BİLECİK ŞEYH EDEBALİ ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
BM208 VERİ YAPILARI VE ALGORİTMALAR DERSİ
UYGULAMA FÖYÜ

9. Deney Föyü: Sıralama Algoritmaları

Adı Soyadı:		Öğrenci No:		14.05.2019

1. Amaç:

Dizi veri yapısından ağaç veri yapısına kadar yaygınlaşmış bazı sıralama algoritmalarını uygulamak.

2.Uygulama Dersinden Önce Yapılması Gerekenler

Sıralama algoritmalarının önerilen kaynaklardan incelenmesi ve uygulanması gerekmektedir.

3.Uygulamaya Hazırlık:

Bir çok uygulamada veri dizilerinin, belirli bir sırada olması istenir. Veriler sayısal ise küçükten büyüğe veya büyükten küçüğe, alfa sayısal ise A'dan Z'ye veya Z'den A'ya doğru sıralanabilir. Sıralı verile üzerinde işlem ve değerlendirme yapmak çok daha kolaydır.

Herhangi bir problem için sıralama ve arama algoritmalrının seçiminde verilerin toplam sayısı ve uzunluğu ile verilerin tipi oldukça önemlidir. Ayrıca sıralamaya uygun olmayan veri sayısının toplam sayıya oranı ile bu sırasız verilerin konumuda algoritmanın performansını etkileyecektir. Kaynaklardan kolayca bulacağınız, en yaygın sıralama algoritmaları:

- Bubble Sort: Kabarcık Sıralama
- Selection Sort: Seçme Sıralama
- Quick Sort: Hızlı Sıralama
- Merge Sort: Birleştirmeli Sıralama
- Insertion Sort: Araya Eklemeli Sıralama

şeklindedir. Aşağıdaki örnek uygulamada "Kabarcık Sıralama" algoritmasının detayları verilmiştir.

4.Örnek Uygulama Kodu: Kabarcık Sıralama Algoritması

Bu algoritma yer değiştirme sıralaması olarak da isimlendirilmektedir. Günümüzde kabarcık sıralaması olarak adlandırılmasının nedeni dizi içindeki *büyük elemanların algoritmanın her adımında dizi sonuna doğru doğrusal olarak ilerlemesidir*. Bunu daha iyi anlayabilmek için {9,5,8,3,1} dizisinin azalan şekilde sıralamasını kabarcık algoritması ile sağlayalım:

1. Adım:

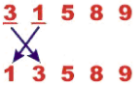
Burada görüldüğü gibi dizinin en büyük elemanı olan 9 kendinden bir önceki elemanla karşılaştırılarak dizinin en sonuna yerleştiriliyor.

2. Adım:

5 ve 8 karşılaştırılıyor 5, 8'den büyük olmadığı için yerinde kalıyor. Daha sonra 1. Adımdaki lineer bir süreç devam ediyor.

3. Adım

4. Adım



Görüldüğü dizinin eleman sayısının 1 eksiği kadar adım sayısı bulunmaktadır. Kabarcık sıralama algoritmasının işletiminde dizinin elemanları üzerinden tekrar tekrar geçilir ve her geçişte ardışık iki eleman karşılaştırılır. Yanda verilen örnek kodu inceleyiniz.

```
10 adet tam sayi giriniz:
95
156
32
4
1462
91
54
61
37
58
Girdiginiz dizi elemanlari:
 95
156
32
4
1462
91
54
61
37
58
Siralanmis Dizi :
4
32
37
54
58
61
91
95
156
1462
```

Ekran çıktısı yandaki gibi olan kullanıcıdan 10 adet sayı girilerek "Bubble Sort" yöntemiyle bu sayıları sıralayan programın kodu aşağıdaki gibidir, inceleyiniz.

```
#include <stdio.h>
#include<conio.h>
using namespace std;
void main()
{
int hold;
int array[10];
printf("10 adet tam sayi giriniz: \n");

for(int i=0; i<10; i++) {
      scanf("%d",&array[i]);

}

printf("Girdiginiz dizi elemanlari:\n ");
for(int j=0; j<10; j++)
{
      printf("%d \n",array[j]);

}

for(int i=0; i<9; i++)
{
for(int j=0; j<9; j++)
{
if(array[j]>array[j+1])
{
hold=array[j];
array[j]=array[j+1];
array[j+1]=hold;
}
}
}
printf("Siralanmis Dizi :\n");

for(int i=0; i<10; i++)
{
printf("%d\n",array[i]);

}
getch();
}
```

Uygulama Sorusu:

Değerlendiren Öğretim Elemanı:		Çalışma Durumu:	0	1	2	3	4	5	Not: