BİL-142 Bilgisayar Programlama II (C/C++)

Hazırlayan: M.Ali Akcayol
Gazi Üniversitesi
Bilgisayar Mühendisliği Bölümü

Konular

- Giriş
- Diziler
- Dizi Kullanımı
- Karakter Dizileri
- Fonksiyonlarda Dizi Kullanımı
- Dizilerde Arama
- Dizilerde Sıralama
- Çok Boyutlu Diziler

Giriş

- Diziler **aynı türden** verilerden oluşan veri yapılarıdır.
- Diziler, class ve structure gibi program çalışma süresinde sabit boyutta elemana sahiptir.
- Kuyruk (queue), yığın (stack) ve ağaç (tree) yapıları dinamiktir ve program çalışması sırasında boyutları değişebilir.

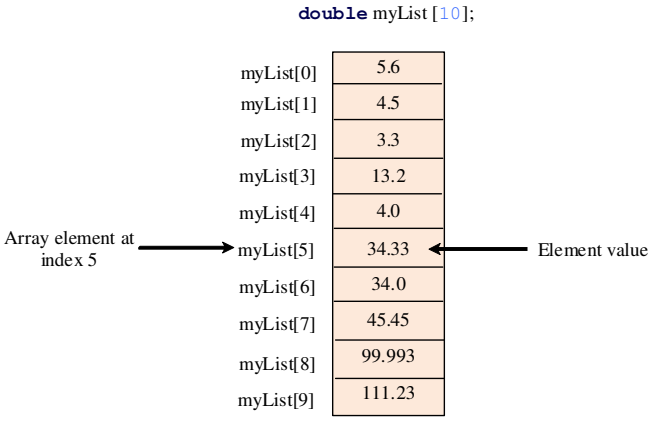
3

Konular

- Giriş
- Diziler
- Dizi Kullanımı
- Karakter Dizileri
- Fonksiyonlarda Dizi Kullanımı
- Dizilerde Arama
- Dizilerde Sıralama
- Çok Boyutlu Diziler

Diziler

- Dizilerde birden çok aynı türde eleman bulunur.
- Dizideki bir eleman **dizi adı** ve **dizideki sırası** ile gösterilir.
- Dizilerin isimlendirmesinde değişken isimlendirme kuralları geçerlidir.



5

Diziler

- Elemanın sırası integer bir sayı ile veya integer ifadeyle gösterilir.
- Dizi elemanları üzerinde değişkenler üzerinde yapılan işlemler yapılabilir.

```
cout << c[ 0 ] + c[ 1 ] + c[ 2 ] << endl;

x = c[ 6 ] / 2;
```

- **Dizilerde ilk elemanın sırası 0 ile başlar son elemanın sırası (eleman sayısı -1) ile gösterilir.**

6

Konular

- Giriş
- Diziler
- Dizi Kullanımı
- Karakter Dizileri
- Fonksiyonlarda Dizi Kullanımı
- Dizilerde Arama
- Dizilerde Sıralama
- Çok Boyutlu Diziler

Dizi Kullanımı

- Dizi elemanlarını gösteren [ ] parantezi, ( ) parantezinden sonra ikinci önceliğe sahiptir ve diğer tüm operatörlerden daha önceliklidir.
- Diziler aşağıdaki gibi tanımlanabilir.

```
int c[ 12 ]; // c is an array of 12 integers

int b[ 100 ], // b is an array of 100 integers
    x[ 27 ]; // x is an array of 27 integers
```

- Dizilere tanımlandığı yerde başlangıç değeri atanabilir.

```
int n[] = { 1, 2, 3, 4, 5 };

int n[ 5 ] = { 32, 27, 64, 18, 95, 14 };
```

- Başlangıç değeri adedi belirtilen eleman adedinden fazla olamaz.
- Dizi elemanlarına klavyeden değer girilebilir.

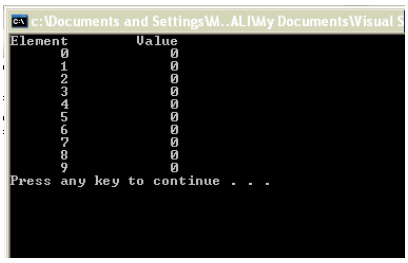
```
for ( int j = 0; j < arraySize; j++ )
   cin >> a[ j ];
```

8

Dizi Kullanımı

- Dizilere başlangıç değeri atama.

```
// Fig. 7.3: fig07_03.cpp
// Initializing an array.
#include <iostream>
using std::cout;
using std::endl;

#include <iomanip>
using std::setw;

int main()
{
   int n[ 10 ]; // n is an array of 10 integers

   // initialize elements of array n to 0
   for ( int i = 0; i < 10; i++ )
      n[ i ] = 0; // set element at location i to 0

   cout << "Element" << setw( 13 ) << "Value" << endl;

   // output each array element's value
   for ( int j = 0; j < 10; j++ )
      cout << setw( 7 ) << j << setw( 13 ) << n[ j ] << endl;

   system("PAUSE");
   return 0; // indicates successful termination
} // end main
```

