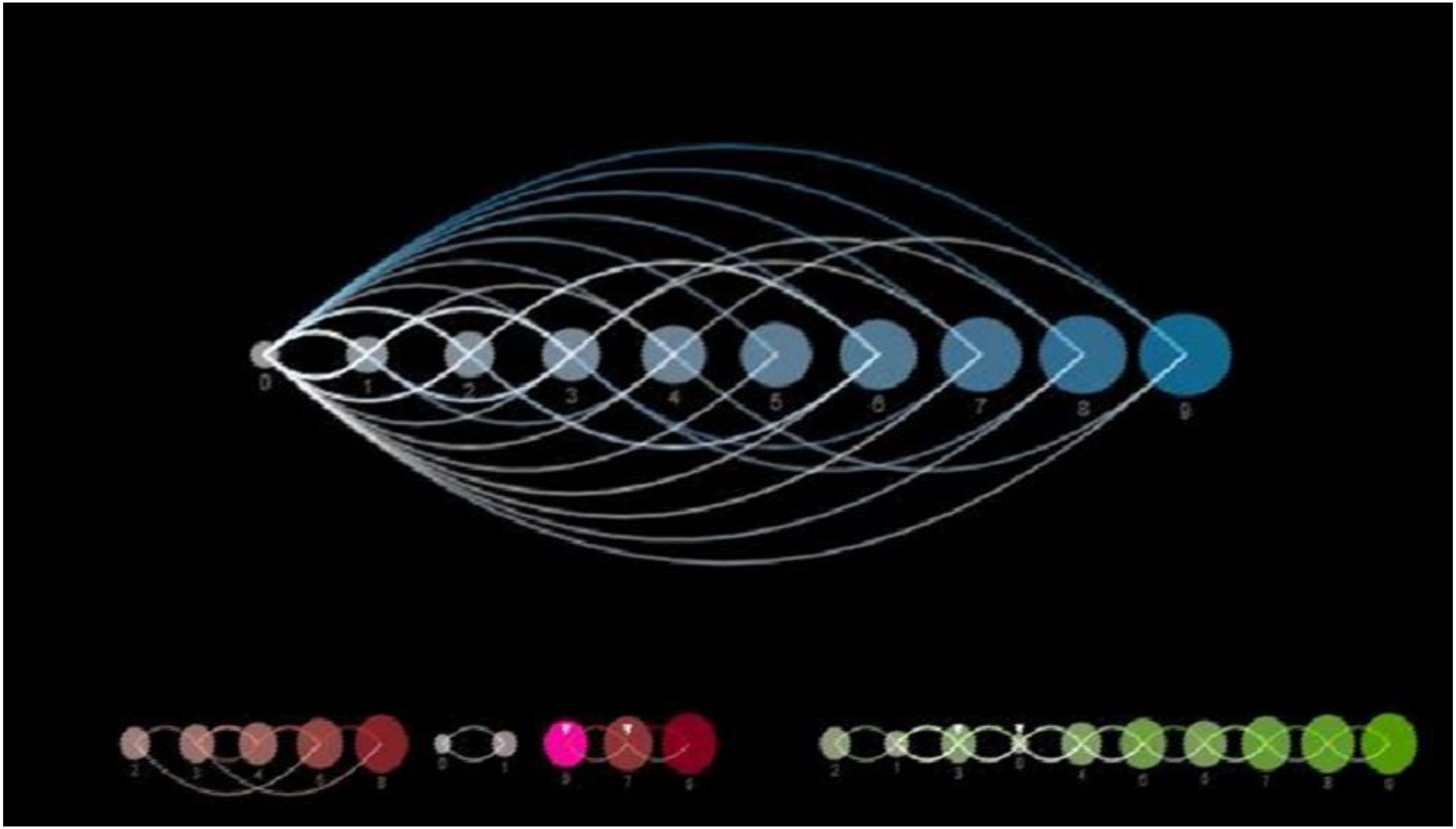
Sıralama Algoritmaları


0
1
2
3
4
5
6
7
8
9

Sıralama Algoritmaları

Herhangi bir sayıdaki verilerin

sınırlı bellek ve

işlem gücü ile belirli bir sıraya göre dizilmesidir.

Burada önemli olan: en az bellek kullanmak ve

en hızlı performans

verecek bir algoritmanın elde edilmesidir.

Sıralama Algoritmaları

Bellek Kullanımı: Çalışırken ek belek ihtiyacı duyan algoritmalarda kullanılır.

Ayrıcada sıralama işleminin yapılması sırasında hafızanın kullanımına göre de:

Harici sıralama (External Sort) ve

Dahili Sıralama (Internal Sort).

