ölçmemiz isteniyordu.

Algoritmamın Çalışma Mantığı

Yazdığım algoritmamın çalışma adımlarını basitçe sıralamam gerekirse,

- Node'larının arası bağlantıların Adjacency List formatında tutulduğu bir graf oluşturulur.
- Beklenen veri sayısının yüksek olmasından dolayı, yerleştirme işlemini hızlandırmak ve daha sonra erişimi kolay kılmak adına, önceki maddede bahsedilen Adjacency List, Hash Table fromundadır.
- Text dosyasından sırasıyla satırlar okunur.
- Bu satırlardaki filmler ve aktörler, isimlerindeki karakterlerin integer değerlerinin toplamının tablo boyutuna modu alınmasıyla oluşturulan key değeri doğrultusunda, hash table'a eklenir.
- Bu key değerlerinin yakın olması ve tablonun büyük boyutta olması sebebiyle, lineer probing değil, quadratic probing'e benzer olan, ancak her probing adımının karesinin son adımda elde edilen değere eklenerek ilerlendiği bir probing tarzının daha efektif ve hızlı çalıştığı gözlemlenmiş ve tercih edilmiştir. Bunu yaparak, büyük boyutlu bu tabloda her probing adımında daha da büyük adreslere varılması hedeflenmiştir.
  $H(key, i) = (h'(key) + \sum(i^2)) \bmod TABLESIZE$
- Aktörler eklenmeden önce, unique olup olmadıkları kontrol edilir. Bu işlem, hash table kullanmış olmamızın verdiği avantaj sayesinde hızlı sonuçlanır.
- Aktör ve filmlerin verileri, Node adında bir struct yapısında tutulur. Bu yapı, verinin ismini, komşuluk listesinin ilk pointer'ını, hash table'daki lokasyonunu, verinin tipini, BFS için visited olup olmadığı ve yine BFS için, Queue yapısına hangi node'un komşusu olduğu için eklendiği bilgisini saklar.
- Bir node'un komşuları, o node'a tanımlanmış bir linked list içinde tutulur. Bu listenin içindeki nodelar, hash table'daki orijinal versiyonlarının kopyaları niteliğindedir. Bütün verileri (key değeri dahil) aynıdır, ancak aslında gerçekten de hash table'da bulunmazlar. Ancak barındırdıkları key değeri sayesinde, onlar üzerinde orijinallerine erişmek kolay ve hızlıdır.
- Hash table oluşturulup grafımızın yapısı tamamlandıktan sonra, kullanıcının tercihine göre bir aktörün Kevin Bacon sayısını ya da verilecek iki aktör arası uzaklığı bulmak adına, uygun inputlar ile BFS fonksiyonu çağırılır.
- BFS fonksiyonu klasik BFS algoritmasına sahip bir fonksiyondur. Node'ların hash table'daki lokasyonunu tutan bir queue yapısını kullanır.

- BFS fonksiyonunda, biz dizi film dequeue edildikten sonra dequeue edilecek ilk aktör ile; ya da bir dizi aktör dequeue edildikten sonra dequeue edilecek ilk film ile, count değişkeni 1 arttırılır.
- Dequeue edilen verinin key değeri, input olarak verilmiş dest node'unun key değerine eşitse, BFS işlemi durur, az önce dequeue edilmiş node'un r değerinden yola çıkılarak, geriye doğru, start node'una kadar, path ekrana yazdırılır. Daha sonra count değerinin yarısı geri döndürülür. Bu, path uzunluğudur.
- Algoritma, BFS'e verilecek geçersiz inputlara karşı kullanıcıyı uyarır. Yeni işlem yapmasını ister ya da uyarı verip programı sonlandırır.
- Kullanıcı, istediği kadar BFS sorgusu yapabilsin diye, her BFS sorgusundan sonra kullanıcıya isteği sorulur. İstemedği takdirde program sonlanır.

Bir sonraki başlık altında programın Kevin Bacon sayısını bulma modu ve iki aktör arası uzaklık bulma modu için 10'ar tane ekran çıktısı gösterilmiştir.