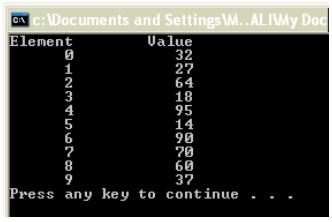


9

Dizi Kullanımı

- Dizilere başlangıç değeri atama.

```
// Fig. 7.4: fig07_04.cpp
// Initializing an array in a declaration.
#include <iostream>
using std::cout;
using std::endl;

#include <iomanip>
using std::setw;

int main()
{
   // use initializer list to initialize array n
   int n[ 10 ] = { 32, 27, 64, 18, 95, 14, 90, 70, 60, 37 };

   cout << "Element" << setw( 13 ) << "Value" << endl;

   // output each array element's value
   for ( int i = 0; i < 10; i++ )
      cout << setw( 7 ) << i << setw( 13 ) << n[ i ] << endl;

   system("PAUSE");
   return 0; // indicates successful termination
} // end main
```



10

Dizi Kullanımı

- Dizi boyutu bir sabitle tanımlanabilir.

```
1  // Fig. 7.5: fig07_05.cpp
2  // Set array s to the even integers from 2 to 20.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  #include <iomanip>
8  using std::setw;
9
10 int main()
11 {
12    // constant variable can be used to specify array size
13    const int arraySize = 10; // must initialize in declaration
14
15    int s[ arraySize ]; // array s has 10 elements
16
17    for ( int i = 0; i < arraySize; i++ ) // set the values
18       s[ i ] = 2 + 2 * i;
19
20    cout << "Element" << setw( 13 ) << "Value" << endl;
21
22    // output contents of array s in tabular format
23    for ( int j = 0; j < arraySize; j++ )
24       cout << setw( 7 ) << j << setw( 13 ) << s[ j ] << endl;
25
26    system("PAUSE");
27    return 0; // indicates successful termination
28 } // end main
29
```

```
c:\Documents and Settings\M..ALI\My Docur
Element        Value
      0            2
      1            4
      2            6
      3            8
      4           10
      5           12
      6           14
      7           16
      8           18
      9           20
Press any key to continue . . .
```

11

Dizi Kullanımı

- Dizi elemanlarının toplamı.

```
1  // Fig. 7.8: fig07_08.cpp
2  // Compute the sum of the elements of the array.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  int main()
8  {
9     const int arraySize = 10; // constant variable indicating size of array
10    int a[ arraySize ] = { 87, 68, 94, 100, 83, 78, 85, 91, 76, 87 };
11    int total = 0;
12
13    // sum contents of array a
14    for ( int i = 0; i < arraySize; i++ )
15       total += a[ i ];
16
17    cout << "Total of array elements: " << total << endl;
18
19    system("PAUSE");
20    return 0; // indicates successful termination
21 } // end main
22
```

```
c:\Documents and Settings\M..ALI\My Docu
Total of array elements: 849
Press any key to continue . . .
```

12

Dizi Kullanımı

- Dizi elemanları üzerinde işlemler.

```
1  // Fig. 7.9: fig07_09.cpp
2  // Bar chart printing program.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  #include <iomanip>
8  using std::setw;
9
10 int main()
11 {
12    const int arraySize = 11;
13    int n[ arraySize ] = { 0, 0, 0, 0, 0, 0, 1, 2, 4, 2, 1 };
14
15    cout << "Grade distribution:" << endl;
16
```


Dizi Kullanımı

- Dizi elemanları üzerinde işlemler. (devam)

```
17    // for each element of array n, output a bar of the chart
18    for ( int i = 0; i < arraySize; i++ )
19    {
20       // output bar labels ("0-9:", ..., "90-99:", "100:" )
21       if ( i == 0 )
22          cout << "  0-9: ";
23       else if ( i == 10 )
24          cout << "  100: ";
25       else
26          cout << i * 10 << "-" << ( i * 10 ) + 9 << ": ";
27
28       // print bar of asterisks
29       for ( int stars = 0; stars < n[ i ]; stars++ )
30          cout << '*';
31
32       cout << endl; // start a new line of output
33    } // end outer for
34
35    system("PAUSE");
36    return 0; // indicates successful termination
37 } // end main
38
```

```
c:\Documents and Settings\M...ALI\My Doc
Grade distribution:
  0-9:
10-19:
20-29:
30-39:
40-49:
50-59:
60-69: *
70-79: **
80-89: ****
90-99: **
  100: *
Press any key to continue . . .
```

Dizi Kullanımı

- Dizi elemanlarının sayaç olarak kullanımı.

```
1  // Fig. 7.10: fig07_10.cpp
2  // Roll a six-sided die 6,000,000 times.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  #include <iomanip>
8  using std::setw;
9
10 #include <cstdlib>
11 using std::rand;
12 using std::srand;
13
14 #include <ctime>
15 using std::time;
16
```

15

Dizi Kullanımı

- Dizi elemanlarının sayaç olarak kullanımı. (devam)

```
17 int main()
18 {
19    const int arraySize = 7; // ignore element zero
20    int frequency[ arraySize ] = { 0 };
21
22    srand( time( 0 ) ); // seed random number generator
23
24    // roll die 6,000,000 times; use die value as frequency index
25    for ( int roll = 1; roll <= 6000000; roll++ )
26       frequency[ 1 + rand() % 6 ]++;
27
28    cout << "Face" << setw( 13 ) << "Frequency" << endl;
29
30    // output each array element's value
31    for ( int face = 1; face < arraySize; face++ )
32       cout << setw( 4 ) << face << setw( 13 ) << frequency[ face ]
33          << endl;
34
35    system("PAUSE");
36    return 0; // indicates successful termination
37 } // end main
38
```

```
c:\Documents and Settings\M...ALI\My Do
Face      Frequency
   1         999952
   2         998779
   3        1000058
   4        1000699
   5         999907
   6        1000605
Press any key to continue . . .
```