Sıralama Algoritmaları

Bellek Kullanımı: Çalışırken ek belek ihtiyacı duyan algoritmalarda kullanılır.

Ayrıcada sıralama işleminin yapılması sırasında hafızanın kullanımına göre de:

Harici sıralama (External Sort) ve

Dahili Sıralama (Internal Sort).

Sıralama Algoritmaları

Hesaplama Karmaşıklığı : Oluşturulmuş olan algoritmanın yaptığı işlem sayısıdır.

iyi(best),

ortalama(average) ve

en kötü(worst) durumu olarak belirtilir.

Ne kadar çok işlem yapılırsa o kadar uzun süre geçer ve dolayısı ile algoritmanın işleyiş süresini de etkiler.

Sıralama Algoritmaları

Yerdeğiştirmenin karmaşıklığı: İçerisinde ek bellek kullanmayan algoritmalarda kullanılan karşılaştırılabilmesi için önemlidir. Ne kadar az yer değiştirme o kadar performans demektir

Durağanlık(stability): Algoritmanın uygulanması sırasında sıralanmış bir verinin tekrar sıralamaya dahil edilmesidir.

Sıralama Algoritmaları

Fakat en önemli kriter:

Bellek (Hafıza) Verimliliği (Memory efficiency) ve

Zaman Verimliliği (Time efficiency)'dir.

* Bir algoritmanın hızlı çalışması demek daha çok hafızaya ihtiyaç duyması demektir.

* Tersi durumda da bir algoritmanın daha az yere ihtiyaç duyması daha yavaş çalışması demektir.

Ancak bir algoritma hem zaman hem de hafıza olarak verimliyse bu durumda diğer algoritmalardan başarılı sayılabilir.

Elemeli Sıralama,Kabarcık Sıralama(Bubble Sort)

Dizinin elemanları üzerinden ilk elemandan başlayarak ve her geçişte sadece yan yana bulunan iki eleman arasında sıralama yapılır. Dizinin başından sonuna kadar tüm elemanlar bir kez işleme tabi tutulduğunda dizinin son elemanı (küçükten büyüğe sıralandığında) en büyük eleman haline gelecektir.

Elemeli Sıralama,Kabarcık Sıralama(Bubble Sort)

Bir sonraki tarama ise bu en sağdaki eleman dışarıda bırakılarak gerçekleştirilmektedir. Bu dışarıda bırakma işlemi de dış döngüdeki sayaç değişkeninin değerinin her işletimde bir azaltılmasıyla sağlanmaktadır.

Elemeli Sıralama, Kabarcık Sıralama(Bubble Sort)

Elemeli Sıralama,Kabarcık Sıralama(Bubble Sort)

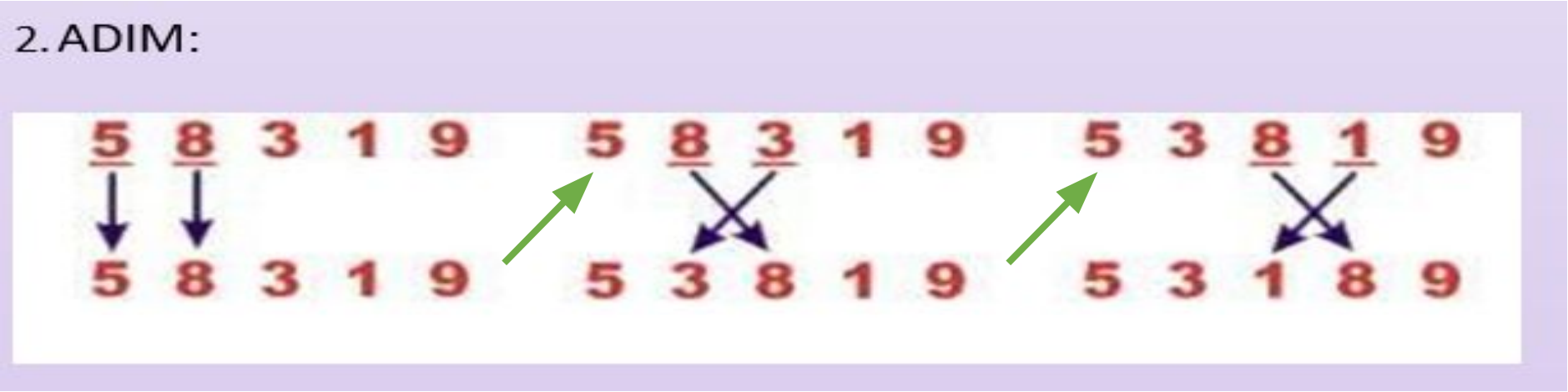
Bubble sort, sıralama teknikleri içinde anlaşılması ve programlanması kolay olmasına rağmen etkinliği en az olan algoritmalardandır (n elemanlı x dizisi için).