Ekran Çıktıları

A) İki Oyuncu Arası Path Bulma Modu

```
Processing, please wait...

Reading the file 'input.txt'...

Scanning elements...

Completed! Total node count (Movies and Actors): 184647

Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>1


Please insert an actor name (surname, name) > Eminem

Please insert the second actor name (surname, name) > Dogg, Snoop Doggy

Calculating the distance...


Eminem >  Wash, The (2001)  >  Dogg, Snoop Doggy

Distance between 'Eminem' and 'Dogg, Snoop Doggy': 1


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>1


Please insert an actor name (surname, name) > Firdevs, Ekmekyemez

Please insert the second actor name (surname, name) > Affleck, Ben

Calculating the distance...


Firdevs, Ekmekyemez >  Abwörts (1984)  >  Hartley, Steven  >  Vampires (1998)  >  Boone Junior, Mark  >  Armageddon (1998)  >  Affleck, Ben

Distance between 'Firdevs, Ekmekyemez' and 'Affleck, Ben': 3


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>1


Please insert an actor name (surname, name) > Dr. Dre

Please insert the second actor name (surname, name) > Watson, Emma

Calculating the distance...


Dr. Dre >  Set It Off (1996)  >  Horsford, Anna Maria  >  Along Came a Spider (2001)  >  Baker, Dylan  >  Tailor of Panama, The (2001)  >  Radcliffe, Daniel  >  Harry P
otter and the Prisoner of Azkaban (2004)  >  Watson, Emma

Distance between 'Dr. Dre' and 'Watson, Emma': 4


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>1


Please insert an actor name (surname, name) > Armstrong, Darrell

Please insert the second actor name (surname, name) > Cube, Ice

Calculating the distance...


Armstrong, Darrell >  $ (1971)  >  Brady, Scott  >  Gremlins (1984)  >  Welker, Frank  >  Anaconda (1997)  >  Cube, Ice

Distance between 'Armstrong, Darrell' and 'Cube, Ice': 3


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>1


Please insert an actor name (surname, name) > Boo Khoo, Ian

Please insert the second actor name (surname, name) > Firdevs, Ekmekyemez

Calculating the distance...


Boo Khoo, Ian >  Tomorrow Never Dies (1997)  >  Watkins, Jason  >  Split Second (1992)  >  Hartley, Steven  >  Abwörts (1984)  >  Firdevs, Ekmekyemez

Distance between 'Boo Khoo, Ian' and 'Firdevs, Ekmekyemez': 3


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>1


Please insert an actor name (surname, name) > DiCaprio, Leonardo

Please insert the second actor name (surname, name) > Furlong, Kevin

Calculating the distance...


DiCaprio, Leonardo >  Poison Ivy (1992)  >  Skerritt, Tom  >  Heist, The (1989)  >  Furlong, Kevin

Distance between 'DiCaprio, Leonardo' and 'Furlong, Kevin': 2


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>1


Please insert an actor name (surname, name) > Xzibit

Please insert the second actor name (surname, name) > Xzibit

Calculating the distance...


You have given the same actor twice. Distance: 0


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>1


Please insert an actor name (surname, name) > Jacoby, Scott

Please insert the second actor name (surname, name) > Ekland, Britt

Calculating the distance...


Jacoby, Scott >  Baxter! (1973)  >  Ekland, Britt

Distance between 'Jacoby, Scott' and 'Ekland, Britt': 1


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>1


Please insert an actor name (surname, name) > Brost, Johannes

Please insert the second actor name (surname, name) > Holland, Tom

Calculating the distance...


Brost, Johannes >  –nglagÕrd (1992)  >  Brynolfsson, Reine  >  Jerusalem (1996)  >  Dukakis, Olympia  >  I Love Trouble (1994)  >  Faison, Frankie  >  Langoliers, The
1995)  >  Holland, Tom

Distance between 'Brost, Johannes' and 'Holland, Tom': 4


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>1


Please insert an actor name (surname, name) > Chen, Kuan Tai

Please insert the second actor name (surname, name) > Alexander, Alexis

Calculating the distance...


Chen, Kuan Tai >  Zou lao wei long (1993)  >  Chu, Paul  >  Murder in Mind (1997,I)  >  Smits, Jimmy  >  Fires Within (1991)  >  Ades, Daniel  >  Cold Heaven (1991)
 Alexander, Alexis

Distance between 'Chen, Kuan Tai' and 'Alexander, Alexis': 4


Would you like to try again? (1/0)
>
```