16

Dizi Kullanımı

- Dizi ile anket uygulaması.

```
// Fig. 7.11: fig07_11.cpp
// Student poll program.
#include <iostream>
using std::cout;
using std::endl;

#include <iomanip>
using std::setw;

int main()
{
   // define array sizes
   const int responseSize = 40; // size of array responses
   const int frequencySize = 11; // size of array frequency

   // place survey responses in array responses
   const int responses[ responseSize ] = { 1, 2, 6, 4, 8, 5, 9, 7, 8,
      10, 1, 6, 3, 8, 6, 10, 3, 8, 2, 7, 6, 5, 7, 6, 8, 6, 7,
      5, 6, 6, 5, 6, 7, 5, 6, 4, 8, 6, 8, 10 };

   // initialize frequency counters to 0
   int frequency[ frequencySize ] = { 0 };
```

17

Dizi Kullanımı

- Dizi ile anket uygulaması. (devam)

```
   // for each answer, select responses element and use that value
   // as frequency subscript to determine element to increment
   for ( int answer = 0; answer < responseSize; answer++ )
      frequency[ responses[ answer ] ]++;

   cout << "Rating" << setw( 17 ) << "Frequency" << endl;

   // output each array element's value
   for ( int rating = 1; rating < frequencySize; rating++ )
      cout << setw( 6 ) << rating << setw( 17 ) << frequency[ rating ]
         << endl;

   system("PAUSE");
   return 0; // indicates successful termination
} // end main
```

```
c:\Documents and Settings\M..ALI\My Docur
Rating          Frequency
     1                  2
     2                  2
     3                  2
     4                  2
     5                  5
     6                 11
     7                  5
     8                  7
     9                  1
    10                  3
Press any key to continue . . .
```

18

Konular

- Giriş
- Diziler
- Dizi Kullanımı
- Karakter Dizileri
- Fonksiyonlarda Dizi Kullanımı
- Dizilerde Arama
- Dizilerde Sıralama
- Çok Boyutlu Diziler

Karakter Dizileri

- Bir string karakter dizileri şeklinde tanımlanabilir.

```
char string1[] = "first";

char string1[] = { 'f', 'i', 'r', 's', 't', '\0' };
```

- Karakter dizileri şeklinde tanımlanan string değişkene klavyeden değer girilebilir.

```
char string2[ 20 ];

cin >> string2;

cout << string2;
```

20

Karakter Dizileri

- Karakter dizileri – örnek

```
1  // Fig. 7.12: fig07_12.cpp
2  // Treating character arrays as strings.
3  #include <iostream>
4  using std::cout;
5  using std::cin;
6  using std::endl;
7
8  int main()
9  {
10    char string1[ 20 ]; // reserves 20 characters
11    char string2[] = "string literal"; // reserves 15 characters
12
13    // read string from user into array string1
14    cout << "Enter the string \"hello there\": ";
15    cin >> string1; // reads "hello" [space terminates input]
16
17    // output strings
18    cout << "string1 is: " << string1 << "\nstring2 is: " << string2;
19
20    cout << "\nstring1 with spaces between characters is:\n";
21
22    // output characters until null character is reached
23    for ( int i = 0; string1[ i ] != '\0'; i++ )
24       cout << string1[ i ] << ' ';
25
26    cin >> string1; // reads "there"
27    cout << "\nstring1 is: " << string1 << endl;
28
29    system("PAUSE");
30    return 0; // indicates successful termination
31 } // end main
32
```

```
c:\Documents and Settings\M...ALI\My Documents\Visua
Enter the string "hello there": merhaba
string1 is: merhaba
string2 is: string literal
string1 with spaces between characters is:
m e r h a b a
```


Karakter Dizileri

- Static ve automatic local arrays

```
1  // Fig. 7.13: fig07_13.cpp
2  // Static arrays are initialized to zero.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  void staticArrayInit( void ); // function prototype
8  void automaticArrayInit( void ); // function prototype
9
10 int main()
11 {
12    cout << "First call to each function:\n";
13    staticArrayInit();
14    automaticArrayInit();
15
16    cout << "\n\nSecond call to each function:\n";
17    staticArrayInit();
18    automaticArrayInit();
19    cout << endl;
20
21    system("PAUSE");
22    return 0; // indicates successful termination
23 } // end main
24
```

Karakter Dizileri

- Static ve automatic local arrays - devam

```
25 // function to demonstrate a static local array
26 void staticArrayInit( void )
27 {
28    // initializes elements to 0 first time function is called
29    static int array1[ 3 ]; // static local array
30
31    cout << "\nValues on entering staticArrayInit:\n";
32
33    // output contents of array1
34    for ( int i = 0; i < 3; i++ )
35       cout << "array1[" << i << "] = " << array1[ i ] << "  ";
36
37    cout << "\nValues on exiting staticArrayInit:\n";
38
39    // modify and output contents of array1
40    for ( int j = 0; j < 3; j++ )
41       cout << "array1[" << j << "] = " << ( array1[ j ] += 5 ) << "  ";
42 } // end function staticArrayInit
43
```