Elemeli Sıralama,Kabarcık Sıralama(Bubble Sort)

9, 5, 8, 3, 1. rakamlarının azalan şekilde sıralanmasını buble sort algoritmasıyla yapalım:

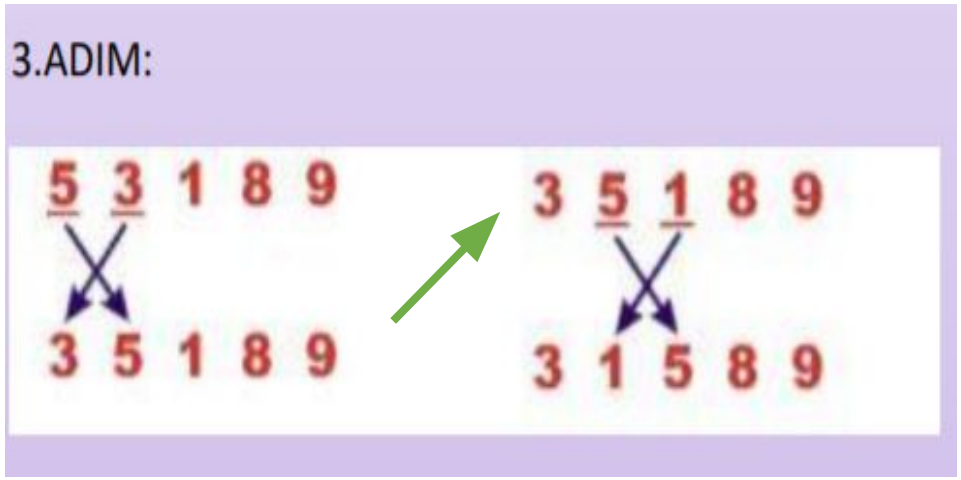
Elemeli Sıralama,Kabarcık Sıralama(Bubble Sort)

9, 5, 8, 3, 1. rakamlarının azalan şekilde sıralanmasını buble sort algoritmasıyla yapalım:



Elemeli Sıralama,Kabarcık Sıralama(Bubble Sort)

9, 5, 8, 3, 1. rakamlarının azalan şekilde sıralanmasını buble sort algoritmasıyla yapalım:



Seçmeli sıralama (selection sort)

- Mantıksal olarak ilk önce dizi sıralanmış ve sıralanmamış 2 parçaya bölünür.
- Başlangıçta tüm dizi sıralanmamıştır.
- Konumunu başlangıca ayarla (i. eleman)
- i. elemandan başlayarak dizinin kalan elemanları ile karşılaştırma yaparak en küçük elemanını bul/seç.
- i. Elemanla değiştir.
- Konumunu 1 artır. (i+1. elemana gel)
- i+1. elmandan başlayarak en küçüğü bul
- i+1. elemanla yer değiştir.

```
#include <stdio.h>
int main()
{int dizi [100],n,c,d,degistir;
printf("Kaç sayı Sıralanacak: "); scanf("%d",&n);
printf("%d sayı giriniz ve her girdiğiniz sayıdan sonra ENTER a basınız\n",n);
for(c=0;c<n;c++) scanf("%d",&dizi[c]);
for(c=0;c<n-1;c++)
{
   for(d=0;d<n-c-1;d++)
      {
         if(dizi[d]>dizi[d+1]) /*büyükten küçüğe sıralamak için '<' yerine '>' */
                        {
                                 degistir=dizi[d];
                                 dizi[d]=dizi[d+1];
                                 dizi[d+1]=degistir;
                        }
      }
}
printf("Küçükten büyüğe sıralı dizi:\n");
for(c=0;c<n;c++)printf("%d ",dizi[c]);
return 0;
}
```

```
#include <stdio.h>
int main()
{
  int dizi[100], n, c, d, position, t;
  printf("Dizi Kaç elemanlı olsun:");scanf("%d", &n);
  printf("%d sayı giriniz ve ENTER a basınız\n",n);
  for (c = 0; c < n; c++)     scanf("%d", &dizi[c]);
  for (c = 0; c < (n - 1); c++) // minimum elemanı n-1 kere bulma
  {
    position = c;
    for (d = c + 1; d < n; d++)
    {
      if (dizi[position] > dizi[d])
        position = d;
    }
    if (position != c)
      {
        t = dizi[c];
        dizi[c] = dizi[position];
        dizi[position] = t;
      }
  }
  printf("Küçükten büyüğe sıralı liste:\n");
  for (c = 0; c < n; c++)
    printf("%d", dizi[c]);

  return 0;
}
```