Dizileri Sıralama

Verileri sıralamak (yani, verileri artan veya azalan düzende yerleştirmek) en önemli bilgi işlem uygulamalarından biridir. Bir banka, tüm çekleri hesap numarasına göre tasnif eder, böylece her ayın sonunda ayrı banka hesap özetleri hazırlayabilir. Telefon şirketleri, telefon numaralarını bulmayı kolaylaştırmak için hesap listelerini soyadına ve bunun içinde ada göre sıralar. Neredeyse her kuruluş bazı verileri ve çoğu durumda büyük miktarlarda verileri sıralamalıdır. Verileri sıralamak, bilgisayar bilimi alanındaki en yoğun araştırma alanlarından biridir. Bu bölümde basit bir sıralama şemasını tartışacağız.

Şekil 6.15, 10 öğeli a dizisinin (satır 10) öğelerindeki değerleri artan düzende sıralar. Kullandığımız tekniğe baloncuk sıralama (bubble sort) veya batan sıralama denir çünkü daha küçük değerler kademeli olarak suyun içinde yükselen hava kabarcıkları gibi dizinin tepesine doğru "kabarcıklanır", daha büyük değerler ise dizinin altına batar. Teknik, diziden değişkenlere birkaç değer ataması yapmaktır. Her döngüde, ardışık eleman çiftleri (eleman 0 ve eleman 1, ardından eleman 1 ve eleman 2, vb.) karşılaştırılır. Bir çift artan sıradaysa (veya değerler aynıysa), değerleri oldukları gibi bırakırız. Bir çift azalan sıradaysa, değerleri dizide değiştirilir.

Program önce a[0] ile a[1]'i, ardından a[1] ile a[2]'yi, sonra a[2] ile a[3]'ü karşılaştırarak döngüyü tamamlayana kadar böyle devam eder ve son olarak a[8] ile a[9]'u karşılaştırır. 10 eleman olmasına rağmen sadece dokuz karşılaştırma yapılmıştır. Ardışık karşılaştırmaların yapılma şekli nedeniyle, büyük bir değer dizide tek bir döngüde birçok konum aşağı inebilir, ancak küçük bir değer yalnızca bir konum yukarı hareket edebilir.

İlk döngüde, en büyük değerin dizinin alt elemanı olan a[9]'a batması garanti edilir. İkinci döngüde, ikinci en büyük değerin a[8]'e düşmesi garanti edilir. Dokuzuncu döngüde, dokuzuncu en büyük değer a[1]'e düşer. Bu, a[0]'daki en küçük değeri bırakır, bu nedenle 10 öğe olmasına rağmen diziyi sıralamak için dizinin yalnızca 9 döngü gerekir.

Sıralama iç içe for döngüleri tarafından gerçekleştirilir (satır 21–34). Bir yer değiştirme gerekliyse, fazladan değişkenin değiştirilen iki değerden birini geçici olarak sakladığı üç atama tarafından gerçekleştirilir.

```
hold = a[i];
a[i] = a[i + 1];
a[i + 1] = hold;
```

Yer değiştirme yalnızca iki atama ile gerçekleştirilemez Örneğin, a[i] 7 ve a[i + 1] 5 ise, ilk atamadan sonra her iki değer de 5 olur ve 7 değeri kaybolur; dolayısıyla ekstra değişken tutma ihtiyacı doğar. Örneğin;

```
a[i] = a[i + 1];
a[i + 1] = a[i];
```

Balon sıralamanın (Bubble sort) başlıca özelliği, programlamanın kolay olmasıdır. Ancak, yavaş çalışır çünkü her değiş tokuş, bir öğeyi nihai hedefine yalnızca bir konum daha yaklaştırır. Bu, büyük dizileri sıralarken daha çok işlem demektir. Baloncuk sıralamanın daha hızlı versiyonlarını geliştirilebilir. Daha ileri düzey algoritmalar, sıralama ve aramayı çok daha etkin yapabilir. Fakat burada bu örnek sıralamanın mantığını vermek açısından önemlidir.

```c
// Fig. 6.15: fig06_15.c
// Sorting an array's values into ascending order.
#include <stdio.h>
#define SIZE 10

// function main begins program execution
int main(void)
{
   // initialize a
   int a[SIZE] = {2, 6, 4, 8, 10, 12, 89, 68, 45, 37};

   puts("Data items in original order");

   // output original array
   for (size_t i = 0; i < SIZE; ++i) {
      printf("%4d", a[i]);
   }


   // bubble sort
   // loop to control number of passes
   for (unsigned int pass = 1; pass < SIZE; ++pass) {

      // loop to control number of comparisons per pass
      for (size_t i = 0; i < SIZE - 1; ++i) {

         // compare adjacent elements and swap them if first
         // element is greater than second element
         if (a[i] > a[i + 1]) {
            int hold = a[i];
            a[i] = a[i + 1];
            a[i + 1] = hold;
         }
      }
   }

   puts("\nData items in ascending order");

   // output sorted array
   for (size_t i = 0; i < SIZE; ++i) {
      printf("%4d", a[i]);
   }

   puts("");
}
```

```
Data items in original order
   2   6   4   8  10  12  89  68  45  37
Data items in ascending order
   2   4   6   8  10  12  37  45  68  89
```