B) Verilen Oyuncunun Kevin Bacon Sayısını Bulma Modu

```
Processing, please wait...

Reading the file 'input.txt'...

Scanning elements...

Completed! Total node count (Movies and Actors): 184647

Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>0


Please insert an actor name (surname, name) > Shakur, Tupac

Calculating the distance...


Shakur, Tupac >  Bullet (1996)  >  Rourke, Mickey  >  Diner (1982)  >  Bacon, Kevin

Kevin Bacon value for 'Shakur, Tupac': 2


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>0


Please insert an actor name (surname, name) > Bacon, Kevin

Calculating the distance...


Kevin Bacon value for 'Bacon, Kevin': 0


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>0


Please insert an actor name (surname, name) > Jay-Z

Calculating the distance...


Jay-Z >  State Property (2002)  >  Burke, Robert John  >  Dust Devil (1992)  >  Matshikiza, John  >  Air Up There, The (1994)  >  Bacon, Kevin

Kevin Bacon value for 'Jay-Z': 3


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>0


Please insert an actor name (surname, name) > Arkin, Alan

Calculating the distance...


Arkin, Alan >  Little Murders (1971)  >  Sutherland, Donald  >  Animal House (1978)  >  Bacon, Kevin

Kevin Bacon value for 'Arkin, Alan': 2


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>0


Please insert an actor name (surname, name) > Eklund, Jacob

Calculating the distance...


Eklund, Jacob >  -nglagÖrd (1992)  >  Brynolfsson, Reine  >  Jerusalem (1996)  >  Dukakis, Olympia  >  Picture Perfect (1997)  >  Bacon, Kevin

Kevin Bacon value for 'Eklund, Jacob': 3


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>0


Please insert an actor name (surname, name) > Nero, Franco

Calculating the distance...


Nero, Franco >  Touch and Die (1991)  >  Galt, John William  >  JFK (1991)  >  Bacon, Kevin

Kevin Bacon value for 'Nero, Franco': 2


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>0


Please insert an actor name (surname, name) > Chan, Lung

Calculating the distance...


Chan, Lung >  She xing diao shou (1978)  >  Chan, Jackie  >  Shanghai Noon (2000)  >  Berkeley, Xander  >  Apollo 13 (1995)  >  Bacon, Kevin

Kevin Bacon value for 'Chan, Lung': 3


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>0


Please insert an actor name (surname, name) > Brooks, Jennifer

Calculating the distance...


Brooks, Jennifer >  Waxwork II: Lost in Time (1992)  >  Margolis, Ilona  >  Flatliners (1990)  >  Bacon, Kevin

Kevin Bacon value for 'Brooks, Jennifer': 2


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>0


Please insert an actor name (surname, name) > Alderson, Erville

Calculating the distance...


Alderson, Erville >  Spirit of St. Louis, The (1957)  >  Hamilton, Murray  >  1941 (1979)  >  Belushi, John  >  Animal House (1978)  >  Bacon, Kevin

Kevin Bacon value for 'Alderson, Erville': 3


Would you like to try again? (1/0)
>
```

```
Please insert one value from below:
0 for finding the Kevin Bacon number of an actor you will insert.
1 (or some other) for finding the distance between two actors you will insert.
>0


Please insert an actor name (surname, name) > Barry, Eugene

Calculating the distance...


Barry, Eugene >  Circus, The (1928)  >  Knight, Bill  >  Escape Through Time (1993)  >  Miller, Mike  >  Big Picture, The (1989)  >  Bacon, Kevin

Kevin Bacon value for 'Barry, Eugene': 3


Would you like to try again? (1/0)
>
```

Algoritmam hakkında daha fazla detaya ekteki c dosyasından erişebilir, kodda comment durumundaki printf satırlarını aktif ederek çalışma şeklini daha detaylı gözlemleyebilirsiniz.