23

Karakter Dizileri

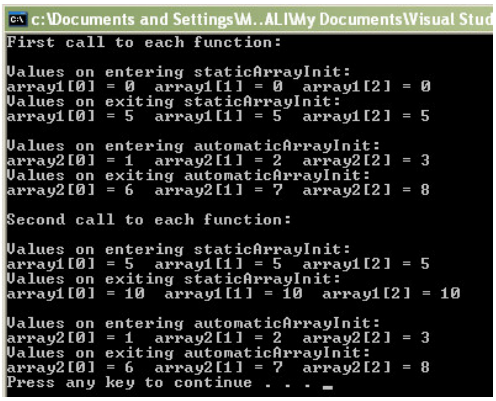
- Static ve automatic local arrays - devam

```
44 // function to demonstrate an automatic local array
45 void automaticArrayInit( void )
46 {
47    // initializes elements each time function is called
48    int array2[ 3 ] = { 1, 2, 3 }; // automatic local array
49
50    cout << "\n\nValues on entering automaticArrayInit:\n";
51
52    // output contents of array2
53    for ( int i = 0; i < 3; i++ )
54       cout << "array2[" << i << "] = " << array2[ i ] << "  ";
55
56    cout << "\nValues on exiting automaticArrayInit:\n";
57
58    // modify and output contents of array2
59    for ( int j = 0; j < 3; j++ )
60       cout << "array2[" << j << "] = " << ( array2[ j ] += 5 ) << "  ";
61 } // end function automaticArrayInit
62
```

24

Karakter Dizileri

- Static ve automatic local arrays - devam



25

Konular

- Giriş
- Diziler
- Dizi Kullanımı
- Karakter Dizileri
- Fonksiyonlarda Dizi Kullanımı
- Dizilerde Arama
- Dizilerde Sıralama
- Çok Boyutlu Diziler

Fonksiyonlarda Dizi Kullanımı

- Fonksiyonlara dizi gönderirken sadece isminin yazılması yeterlidir.

```
int hourlyTemperatures [24];    // dizi tanımı


modifyArray(hourlyTemperatures);    // fonksiyonu dizi göndererek çağırma


void modifyArray(int []);       // fonksiyonun prototip tanımı

void modifyArray(int a[])       // fonksiyonun tanımı
{
    ...
}
```

- Bir dizi fonksiyona gönderilirken eleman sayısıda gider.
- Dizilerde değişkenlerde olduğu gibi call-by-reference şeklinde gönderilebilir.

27

Fonksiyonlarda Dizi Kullanımı

- Örnek

```
1  // Fig. 7.14: fig07_14.cpp
2  // Passing arrays and individual array elements to functions.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  #include <iomanip>
8  using std::setw;
9
10 void modifyArray( int [], int ); // appears strange
11 void modifyElement( int );
12
```

28

Fonksiyonlarda Dizi Kullanımı

- Örnek – devam

```
13 int main()
14 {
15    const int arraySize = 5; // size of array a
16    int a[ arraySize ] = { 0, 1, 2, 3, 4 }; // initialize array a
17
18    cout << "Effects of passing entire array by reference:"
19       << "\n\nThe values of the original array are:\n";
20
21    // output original array elements
22    for ( int i = 0; i < arraySize; i++ )
23       cout << setw( 3 ) << a[ i ];
24
25    cout << endl;
26
27    // pass array a to modifyArray by reference
28    modifyArray( a, arraySize );
29    cout << "The values of the modified array are:\n";
30
31    // output modified array elements
32    for ( int j = 0; j < arraySize; j++ )
33       cout << setw( 3 ) << a[ j ];
34
35    cout << "\n\n\nEffects of passing array element by value:"
36       << "\n\na[3] before modifyElement: " << a[ 3 ] << endl;
37
38    modifyElement( a[ 3 ] ); // pass array element a[ 3 ] by value
39    cout << "a[3] after modifyElement: " << a[ 3 ] << endl;
40
41    system("PAUSE");
42    return 0; // indicates successful termination
43 } // end main
```

29

Fonksiyonlarda Dizi Kullanımı

- Örnek – devam

```
45 // in function modifyArray, "b" points to the original array "a" in memory
46 void modifyArray( int b[], int sizeOfArray )
47 {
48    // multiply each array element by 2
49    for ( int k = 0; k < sizeOfArray; k++ )
50       b[ k ] *= 2;
51 } // end function modifyArray
52
53 // in function modifyElement, "e" is a local copy of
54 // array element a[ 3 ] passed from main
55 void modifyElement( int e )
56 {
57    // multiply parameter by 2
58    cout << "Value of element in modifyElement: " << ( e *= 2 ) << endl;
59 } // end function modifyElement
60
61
```

```
c:\Documents and Settings\M..ALI\My Documents\Visual Stu
Effects of passing entire array by reference:

The values of the original array are:
  0  1  2  3  4
The values of the modified array are:
  0  2  4  6  8


Effects of passing array element by value:

a[3] before modifyElement: 6
Value of element in modifyElement: 12
a[3] after modifyElement: 6
Press any key to continue . . .
```