Şimdi bu programı yazıp çalıştıralım. Artan değil azalan sıra ile dizelim. Dizi boyutunu değiştirip deneyelim.

Ödev: Bu sıralama algoritmasında dizi büyüklüğünü ve dizi elemanlarını klavyeden girilecek ve azalan sırada sıralama yapacak şekilde yeniden düzenleyin. Ödevi en hızlı gönderen ilk 10 kişiye vize notuna fazladan 2 puan eklenecektir.

Örnek problem: Dizileri Kullanarak Ortalama, Medyan ve Mod Hesaplama

Şimdi daha büyük bir örnek ele alalım. Bilgisayarlar, anketlerin ve kamuoyu yoklamalarının sonuçlarını derlemek ve analiz etmek için anket verilerinin analizinde yaygın olarak kullanılır. Şekil 6.16, bir ankete verilen 99 yanıtla başlatılan dizi yanıtını kullanır. Her yanıt 1'den 9'a kadar bir sayıdır. Program 99 değerin ortalamasını, ortancasını (median) ve modunu hesaplar. Bu örnek, dizileri fonksiyonlara geçirmek de dahil olmak üzere genellikle dizi problemlerinde gerekli olan genel düzenlemelerin çoğunu içerecektir.

Ortalama; 99 değerin aritmetik ortalamasıdır. Fonksiyon ortalamayı (Şekil 6.16, satır 39–56), 99 elemanı toplayıp sonucu 99'a bölerek ortalamayı hesaplar. Bu örnekte Ortalama, veri öğelerinin ortalama değeridir. Ortalama, veri öğelerinin sayısına (99) bölünen tüm veri öğelerinin toplamına eşittir. Bu çalışma için ortalama değer: 681 / 99 = 6,8788

Medyan; ortadaki değerdir. medyan fonksiyonu (59-77. satırlar), answer dizisini artan düzende sıralamak için bubbleSort fonksiyonunu (127-143. satırlarda tanımlanmıştır) çağırarak ve ardından sıralanan dizinin answer[SIZE / 2] (ortadaki öğe) öğesini seçerek medyanı belirler. Eleman sayısı çift olduğunda, medyan ortadaki iki elemanın ortalaması olarak hesaplanmalıdır. Dizinin çıktısını almak için printArray fonksiyonu (satır 146–157) çağrılır. Bu örnekte, Medyan, sıralanmış 99 öğe dizisinin 49. öğesidir. Bu dizide medyan 7'dir.

Mod, 99 yanıt arasında en sık görülen değerdir. Fonksiyon (satır 80–124), her türün yanıt sayısını sayarak ve ardından en büyük sayıya sahip değeri seçerek modu belirler. Mod, en sık kullanılan değerdir. Bu çalışma için mod, 27 kez meydana gelen 8'dir.

Bu programa göz atalım. İlk bölümde program kütüphaneleri ve değişkenlerin tanımlamaları verilmiş;

```
1  // Fig. 6.16: fig06_16.c
2  // Survey data analysis with arrays:
3  // computing the mean, median and mode of the data.
4  #include <stdio.h>
5  #define SIZE 99
```

Bu bölümde fonksiyon prototipleri verilmiş; Burada satır 11 de hata var. Sadece “int a[]” yazılmış. Ama (void bubbleSort(unsigned int answer[]);) şeklinde olmalı.

```
6
7  // function prototypes
8  void mean(const unsigned int answer[]);
9  void median(unsigned int answer[]);
10 void mode(unsigned int freq[], unsigned const int answer[]) ;
11 void bubbleSort(int a[]);
12 void printArray(unsigned const int a[]);
```