Program Kodu

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAXSIZE 5000
#define TABLESIZE 190000

/* Fonksiyonlar icin gerekli aciklamalar declaration kısmında yapilmistir. Daha detayli
bilgiler, fonksiyon satirlarinda bulunabilir.
Ayrica, kodun calisma mantigini daha iyi anlamak adina, comment durumuna
alinmis printf satirlari kullanilabilir. */

typedef struct NODE{
        char *name;
        char type; //aktor veya film
        struct NODE *next; //baglantilari
        int key;
        char isVisited; //bfs icin
        int r; //sadece bfs'de kullanmasi icin. Queue'ya kimin komsusu oldugu icin eklendigi bilgisi.
}NODE;

typedef struct GRAPH{
        int nodeCount; //filmler + aktorler
        int hashSize; //veriler hashtable duzeninde saklanacak.
        struct NODE **AdjList;
}GRAPH;

typedef struct QUEUE { //BFS icin
   int items[TABLESIZE]; //node'larin key'larini tutar.
   int front;
   int rear;
}QUEUE;



NODE* createNode();
GRAPH* createGraph(int);
QUEUE* createQueue();
int isQEmpty(QUEUE*); //Queue fonksiyonlari
void enqueue(QUEUE*, int, GRAPH*);
int dequeue(QUEUE*, GRAPH*);
void printQueue(QUEUE*, GRAPH*);
void callTheBfs(GRAPH*); //iki aktorun uzakligi
void callKB(GRAPH*); //kevin bacon sayisi
int bfs(GRAPH*, NODE*, NODE*);
void readFile(char*,GRAPH*); //Dosya okuma ve graf doldurma fonksiyonlari
void locateElements(char*, GRAPH*); //okunan satirdaki verileri ayirip fonksiyonlara gonderir
int addList(NODE*,  GRAPH*); //veri unique ise hashtable'a ekler
void connect(NODE*, NODE*, GRAPH*); //film-aktor baglantisini yapar
NODE* getTheNode(char* ,  GRAPH*); //name verilince ayni name'e sahip node'u dondurur
int getKey(char*, GRAPH*); //name'den key degerini hesaplar
```

```
int main(int argc, char *argv[]) {
	int i=1, j;
	static char fileName[] = "input-mpaa.txt";
	GRAPH *gr = createGraph(TABLESIZE);

	printf("\nProcessing, please wait...");
	readFile(fileName,gr);
	printf("\n\nCompleted! Total node count (Movies and Actors): %d", gr->nodeCount);

	while (i){ //kullanici istedigi kadar bfs sorgusu yapabilir
		printf("\n\nPlease insert one value from below:");
		printf("\n0 for finding the Kevin Bacon number of an actor you will insert.");
		printf("\n1 (or some other) for finding the distance between two actors you will
insert.\n>");
		scanf("%d", &j);
		if (j) callTheBfs(gr); else callKB(gr);
		printf("\n\n\nWould you like to try again? (1/0)\n>");
		scanf("%d",&i);
		system("cls");
	}


	//bu alanı kullanarak ismini girdiginiz her node'un baglantilarini listeleyebilirsiniz.
	//kullanmak icin usttteki while'i commente donusturmelisiniz.
	/*char name[MAXSIZE]; printf("\n\n\nGive a name... "); gets(name); NODE* tmp = createNode();
	tmp->next=getTheNode(name, gr); printf("\nGot the node: '%s'",tmp->next->name);
	while (tmp->next->next!=NULL){ tmp->next = tmp->next->next; printf("\n%s",tmp->next->name);
}*/

	return 0;
}


NODE* createNode(){
	NODE *tmp = (NODE*)malloc(sizeof(NODE));
	tmp->next = NULL;
	tmp->name = (char*)malloc(MAXSIZE*sizeof(char));
	tmp->key = 0;
	tmp->r = -1;
	tmp->isVisited = '0';
	return tmp;
}

