Programming In C++

Course 2: Lecture 2, Arrays

Prepared By Dr. Ali Al-Sabaawi

Nineveh University- Faculty of IT- department of software

1-D Arrays

- A program to add two arrays.

```
1   #include <iostream>
2
3   using namespace std;
4
5   int main()
6   {
7      int arr1[5],arr2[5],arr3[5],i;
8      for(i=0;i<5;i++)
9         cin>>arr1[i]>>arr2[i];
10
11     for(i=0;i<5;i++)
12        arr3[i]=arr1[i]+arr2[i];
13
14     for(i=0;i<5;i++)
15        cout<<arr3[i]<<" ";
16     return 0;
17  }
18
```

1-D array

Lab work:

1- read an array and check how many prime numbers in this array?

```
#include <iostream>
using namespace std;
int main()
{
int arr[5]={7,50,4,20,8};
int z=0;
for(int i=0;i<5;i++) {
   int  x=arr[i];
   int n=0;

   for(int j=1;j<=x;j++)
      if(x%j==0)
      n++;

      if(n==2)
         z++;

}
cout<<"There are "<<z<<" prime number ";
   return 0;
}
```

2- read an array and find the max value of this array.

2-D arrays

An array is useful for storing and working with a set of data. Sometimes, though, it's necessary to work with multiple sets of data.

For example, in a grade-averaging program a teacher might record all of one student's test scores in an array of doubles.

If the teacher has 30 students, that means she'll need 30 arrays of doubles to record the scores for the entire class. Instead of defining 30 individual arrays, however, it would be better to define a two-dimensional array.

	Column 0	Column 1	Column 2	Column 3
Row 0	`scores[0] [0]`	`scores[0] [1]`	`scores[0] [2]`	`scores[0] [3]`
Row 1	`scores[1] [0]`	`scores[1] [1]`	`scores[1] [2]`	`scores[1] [3]`
Row 2	`scores[2] [0]`	`scores[2] [1]`	`scores[2] [2]`	`scores[2] [3]`

Declaring 2-D array

To define a two-dimensional array, two size declarators are required. The first one is for the number of rows, and the second one is for the number of columns.

```
double scores[3][4];
```

The element of the first row (row 0)

```
scores[0][0]
scores[0][1]
scores[0][2]
scores[0][3]
```

The elements of row 1 are

```
scores[1][0]
scores[1][1]
scores[1][2]
scores[1][3]
```

The elements of row 2 are

```
scores[2][0]
scores[2][1]
scores[2][2]
scores[2][3]
```

To assign the value 92.25 to the element at row 2, column 1 of the scores array

Scores [2][1]=92.25;

Initialization 2-D array

One way of initializing a two-dimensional array is

```
int hours[3][2] = {{8, 5}, {7, 9}, {6, 3}};
```

Second way of assigning elements to array

```
hours[0][0] = 8
hours[0][1] = 5
hours[1][0] = 7
hours[1][1] = 9
hours[2][0] = 6
hours[2][1] = 3
```

Third way is using loops

```
for (int i = 0; i < 3; i++)
        for (int j = 0; j < 2; j++)
            cin >> hours[i][j] ;
```

2-D array

Summing the Rows of a Two-Dimensional Array.

Suppose a two-dimensional array is used to hold a set of test scores for a set of students. Each row in the array is a set of test scores for one student. To get the sum of a student's test scores (perhaps so an average may be calculated), you use a loop to add all the elements in one row.

2-D array

```
3   int main()
4   {
5   const int NUM_STUDENTS = 3;      // Number of students
6   const int NUM_SCORES = 5;        // Number of test scores
7   double total;                    // Accumulator is set in the loops
8   double average;                  // To hold each student's average
9   double scores[NUM_STUDENTS][NUM_SCORES] = {{88, 97, 79, 86, 94},{86, 91, 78, 79, 84},{82, 73, 77, 82, 89}};
10  // Get each student's average score.
11  for (int row = 0; row < NUM_STUDENTS; row++)
12  {
13    // Set the accumulator.
14    total = 0;
15    // Sum a row.
16    for (int col = 0; col < NUM_SCORES; col++)
17       total += scores[row][col];
18    // Get the average.
19    average = total / NUM_SCORES;
20    // Display the average.
21    cout << "Score average for student "<< (row + 1) << " is " << average <<endl;
22  }
23     return 0;
24  }
```

```
Score average for student 1 is 88.8
Score average for student 2 is 83.6
Score average for student 3 is 80.6
```

2-D array

Create 2-D array 3*4 and then find the summation of the each column, assign the summation in 1-D array.

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5  int arr[3][4]={{10,20,30,40},{50,60,70,80},{90,100,110,120}}, arr2[4];
6  int sum;
7  for(int col=0;col<4;col++) {
8        sum=0;
9      for(int row=0;row<3;row++)
10       sum+=arr[row][col];
11    arr2[col]= sum;
12 }
13
14 for(int col=0;col<4;col++)
15    cout<<arr2[col]<<" ";
16       return 0;
17 }
```

The End