Derleme komutları

1) gcc ogrencinumarasi.c
./a.out<input1> myOutput1
Bu komut programınızın çıktısını myOutput1 dosyasına kaydeder.

2) diff myOutput1 output1
Bu komutu kullanarak kendi çıktınız ile olması gereken çıktıyı karşılaştırınız. Bu komutu girdikten sonra ekranda bir uyarı çıkmıyorsa, programınız bu değerler için doğru çalışıyor demektir. Eğer komutu girdikten sonra komut sisteminde uyarı görüyorsanız bu çıktınızda problem olduğunu gösterir, kodunuzu düzeltmeniz gerekmektedir.

SORU 3: Kullanıcıdan dizinin eleman sayısını ve dizinin elemanlarını alan ve bu diziyi küçükten büyüğe doğru aşağıda belirtilen kurallar çerçevesinde sıralayan bir **C** kodu yazınız. Kodunuzda aşağıdaki belirtilen fonksiyonları istenilen şekilde kullanmak **zorunludur!**

3 temel aşama söz konusudur.

1. Dizinin herhangi bir elemanını işaretleyin. (Soruda bu eleman **mid_element** olarak isimlendirilmiştir)

2. Ardından dizini **1 , ... , i-1** ‘e kadar olan elemanlar **mid_element** den küçük ya da eşit, **i+1,. . . , n** ‘e kadar olan elemanlar **mid_element** den büyük ya da eşit olacak şekilde düzenleyin.

3. **1, . . . , i - 1** ve **i +1, . . , n** aralığındaki elemanları bu stratejiyi kullanarak sıralayın.

Tüm algoritma **low, high** indisleri ile yürütülmektedir. **low** başlangıçta ilk elemanı **high** ise son elemanı göstermektedir. **mid_element** olarak eleman seçmek demek işaretlenecek elemanı bu değişkende tutmak demektir. Böylelikle dizinde bir boşluk yaratmış olacaksınız. Ardından **high** indisini sağdan sola doğru **mid_element** den daha küçük bir elemanı gösterinceye kadar kaydıracaksınız.
Bu durumda **low** indisinin gösterdiği kutuya **high** indis tarafından gösterilen elemanı kopyalayacaksınız. Bu durum **high** ile gösterilen yeni bir boşluk yaratacaktır. Ardından soldan sağa doğru **low** indisini yürüteceksiniz ve **mid_element** den daha büyük bir eleman arayışı devam edecek. Böyle bir eleman bulduğunuzda bu eleman **high** indisinin gösterdiği boşluğa taşınacak. Bu durum **low** ve **high** indisleri karşılaşıncaya kadar tekrarlanacak. İndisler aynı noktada buluştuklarında bir orta nokta bulmuş olacaksınız. Her iki indis de bu boşluğu gösterdiğinde **mid_element** diye isimlendirdiğiniz elemanı buraya taşıyacaksınız. Sağ kısım **mid_element** den büyük, sol kısım ise **mid_element** den küçük değerler ile düzenlenmiş olacaktır.

Örneğin;

low ilk elemanı **high** son elemanı göstermektedir.

12	3	6	18	7	15	10
low						high

ilk eleman 12 **mid_element** olarak seçilmiştir. Bunu bir değişkende tutup, **low** indisli adreste bir boşluk oluşturduğumuzu varsayalım.

	3	6	18	7	15	10

low high

Şimdi **high** indisli elemanın değerini 10 ile 12'yi karşılaştıralım. 10<12 olduğundan 10 değerini boşluğa taşıyın ardından **low** indisini 1 adım yürütün. **high** indisli yerde bir boşluk yaratıldı.

10	3	6	18	7	15	

low high

low 3 değerini gösteriyor. Şimdi **low** ile 12 'yi karşılaştırın. 3<12 harekete gerek yok. **low** indisini 1 adım yürütün.

10	3	6	18	7	15	

low high

low 6'yı gösteriyor. 6 ile 12'yi karşılaştırın. 6<12 harekete gerek yok **low** indisini 1 adım yürütün.