30

Fonksiyonlarda Dizi Kullanımı

- `const` olarak tanımlanmış dizilerde değişklik yapılamaz.

```
1  // Fig. 7.15: fig07_15.cpp
2  // Demonstrating the const type qualifier.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  void tryToModifyArray( const int [] ); // function prototype
8
9  int main()
10 {
11    int a[] = { 10, 20, 30 };
12
13    tryToModifyArray( a );
14    cout << a[ 0 ] << ' ' << a[ 1 ] << ' ' << a[ 2 ] << '\n';
15
16    system("PAUSE");
17    return 0; // indicates successful termination
18 } // end main
19
20 // In function tryToModifyArray, "b" cannot be used
21 // to modify the original array "a" in main.
22 void tryToModifyArray( const int b[] )
23 {
24    b[ 0 ] /= 2; // error
25    b[ 1 ] /= 2; // error
26    b[ 2 ] /= 2; // error
27 } // end function tryToModifyArray
28
```

```
rrr.cpp(24) : error C3892: 'b' : you cannot assign to a variable that is const
```

31

Fonksiyonlarda Dizi Kullanımı

- GradeBook örnek program.

```
1  // Fig. 7.16: GradeBook.h
2  // Definition of class GradeBook that uses an array to store test grades.
3  // Member functions are defined in GradeBook.cpp
4
5  #include <string> // program uses C++ Standard Library string class
6  using std::string;
7
8  // GradeBook class definition
9  class GradeBook
10 {
11 public:
12    // constant -- number of students who took the test
13    const static int students = 10; // note public data
14
15    // constructor initializes course name and array of grades
16    GradeBook( string, const int [] );
17
18    void setCourseName( string ); // function to set the course name
19    string getCourseName(); // function to retrieve the course name
20    void displayMessage(); // display a welcome message
21    void processGrades(); // perform various operations on the grade data
22    int getMinimum(); // find the minimum grade for the test
23    int getMaximum(); // find the maximum grade for the test
24    double getAverage(); // determine the average grade for the test
25    void outputBarChart(); // output bar chart of grade distribution
26    void outputGrades(); // output the contents of the grades array
27 private:
28    string courseName; // course name for this grade book
29    int grades[ students ]; // array of student grades
30 }; // end class GradeBook
31
```

32

Fonksiyonlarda Dizi Kullanımı

- GradeBook örnek program - devam.

```
1  // Fig. 7.17: GradeBook.cpp
2  // Member-function definitions for class GradeBook that
3  // uses an array to store test grades.
4  #include <iostream>
5  using std::cout;
6  using std::cin;
7  using std::endl;
8  using std::fixed;
9
10 #include <iomanip>
11 using std::setprecision;
12 using std::setw;
13
14 #include "GradeBook.h" // GradeBook class definition
15
16 // constructor initializes courseName and grades array
17 GradeBook::GradeBook( string name, const int gradesArray[] )
18 {
19    setCourseName( name ); // initialize courseName
20
21    // copy grades from gradeArray to grades data member
22    for ( int grade = 0; grade < students; grade++ )
23       grades[ grade ] = gradesArray[ grade ];
24 } // end GradeBook constructor
25
```


Fonksiyonlarda Dizi Kullanımı

- GradeBook örnek program - devam.

```
26 // function to set the course name
27 void GradeBook::setCourseName( string name )
28 {
29    courseName = name; // store the course name
30 } // end function setCourseName
31
32 // function to retrieve the course name
33 string GradeBook::getCourseName()
34 {
35    return courseName;
36 } // end function getCourseName
37
38 // display a welcome message to the GradeBook user
39 void GradeBook::displayMessage()
40 {
41    // this statement calls getCourseName to get the
42    // name of the course this GradeBook represents
43    cout << "Welcome to the grade book for\n" << getCourseName() << "!"
44       << endl;
45 } // end function displayMessage
46
```

Fonksiyonlarda Dizi Kullanımı

- GradeBook örnek program - devam.

```
47  // perform various operations on the data
48  void GradeBook::processGrades()
49  {
50     // output grades array
51     outputGrades();
52
53     // call function getAverage to calculate the average grade
54     cout << "\nClass average is " << setprecision( 2 ) << fixed <<
55        getAverage() << endl;
56
57     // call functions getMinimum and getMaximum
58     cout << "Lowest grade is " << getMinimum() << "\nHighest grade is "
59        << getMaximum() << endl;
60
61     // call function outputBarChart to print grade distribution chart
62     outputBarChart();
63  } // end function processGrades
64
```

35

Fonksiyonlarda Dizi Kullanımı

- GradeBook örnek program - devam.

```
65  // find minimum grade
66  int GradeBook::getMinimum()
67  {
68     int lowGrade = 100; // assume lowest grade is 100
69
70     // loop through grades array
71     for ( int grade = 0; grade < students; grade++ )
72     {
73        // if current grade lower than lowGrade, assign it to lowGrade
74        if ( grades[ grade ] < lowGrade )
75           lowGrade = grades[ grade ]; // new lowest grade
76     } // end for
77
78     return lowGrade; // return lowest grade
79  } // end function getMinimum
80
```

36

Fonksiyonlarda Dizi Kullanımı

- GradeBook örnek program - devam.

```
81  // find maximum grade
82  int GradeBook::getMaximum()
83  {
84     int highGrade = 0; // assume highest grade is 0
85
86     // loop through grades array
87     for ( int grade = 0; grade < students; grade++ )
88     {
89        // if current grade higher than highGrade, assign it to highGrade
90        if ( grades[ grade ] > highGrade )
91           highGrade = grades[ grade ]; // new highest grade
92     } // end for
93
94     return highGrade; // return highest grade
95  } // end function getMaximum
96
97  // determine average grade for test
98  double GradeBook::getAverage()
99  {
100    int total = 0; // initialize total
101
102    // sum grades in array
103    for ( int grade = 0; grade < students; grade++ )
104       total += grades[ grade ];
105
106    // return average of grades
107    return static_cast< double >( total ) / students;
108 } // end function getAverage
109
```

37

Fonksiyonlarda Dizi Kullanımı

- GradeBook örnek program - devam.

```
110 // output bar chart displaying grade distribution
111 void GradeBook::outputBarChart()
112 {
113    cout << "\nGrade distribution:" << endl;
114
115    // stores frequency of grades in each range of 10 grades
116    const int frequencySize = 11;
117    int frequency[ frequencySize ] = { 0 };
118
119    // for each grade, increment the appropriate frequency
120    for ( int grade = 0; grade < students; grade++ )
121       frequency[ grades[ grade ] / 10 ]++;
122
123    // for each grade frequency, print bar in chart
124    for ( int count = 0; count < frequencySize; count++ )
125    {
126       // output bar labels ("0-9:", ..., "90-99:", "100:" )
127       if ( count == 0 )
128          cout << "  0-9: ";
129       else if ( count == 10 )
130          cout << "  100: ";
131       else
132          cout << count * 10 << "-" << ( count * 10 ) + 9 << ": ";
133
134       // print bar of asterisks
135       for ( int stars = 0; stars < frequency[ count ]; stars++ )
136          cout << '*';
137
138       cout << endl; // start a new line of output
139    } // end outer for
140 } // end function outputBarChart
141
```

38

Fonksiyonlarda Dizi Kullanımı

- GradeBook örnek program - devam.

```
142  // output the contents of the grades array
143 void GradeBook::outputGrades()
144  {
145     cout << "\nThe grades are:\n\n";
146
147     // output each student's grade
148     for ( int student = 0; student < students; student++ )
149        cout << "Student " << setw( 2 ) << student + 1 << ": " << setw( 3 )
150           << grades[ student ] << endl;
151  } // end function outputGrades
152
153
```

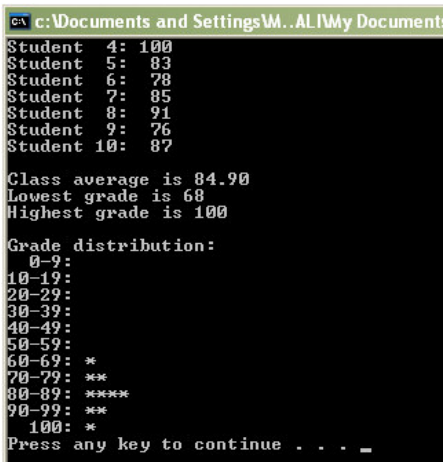

Fonksiyonlarda Dizi Kullanımı

- GradeBook örnek program - devam.

```
1  // Fig. 7.18: fig07_18.cpp
2  // Creates GradeBook object using an array of grades.
3
4  #include "GradeBook.h" // GradeBook class definition
5
6  // function main begins program execution
7  int main()
8  {
9     // array of student grades
10    int gradesArray[ GradeBook::students ] =
11       { 87, 68, 94, 100, 83, 78, 85, 91, 76, 87 };
12
13    GradeBook myGradeBook(
14       "CS101 Introduction to C++ Programming", gradesArray );
15    myGradeBook.displayMessage();
16    myGradeBook.processGrades();
17    system("PAUSE");
18    return 0;
19 } // end main
20
21
```

Fonksiyonlarda Dizi Kullanımı

- GradeBook örnek program - devam.



41

Konular

- Giriş
- Diziler
- Dizi Kullanımı
- Karakter Dizileri
- Fonksiyonlarda Dizi Kullanımı
- Dizilerde Arama
- Dizilerde Sıralama
- Çok Boyutlu Diziler

Dizilerde Arama

- Dizi içinde belirli bir kriteri sağlayan elemanın bulunması için farklı arama yöntemleri kullanılır.
- Sıralı dizilerde elemanın olup olmadığını bulmak için **binary search** yöntemi kullanılabilir.
- Sırasız dizilerde **linear search** yöntemi kullanılabilir.
- Binary search her adımda aranan kümenin ortasındaki elemana bakar. Aranan değerden küçük/büyük olma durumunda göre sol veya sağ kısmını eler.
- Linear search her adımda elemanlardan birisini karşılaştırır ve arama işlemini baştan sona doğru sıralı yapar.

43

Dizilerde Arama

Linear search

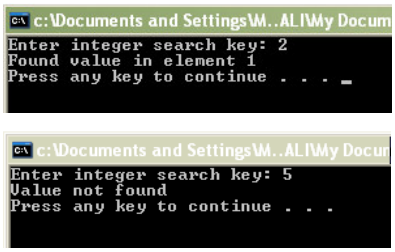
```
1  // Fig. 7.19: fig07_19.cpp
2  // Linear search of an array.
3  #include <iostream>
4  using std::cout;
5  using std::cin;
6  using std::endl;
7
8  int linearSearch( const int [], int, int ); // prototype
9
10 int main()
11 {
12    const int arraySize = 100; // size of array a
13    int a[ arraySize ]; // create array a
14    int searchKey; // value to locate in array a
15
16    for ( int i = 0; i < arraySize; i++ )
17       a[ i ] = 2 * i; // create some data
18
19    cout << "Enter integer search key: ";
20    cin >> searchKey;
21
22    // attempt to locate searchKey in array a
23    int element = linearSearch( a, searchKey, arraySize );
24
25    // display results
26    if ( element != -1 )
27       cout << "Found value in element " << element << endl;
28    else
29       cout << "Value not found" << endl;
30
31    system("PAUSE");
32    return 0; // indicates successful termination
33 } // end main
34
```

44

Dizilerde Arama

Linear search

```
35 // compare key to every element of array until location is
36 // found or until end of array is reached; return subscript of
37 // element if key or -1 if key not found
38 int linearSearch( const int array[], int key, int sizeOfArray )
39 {
40    for ( int j = 0; j < sizeOfArray; j++ )
41       if ( array[ j ] == key ) // if found,
42          return j; // return location of key
43
44    return -1; // key not found
45 } // end function linearSearch
46
```



45

Konular

- Giriş
- Diziler
- Dizi Kullanımı
- Karakter Dizileri
- Fonksiyonlarda Dizi Kullanımı
- Dizilerde Arama
- Dizilerde Sıralama
- Çok Boyutlu Diziler

Dizilerde Sıralama

Insertion sort

- Bir dizideki elemanların sıralanması için çok sayıda algoritma vardır (selection sort, insertion sort, bubble sort, quick sort, shell sort, merge sort, heap sort, radix sort, count sort, bucket sort).
- Insertion sort her adımda dizideki bir elemanı bulunduğu yere taşır.
- Elemanın yerleştirildiği yerden sonraki tüm elemanların bir kaydırılması gerekir.

47

Dizilerde Sıralama

Insertion sort

A S O R T I N G E X A M P L E
A (S) O R T I N G E X A M P L E
A (O) S R T I N G E X A M P L E
A O (R) S T I N G E X A M P L E
A O R S (T) I N G E X A M P L E
A (I) O R S T N G E X A M P L E
A I (N) O R S T G E X A M P L E
A (G) I N O R S T E X A M P L E
A (E) G I N O R S T X A M P L E
A E G I N O R S T (X) A M P L E
A (A) E G I N O R S T X M P L E
A A E G I (M) N O R S T X P L E
A A E G I M N O (P) R S T X L E
A A E G I (L) M N O P R S T X E
A A (E) E G I L M N O P R S T X
A A E E G I L M N O P R S T X

48

Dizilerde Sıralama

Insertion sort

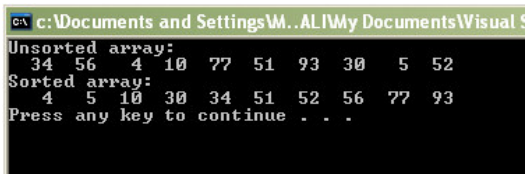
```
1  // Fig. 7.20: fig07_20.cpp
2  // This program sorts an array's values into ascending order.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  #include <iomanip>
8  using std::setw;
9
10 int main()
11 {
12    const int arraySize = 10; // size of array a
13    int data[ arraySize ] = { 34, 56, 4, 10, 77, 51, 93, 30, 5, 52 };
14    int insert; // temporary variable to hold element to insert
15
16    cout << "Unsorted array:\n";
17
18    // output original array
19    for ( int i = 0; i < arraySize; i++ )
20       cout << setw( 4 ) << data[ i ];
21
```


Dizilerde Sıralama

```
22    // insertion sort
23    // loop over the elements of the array
24    for ( int next = 1; next < arraySize; next++ )
25    {
26       insert = data[ next ]; // store the value in the current element
27
28       int moveItem = next; // initialize location to place element
29
30       // search for the location in which to put the current element
31       while ( ( moveItem > 0 ) && ( data[ moveItem - 1 ] > insert ) )
32       {
33          // shift element one slot to the right
34          data[ moveItem ] = data[ moveItem - 1 ];
35          moveItem--;
36       } // end while
37
38       data[ moveItem ] = insert; // place inserted element into the array
39    } // end for
40
41    cout << "\nSorted array:\n";
42
43    // output sorted array
44    for ( int i = 0; i < arraySize; i++ )
45       cout << setw( 4 ) << data[ i ];
46
47    cout << endl;
48    system("PAUSE");
49    return 0; // indicates successful termination
50 } // end main
51
```

```
c:\Documents and Settings\M...AL I\My Documents\Visual
Unsorted array:
  34  56   4  10  77  51  93  30   5  52
Sorted array:
   4   5  10  30  34  51  52  56  77  93
Press any key to continue . . .
```