GRAPH* createGraph(int size){
	//printf("\n\nThe graph has been created...");
	int i;

	GRAPH *tmp = (GRAPH*)malloc(sizeof(GRAPH));
	tmp->nodeCount = 0;
	tmp->hashSize = size;

	tmp->AdjList=(NODE**)malloc(size*sizeof(NODE*));
	for (i=0;i<size;i++){
		tmp->AdjList[i] = createNode();
		tmp->AdjList[i]->key = -1;
	}
	return tmp;
}
```

```
QUEUE* createQueue(){
        QUEUE *tmp = (QUEUE*)malloc(sizeof(QUEUE));
  tmp->front = -1;
  tmp->rear = -1;
  return tmp;
}

int isQEmpty(QUEUE* q){
        if(q->rear == -1)
    return 1;
  else
    return 0;
}

void enqueue(QUEUE* q, int key, GRAPH *gr){ //enqueue ve dequeue'nin graph almasinin tek sebebi
printQueue'ye gonderebilmektir
        if(q->rear == TABLESIZE-1)
    printf("\nQueue is full!");
  else {
    if(q->front == -1) q->front = 0;
    q->rear++;
    q->items[q->rear] = key;
  }
  //istendigi takdirde kullanilabilir
  //printQueue(q, gr);
  //printf("\n");
}

int dequeue(QUEUE *q, GRAPH *gr){
        int dKey;
  if(isQEmpty(q)){
    //printf("\nQUEUE IS EMPTY");
    dKey = -1;
  }
  else{
    dKey = q->items[q->front];
    q->front++;
    if(q->front > q->rear){
      //printf("\nResetting queue");
      q->front = -1;
                                q->rear = -1;
    }
  }
  //istendigi takdirde kullanilabilir
  //printQueue(q, gr);
  //printf("\n");
  return dKey;
}

void printQueue(QUEUE* q, GRAPH* gr){ //program icin gerekliligi yoktur ama istenirse kullanilabilir
  if(isQEmpty(q))
    printf("\nQueue is Empty! Could not print.");
  else{
    int i;
    printf("\nQueue elements are:\n");
    for(i=q->front; i<=q->rear; i++){
    printf(" ; ");
    printf("%s",gr->AdjList[q->items[i]]->name);
        }
```

```
   }
}

void callTheBfs(GRAPH *gr){ //BFS fonksiyonu icin kullanicidan input alan ve bfs outputunu degerlendiren
fonksiyon
	getchar();
	int i, distance;
	char name[30],name2[30];

	printf("\n\nPlease insert an actor name (surname, name) > ");
	gets(name);
	//printf("\nGiven name: '%s'",name);
	printf("\nPlease insert the second actor name (surname, name) > ");
	gets(name2);
	//printf("\nGiven name: '%s'",name2);

	distance = bfs(gr, getTheNode(name2, gr), getTheNode(name, gr));

	if (distance==-1) printf("\n\nCould not find any path between '%s' and '%s'...", name, name2);
	else if (distance == 0) printf("\n\nYou have given the same actor twice. Distance: %d", distance);
	else if (distance == -2) printf("\n\nWARNING: Invalid input. Please try again and insert actors...");
	else printf("\n\nDistance between '%s' and '%s': %d", name, name2, distance);

	for (i = 0; i< gr->hashSize; i++){ //node'lar yeni bfs sorgusu icin hazir hale getiriliyor
		gr->AdjList[i]->r = -1;
		gr->AdjList[i]->isVisited = '0';
	}
}

void callKB(GRAPH *gr){ //Kevin Bacon modu
	getchar();
	int i, distance;
	char name[30];

	printf("\n\nPlease insert an actor name (surname, name) > ");
	gets(name);
	//printf("\nGiven name: '%s'",name);

	distance = bfs(gr, getTheNode("Bacon, Kevin", gr), getTheNode(name, gr));

	if (distance==-1) printf("\n\nCould not find the Kevin Bacon number for '%s'...", name);
	else if (distance == -2) printf("\n\nWARNING: Invalid input. Please try again and insert an actor...");
	else
	{
		if (distance > 6) printf("\n\nCould not find the Kevin Bacon number for '%s' (Distance value is
too high)...", name);
		else printf("\n\nKevin Bacon value for '%s': %d", name, distance);
	}
	for (i = 0; i< gr->hashSize; i++){ //node'lar yeni bfs sorgusu icin hazir hale getiriliyor
		gr->AdjList[i]->r = -1;
		gr->AdjList[i]->isVisited = '0';
	}

}
```

```
int bfs(GRAPH *gr, NODE *start, NODE *dest){ //distance degeri dondurur
        if((start->type == 'm') || (dest->type == 'm')) return -2; //ikisi de aktor mu?

        int count = 0;
        char prev, cur;
        QUEUE *q = createQueue();

        prev = 'n';
  cur = 'n';
  printf("\nCalculating the distance...\n");
  start->isVisited = '1';
  enqueue(q, start->key, gr);

  while(!isQEmpty(q)){
                int current = dequeue(q, gr);
                //printf("\nVisited %s", gr->AdjList[current]->name);
                //dequeue edilen verinin turu (film->aktor gibi) degisiyorsa count++
                prev = cur;
                cur = gr->AdjList[current]->type;
    if ((prev != 'n')&&(prev != cur)) count++;

    if (dest->key == current) //BULUNDU
                {
                        if (count == 0) return count;
                        NODE* result = createNode();
                        printf("\n\n%s > ",gr->AdjList[current]->name); //Path ekrana yazdirilir
                        result -> next = gr->AdjList[gr->AdjList[current]->r];
                        printf(" %s ",result -> next -> name);
                        while (result->next->key != start->key){
                                result -> next = gr->AdjList[result->next->r];
                                printf(" >  %s ",result -> next -> name);
                        }
                        return count/2; //distance dondurulur
                }

    NODE *tmp = createNode();
          tmp->next = gr->AdjList[current];

    while(tmp->next->next!=NULL) { //current'in komsularini kuyruga alalim
                tmp->next = tmp->next->next;
       if(gr->AdjList[tmp->next->key]->isVisited == '0'){
          gr->AdjList[tmp->next->key]->isVisited = '1';
          gr->AdjList[tmp->next->key]->r = current;
          enqueue(q, tmp->next->key, gr);
       }
    }
    free(tmp);
  }
  return -1; //bulunamadi
}

void readFile(char* fileName, GRAPH* gr){
        printf("\n\nReading the file '%s'...", fileName);
        FILE *file = fopen ( fileName, "r" );

  if ( file != NULL )
  {
          printf("\n\nScanning elements...");
          char line[MAXSIZE];
```

```
    while ( fgets ( line, sizeof line, file ) != NULL )
    {
                //printf("\nLine found:\n%s",line);
                locateElements(line,gr); //her satir bu fonksiyona gonderilir
    }
    fclose ( file );
  }
  else
  {
    perror ( fileName );
  }
}

void locateElements(char* str, GRAPH* gr){ //gelen satirdaki oyuncu ve filmler ayrılıp ilgili fonksiyonlar
cagiriliyor
        int i;
        const char *delim = "/";
        NODE *tmpM = createNode();
        tmpM->type='m';
        NODE *tmpA;


        char *element = (char*)malloc(MAXSIZE*sizeof(char));
        element = strtok(str, delim); //ilk element her zaman filmdir
        strcpy(tmpM->name,element);
        tmpM->key = getKey (tmpM->name, gr);
        tmpM->key = addList(tmpM,gr);
        //printf("\nMovie: %s",tmpM->name);

        while(1)
        {
                element = strtok(NULL, delim);
                if (element != NULL)
                {
                        //printf("\nActor found: %s",element);
                        i = 0;
                        while (element[i] != '\0')
                        {
                                if (element[i] == '\n') element[i] = '\0';
                                i++;
                        }
                        tmpA = createNode();
                        tmpA->type = 'a';
                        strcpy(tmpA->name,element);
                        tmpA->key = getKey (tmpA->name, gr);

                        //bulunan aktor listeye eklenir, daha sonra film ile baglanir
                        tmpA->key = addList(tmpA,gr);
                        connect(tmpM, tmpA, gr);
                        connect(tmpA, tmpM, gr);
                        free(tmpA);
                }
                else break;
        }
}

int addList(NODE* node,  GRAPH* gr){ //bulunan elementler hashtable'a ekleniyor
        int tmpKey = node->key, i = 0;
        if (node->type == 'a'){ //aktorse unique mi diye bakilir, filmse zaten uniquetir
```

```
		while ((gr->AdjList[tmpKey]->key!=-1)  &&  (strcmp(node->name, gr->AdjList[tmpKey]->name)))
		{
			i++;
			tmpKey = tmpKey + (i * i); //ben bu basit probing yonteminin efektif calistigini gozlemledim
			tmpKey = tmpKey % (gr->hashSize);
		}
		//printf("\ncheck DONE");
		if (gr->AdjList[tmpKey]->key!=-1){
			//unique degil, zaten mevcut
			//printf("\nALREADY EXISTS: %s",node->name);
			return tmpKey;
		}
		else
		{
		//unique, ekleniyor
		strcpy(gr->AdjList[tmpKey]->name,node->name);
		gr->AdjList[tmpKey]->type = node -> type;
		gr->AdjList[tmpKey]->key = tmpKey;
		gr->nodeCount++;
		//printf("\nADDED: %s",gr->AdjList[tmpKey]->name);
		return tmpKey;
		}
	}

	//filmler burada ekleniyor. sadece probing yapılıyor
	while (gr->AdjList[tmpKey]->key!=-1){
		i++;
		tmpKey = tmpKey + (i * i);
		tmpKey = tmpKey % (gr->hashSize);
	}
	strcpy(gr->AdjList[tmpKey]->name,node->name);
	gr->AdjList[tmpKey]->type = node -> type;
	gr->AdjList[tmpKey]->key = tmpKey;
	gr->nodeCount++;
	//printf("\nADDED: %s",gr->AdjList[tmpKey]->name);
	return tmpKey;
}

void connect(NODE* a, NODE* b, GRAPH *gr){ //verilen a node'undan b node'una baglanti kurar
	//printf("\nCONNECTING: '%s' ve '%s'",a->name,b->name);
	NODE *tmp, *tmp2;

	tmp = createNode();
	tmp->type = b->type;
	strcpy(tmp->name,b->name);
	tmp->key = b->key;

	tmp2 = createNode();
	tmp2->next = gr->AdjList[a->key];
	while (tmp2->next->next != NULL) tmp2->next = tmp2->next->next;
	tmp2->next->next = tmp;
}

NODE* getTheNode(char* name,  GRAPH* gr){ //verilen isme sahip node'u dondurur
	int i = 0;
	int key = getKey(name, gr);
	if (gr->AdjList[key]->key==-1){
```

```
                printf("\n\nCould not find: '%s'. Press any to terminate...", name);
                getch();
                exit(0);
        }
        while (strcmp(name,gr->AdjList[key]->name))
        {
                i++;
                key = key + (i * i);
                key = key % (gr->hashSize);
                if (gr->AdjList[key]->key==-1){
                        printf("\n\nCould not find: '%s'. Press any to terminate...", name);
                        getch();
                        exit(0);
                }
        }
        return (gr->AdjList[key]);
}

int getKey(char* name, GRAPH *gr){
        int i = 0, key = 0;
        while (name[i]!='\0'){
                key = key + (int)name[i];
                i++;
        }
        key = key % (gr->hashSize);
        return key;
}
```