10	3	6	18	7	15	

low high

low 18'i gösteriyor. 18>12, 18'i boşluğa taşıyıp **high** indisini 1 adım sola yürütün. **low** indisli yerde bir boşluk yaratıldı.

10	3	6		7	15	18

low high

high 15'i gösteriyor. 15>12 harekete gerek yok **high** indisini 1 adım sola taşıyın.

10	3	6		7	15	18

low high

high 7'yi gösteriyor. 7'yi boşluğa taşıyıp **low** indisini 1 adım sağa yürütün.

10	3	6	7		15	18

low,high

low ve **high** eşit olduğundan 12'yi boşluğa yerleştirin.

10	3	6	7	12	15	18

Bu durumda 12'nin solunda kalan tüm elemanlar 12'den küçük ya da eşit, sağındakiler ise 12'den büyük ya da eşit oldu. Dizin ikiye bölündü gibi düşünebilirsiniz. Şimdi iki parçaya da aynı algoritmayı uygulayarak **(10,3,6,7)** ve **(15,18)** tüm dizini sıraya sokabilirsiniz (Böl ve yönet stratejisi).

Dolayısıyla **recursive** olarak bu işlemi tekrar etmelisiniz.

Not: Bu stratejide amaç orta nokta bulmaktır (aşağıdaki kod satırında **middle** olarak adlandırılan değişken). Recursive olarak bu noktanın önü ve arkası için aynı strateji uygulanacak ve böylelikle tüm dizin sıralanmış olacaktır. Yani önce dizini parçalayıp ardından düzenleme işi şeklinde düşünebilirsiniz.
Her bir iterasyonda l**ow** ya da **high** indisli elemanın boşluğa taşınıp taşınmayacağına ve indislerin birer adım uygun yöne yürüyüp yürümeyeceğine karar veriyorsunuz.

İkinci olarak aşağıdaki kod satırında **mid_element** olarak adlanırılan ve ortaya yerleştirilecek elemanın seçimi vardır. Bu eleman için farklı seçim önerileri ve yeni stratejilerde olabilimekle birlikte bu soruda basitlik için ilk eleman olarak seçilmiştir. Bu durum algoritmanın performansını değiştirecektir ancak şu an eleman sayısı az bir örnek üzerinde çalıştığımızdan herhangi bir sorun söz konusu değildir.

Kullanılacak fonksiyonlar:

void my_sort (int a [ ] , int low, int high) recursive sıralama fonksiyonu

int divide (int a [ ] , int low, int high) diziyi 2'ye bölme fonksiyonu

Program şu şekilde bir görünecektir.

```
void my_sort ( int a [ ] , int low, int high );
int divide ( int a [ ] , int low, int high );

int main( ){
	int size;
	scanf("%d", &size);
	int a[size];

	for(int i=0:i<size;i++)
		scanf( “%d”, &a[i] );

	my_sort ( a, 0, size-1 );

	for(i=0:i<size;i++)
		printf( “%d ”, a[i] );

	return 0;
}
void my_sort ( int a [ ] , int low, int high ) {
	int middle ;
	if ( low >= high)
		return ;

	middle = divide ( a, low, high ) ;
	my_sort ( a, low, middle - 1 ) ;
	my_sort ( a, middle + 1, high ) ;
}
```

```
int divide ( int a [ ] , int low, int high ) {
        int mid_element = a [ low ] ;
        ….
        ….
        ….
}
```

Örnek I/O:

Input:

7	# dizinin eleman sayısı
12 3 6 18 7 15 10	# dizinin eleman değerleri (eleman değerleri arasında bir boşluk var)

Output:

3 6 7 10 12 15 18	# dizinin sıralanmış hali (eleman değerleri arasında bir boşluk var)

Not: Sorunun tamamen yukarıda verilen sözde koda (pseucode) uygun şekilde, istenilen fonksiyonlar kullanılarak çözülmesi **zorunludur!** Farklı bir yöntem ile sıralama yapmanız halinde çıktınız birebir aynı olsa bile **puan verilmeyecektir!**

Sorularınızı sınav saati içinde **bsoylu@ankara.edu.tr** adresine gönderebilirsiniz.