Dizilerde Sıralama



51

Konular

- Giriş
- Diziler
- Dizi Kullanımı
- Karakter Dizileri
- Fonksiyonlarda Dizi Kullanımı
- Dizilerde Arama
- Dizilerde Sıralama
- Çok Boyutlu Diziler

Çok Boyutlu Diziler

- Diziler birden fazla boyutla oluşturulabilir. Her boyut için ayrı bir alt indis kullanılır.
- 2-D diziler yaygındır ve satır ve sütunlar halinde tabloları göstermek için kullanılır (`a[i][j]`).

```
int b[ 2 ][ 2 ] = { { 1, 2 }, { 3, 4 }};
```

- `b[0][0]= 1, b[0][1]= 2, b[1][0]= 3, b[1][1]= 4` değerini alır.
- Bir satırda daha az değer atanırsa kalanlara 0 otomatik olarak aktarılır.

```
int b[ 2 ][ 2 ] = { { 1 }, { 3, 4 }};
```

- `b[1][2]` için 0 değeri atanır.

53

Çok Boyutlu Diziler

- İki boyutlu dizilerde ilk indis satır ikinci indis sütunu gösterir.

	Sütun 0	Sütun 1	Sütun 2	Sütun 3
Satır 0	`a[0][0]`	`a[0][1]`	`a[0][2]`	`a[0][3]`
Satır 1	`a[1][0]`	`a[1][1]`	`a[1][2]`	`a[1][3]`
Satır 2	`a[2][0]`	`a[2][1]`	`a[2][2]`	`a[2][3]`

54

Çok Boyutlu Diziler

- Çok boyutlu dizilere değer atama.

```
1  // Fig. 7.22: fig07_22.cpp
2  // Initializing multidimensional arrays.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  void printArray( const int [][ 3 ] ); // prototype
8
9  int main()
10 {
11    int array1[ 2 ][ 3 ] = { { 1, 2, 3 }, { 4, 5, 6 } };
12    int array2[ 2 ][ 3 ] = { 1, 2, 3, 4, 5 };
13    int array3[ 2 ][ 3 ] = { { 1, 2 }, { 4 } };
14
15    cout << "Values in array1 by row are:" << endl;
16    printArray( array1 );
17
18    cout << "\nValues in array2 by row are:" << endl;
19    printArray( array2 );
20
21    cout << "\nValues in array3 by row are:" << endl;
22    printArray( array3 );
23    system("PAUSE");
24    return 0; // indicates successful termination
25 } // end main
26
```


Çok Boyutlu Diziler

- Çok boyutlu dizilere değer atama – devam.

```
27 // output array with two rows and three columns
28 void printArray( const int a[][ 3 ] )
29 {
30    // loop through array's rows
31    for ( int i = 0; i < 2; i++ )
32    {
33       // loop through columns of current row
34       for ( int j = 0; j < 3; j++ )
35          cout << a[ i ][ j ] << ' ';
36
37       cout << endl; // start new line of output
38    } // end outer for
39 } // end function printArray
40
41
```

```
c:\Documents and SettingsM...ALI\My Docur
Values in array1 by row are:
1 2 3
4 5 6

Values in array2 by row are:
1 2 3
4 5 0

Values in array3 by row are:
1 2 0
4 0 0
Press any key to continue . . .
```

Ödev

- Aşağıdaki gibi 3x3 bir tablo oluşturunuz ve bir tanesi hariç her birisinin içine **rastgele** 1 ile 8 arasında rakamlar atayınız. Program ilk çalıştığında aşağıdaki gibi bir ekran oluşturacaktır. Her rakam bir kez atanacaktır.

```
1 3 7                                                  1 3 7
  4 6       4'e basıldıktan sonraki durum ---->        4   6
8 2 5                                                  8 2 5
```

- Kullanıcı 1 ile 9 arasındaki bir tuşa bastığında eğer boşluk basılan tuşun etrafında ise basılan rakamla boşluğu yer değiştirecektir. Geçişler sadece yatay ve dikey olacaktır.
- Eğer basılan tuşun etrafında boşluk yoksa herhangi bir işlem yapmayacatır.
- Her tuşa basılıp yer değiştirmeden sonra ekran yenilenecektir.
- Aşağıdaki sıralamanın oluşup oluşmadığı her yer değiştirmeden sonra kontrol edilecek ve sıralama oluşmuşsa "Tebrikler doğru sıraladınız." mesajı görüntülenecektir.

```
1 2 3
4 5 6
7 8
```

57

Ödev

- Kullanıcıya her doğru sıralamadan sonra yeniden başlamak isteyip istemediği sorulacak ve devam etmek isterse yeniden rastgele bir oyun başlatılacaktır.
- Kullanıcı oyun devam ederken "Y" tuşuna basarsa "Yeniden başlamak istediğinizden eminmisiniz ?" sorusu sorulacak ve evet ("E") cevabı girilirse oyun rastgele sayılarla yeniden başlatılacaktır.
- Klavyeden tuşa basıldığını kontrol etmek ve basılan tuşu okumak için aşağıdaki kod kullanılabilir.

```
char ch = getch();
```

- **getch()** fonksiyonu için aşağıdaki kütüphane include edilmelidir.

```
#include <conio.h>
```

58