Bu Bölümde ana program verilmiş, 10 elemanlı “frequency” dizisi ilk değerler 0 olarak başlatılmış. “response” dizisinde ise 99 değer girilmiştir. (SIZE=99 olduğuna dikkat edelim.) Daha sonra “mean”, “median” ve “mode” fonksiyonları çağrılarak ana program sonlandırılmıştır.

```
13
14 // function main begins program execution
15 int main(void)
16 {
17    unsigned int frequency[10] = {0}; // initialize array frequency
18
19    // initialize array response
20    unsigned int response[SIZE] =
21       {6, 7, 8, 9, 8, 7, 8, 9, 8, 9,
22        7, 8, 9, 5, 9, 8, 7, 8, 7, 8,
23        6, 7, 8, 9, 3, 9, 8, 7, 8, 7,
24        7, 8, 9, 8, 9, 8, 9, 7, 8, 9,
25        6, 7, 8, 7, 8, 7, 9, 8, 9, 2,
26        7, 8, 9, 8, 9, 8, 9, 7, 5, 3,
27        5, 6, 7, 2, 5, 3, 9, 4, 6, 4,
28        7, 8, 9, 6, 8, 7, 8, 9, 7, 8,
29        7, 4, 4, 2, 5, 3, 8, 7, 5, 6,
30        4, 5, 6, 1, 6, 5, 7, 8, 7};
31
32    // process responses
33    mean(response);
34    median(response);
35    mode(frequency, response);
36 }
```

Bu bölümde “mean” fonksiyonu çalıştırılmaktadır. Burada “mean(response);” fonksiyonu çağrılarak “answer[j]” ile “response” dizisindeki 99 değer “total” değişkeninde toplanmaktadır. Sonuçta da toplam ve toplam/SIZE (yani total/99) yazdırılmaktadır.

```
37
38  // calculate average of all response values
39  void mean(const unsigned int answer[])
40  {
41     printf("%s\n%s\n%s\n", "********", "  Mean", "********");
42
43     unsigned int total = 0; // variable to hold sum of array elements
44
45     // total response values
46     for (size_t j = 0; j < SIZE; ++j) {
47        total += answer[j];
48     }
49
50     printf("The mean is the average value of the data\n"
51            "items. The mean is equal to the total of\n"
52            "all the data items divided by the number\n"
53            "of data items (%u). The mean value for\n"
54            "this run is: %u / %u = %.4f\n\n",
55            SIZE, total, SIZE, (double) total / SIZE);
56  }
```

Bu bölümde ise medyan hesaplanmaktadır. “median(response);” ile median fonksiyonu çağrılınca program bu bölüme atlar. Burada printArray(answer) ve bubleSort(answer) fonksiyonları çağrılır. Bu fonksiyonlarda diziyi sıralayıp ortadaki elemanı yazdırır. Burada 73 ncü satırda; %u yazıldığı için hesaplanan değer işaretsiz tamsayı (unsigned int) olacaktır. Ayrıca 76 ncı satırda SIZE/2'nin değeri 49.5 olması gerekirken tamsayı değer aldığımızdan bu değer 49 olarak okunur.

```
57
58  // sort array and determine median element's value
59  void median(unsigned int answer[])
60  {
61     printf("\n%s\n%s\n%s\n%s",
62            "********", " Median", "********",
63            "The unsorted array of responses is");
64
65     printArray(answer); // output unsorted array
66
67     bubbleSort(answer); // sort array
68
69     printf("%s", "\n\nThe sorted array is");
70     printArray(answer); // output sorted array
71
72     // display median element
73     printf("\n\nThe median is element %u of\n"
74            "the sorted %u element array.\n"
75            "For this run the median is %u\n\n",
76            SIZE / 2, SIZE, answer[SIZE / 2]);
77  }
```

Bu bölümde mod hesaplanmaktadır. Mod işlemi ana programdan mode(frequency, response); fonksiyonu ile çağrıldığında buraya atlar. Bu fonksiyonda "freq[]" dizisinde benzer cevapların sayısı tutulmaktadır.
85-87 nci satıtlarda; for döngüsü ile freq[1]=0 ; freq[2]=0 ;freq[3]=0 ; freq[4]=0 ; freq[5]=0 ; freq[6]=0 ; freq[7]=0 ; freq[8]=0 ; freq[9]=0 atamaları yapılır.
90-92 nci satırlarda; freq[answer[0]]=freq[6] ; freq[answer[1]]=freq[7] ; freq[answer[2]]=freq[8] ; freq[answer[3]]=freq[9] ; freq[answer[4]]=freq[8], ... freq[answer[99]]=freq[7] atamaları yapılır.
Burada freq[1], yada freq[2],.....freq[9} tekrar eden 1, yada 2, ...yada 9 un sayısını tutmaktadır.
Örneğin 8 değeri 27 defa tekrar etmişse freq[8] 27 defa artırılacak ve değeri 27 olacaktır. burada aslında anahtar satır;
++freq[answer[j]];
her defasında sadece ilgili değeri 1 artırıyor.

Ayrıca 103 ncü satırda bir hata var (for (size_t rating = 1; rating <= 9; ++rating) {) olarak düzeltilmeli. Çünki "rating" değişkeninin veri tipi belirtilmemiş. Burada "size_t" dizilerde veri tipini belirtmek için kullanılmaktadır.

```
78
79  // determine most frequent response
80  void mode(unsigned int freq[], const unsigned int answer[])
81  {
82     printf("\n%s\n%s\n%s\n","********", "  Mode", "********");
83
84     // initialize frequencies to 0
85     for (size_t rating = 1; rating <= 9; ++rating) {
86        freq[rating] = 0;
87     }
88
89     // summarize frequencies
90     for (size_t j = 0; j < SIZE; ++j) {
91        ++freq[answer[j]];
92     }
93
94     // output headers for result columns
95     printf("%s%11s%19s\n\n%54s\n%54s\n\n",
96            "Response", "Frequency", "Histogram",
97            "1    1    2    2", "5    0    5    0    5");
98
99     // output results
100    unsigned int largest = 0; // represents largest frequency
101    unsigned int modeValue = 0; // represents most frequent response
102
103    for (rating = 1; rating <= 9; ++rating) {
104       printf("%8u%11u          ", rating, freq[rating]);
105
```

107-110 ncu satırlarda; Burada karşılaştırma yaparak en büyük freq[] değerini buluyor.

```
106         // keep track of mode value and largest frequency value
107         if (freq[rating] > largest) {
108            largest = freq[rating];
109            modeValue = rating;
110         }
111
112         // output histogram bar representing frequency value
113         for (unsigned int h = 1; h <= freq[rating]; ++h) {
114            printf("%s", "*");
115         }
116
117         puts(""); // being new line of output
118      }
119
120      // display the mode value
121      printf("\nThe mode is the most frequent value.\n"
122             "For this run the mode is %u which occurred"
123             " %u times.\n", modeValue, largest);
124   }
```

Bu bölümde “bubleSort” fonksiyonu ile sıralama yapılmaktadır. 133 ncü satırda Burada 99 değerin sırayla karşılaştırılması için 98 karşılaştırma yapmak yeterli olacaktır. Bu nedenle SIZE-1 e kadar döngü oluşturulması gerekir.

```
125
126   // function that sorts an array with bubble sort algorithm
127   void bubbleSort(unsigned int a[])
128   {
129      // loop to control number of passes
130      for (unsigned int pass = 1; pass < SIZE; ++pass) {
131
132         // loop to control number of comparisons per pass
133         for (size_t j = 0; j < SIZE - 1; ++j) {
134
135            // swap elements if out of order
136            if (a[j] > a[j + 1]) {
137               unsigned int hold = a[j];
138               a[j] = a[j + 1];
139               a[j + 1] = hold;
140            }
141         }
142      }
143   }
```

Bu bölümde ise diziyi yazdıran fonksiyon çalışmaktadır. 151 nci satırda Her satırda 20 değer olacak şekilde ekrana bastırır.

```
144
145 // output array contents (20 values per row)
146 void printArray(const unsigned int a[])
147 {
148    // output array contents
149    for (size_t j = 0; j < SIZE; ++j) {
150
151       if (j % 20 == 0) { // begin new line every 20 values
152          puts("");
153       }
154
155       printf("%2u", a[j]);
156    }
157 }
```

Ödev: Responses dizisinin elemanlarını klavyeden girilecek şekilde düzenleyip gönderirseniz vize notunuzda fazladan 2 puan eklenecek (Gönderen ilk 10 kişiye)