모듈 21

2차원 배열Array들

2차원 배열을 만들고, 그 값을 표에 넣어서 표현할 수 있게 됩니다.

이미 배운 바와 같이, 배열은 타입이 동일한 요소들이 줄지어 있는 것으로 이름은 하나이지만 각자 다른 인덱스번호를 가지고 있다. 모든 배열에서 각 요소는 (메모리 안에서) 서로 옆에 붙어 있다.

첫 번째 요소	두 번째 요소	세 번째 요소	···	N 번째 요소

우리는 숫자나 문자열 또는 다른 타입으로 된 데이터들을 한 줄로 늘어 놓고 작업할 때 1차원 배열을 사용한다.

모든 배열은, 본질적으로, 1차원으로 보일 수 있어야 한다. 2차원이나 다차원 배열은 데이터를 보다 쉽게 표현하고 다루기 위해 필요하다.

2차원 배열에는 인덱스번호가 2개 있고, N차원 배열에는 인덱스번호가 N개 있다.

표 형식으로 되어 있는 데이터와 관련된 문제를 해결하는 과정은 2차원 배열을 사용하면 편리하다.

예를 들어, 데이터를 받은 후에, 열이 3개이고, 행이 5개인 표로 표시하고 다루어 보자.

첫 번째 열					두 번째 열					세 번째 열				
el 1	el 2	el 3	el 4	el 5	el 6	el 7	el 8	el 9	el 10	el 11	el 12	el 13	el 14	el 15

이 배열의 요소Element들

위에 있는 데이터는 열 3개로 구성된 배열 하나인데, 요소인 각각의 열은 행 5개로 구성된 배열이다. 이것을 '배열들의 배열'이라고 부른다.

이런 종류의 배열은 *StringGrid*에 다음과 같이 들어간다.

	첫 번째 열	두 번째 열	세 번째 열
첫 번째 행	요소1	요소 6	요소 11
두 번째 행	요소 2	요소 7	요소 12
세 번째 행	요소 3	요소 8	요소 13
네 번째 행	요소 4	요소 9	요소 14
다섯 번째 행	요소 5	요소 10	요소 15

이 배열을 선언해 보자.

먼저 열 배열을 선언한다.

```
const
   m=3;
type
   TCol_Array=Array[1..m] of TRow_Array;
```

이제, 각 열 안에 들어갈 행 배열을 선언하여 덧붙인다.

```
const
   m=3;
   n=5;
type
   TRow_Array=Array[1..n] of Integer;
   TCol_Array=Array[1..m] of TRow_Array;
```

다른 예로, 데이터를 받은 후에, 행이 3개이고, 열이 5개인 표로 표시하고 다루려면,

첫 번째 행					두 번째 행					세 번째 행				
el 1	el 2	el 3	el 1	el 2	el 3	el 1	el 2	el 3	el 1	el 2	el 3	el 1	el 2	el 3

이 배열의 요소Element들

위에 있는 데이터는 행 3개로 구성된 배열 하나인데, 요소인 각각의 행은 열 5개로 구성된 배열이다. 우리는 '배열들의 배열'을 또 하나 가지게 되었다.

이 경우, *StringGrid*에는 다음과 같이 들어간다.

	첫 번째 열	두 번째 열	세 번째 열	네 번째 열	다섯 번째 열
첫 번째 행	요소 1	요소 2	요소 3	요소 4	요소 5
두 번째 행	요소 6	요소 7	요소 8	요소 9	요소 10
세 번째 행	요소 11	요소 12	요소 13	요소 14	요소 15

이 배열 타입을 정의해보자.

먼저, 행 배열을 선언한다.

```
const
   n=3;
type
   TRow_Array=Array[1..n] of TCol_Array;
```

이제, 각 행 안에 들어갈 열 배열을 선언하여 덧붙인다.

```
const
   n=3;
   m=5;
Type
   TCol_Array =Array[1..m] of Integer;
   TRow_Array =Array[1..n] of TCol_Array;
```

위의 예문들처럼, 2차원 배열(과 N차원 배열)은 배열들의 배열로 표현할 수 있다. 2차원 배열은 행들의 배열 또는 열들의 배열이 될 수 있다.

2차원 배열 A의 요소들을 다룰 때에는 다음과 같이 한다.

A[1][4] 또는 A[3][5] 또는 A[I][j], I 는 2차원 배열에서의 인덱스번호이고, j는 1차원 배열(바깥쪽 2차원 배열 안의 요소 각각에 들어 있는 1차원 배열)에서의 인덱스번호이다.

예제를 가지고 작업해보자.

열 M개와 행 N개로 구성된 2차원 배열의 정수 요소에 -10 에서 10까지 (-10과 10을 포함하여) 무작위 숫자로 채운다. 그리고 이 배열을 *StringGrid* 콘트롤 요소에 표현한다. *StringGrid*의 고정 셀[fixed cell] 인 열과 행의 개수는 0으로 하는 것이 좋겠다.

먼저, 배열 타입을 선언한다.

```
const
   n=3;
   m=6;
type
   TArray_1=Array[1..n] of Integer;
   TArray_2=Array[1..m] of TArray_1;
```

이제, 해법을 적어보자.

```
var
   c: TArray_2;
begin
   FillArray(c);//배열 C를 채워 넣는다.
   TableOutput(c);//배열 C를 문자표로 내보낸다.
end;
```

이제, 프로시저들을 작성하자. 열 배열을 먼저 채우는 프로시저부터 시작해보자.

```
procedure FillArray(var arr: TArray_2);
    var i: Integer;
begin
    Randomize;
    for i:=1 to m do //i가 1 에서 시작, M이 될 때까지 반복
        FillColumn(arr[i]); //i번째 열배열을 채워 넣는다.
end;
```

이제, 열을 무작위 숫자로 채우는 프로시저가 필요하다. 숫자의 범위는 −10에서 10까지이다.

```
procedure FillColumn(var ma_1: TArray_1);
var
    j: Integer;
begin
    for j:= 1 to n do //j가 1 에서 시작, N이 될 때까지 반복
        ma_1[j]:= Random(21)-10; //j 번째 배열 요소에는 -10에서 10까지(-10과 10도 포함)
의 숫자가 무작위로 들어간다.
end;
```

만들어진 결과 배열을 *StringGrid*로 내보내는 프로시저는 다음과 같이 정의한다.

폼의 이름은 `frm2arr`로, *StringGrid*의 이름은 `sgdMy`라고 하자.

```
procedure TableOutput(arr: TArray_2);
var
    i, j: Integer;
begin
    frm2arr.sgdMy.FixedCols:=0; //고정열의 개수
    frm2arr.sgdMy.FixedRows:=0; //고정행의 개수
    frm2arr.sgdMy.ColCount:=m; //열의 개수
    frm2arr.sgdMy.RowCount:=n; //행이 개수
    for i:=1 to m do
        for j:=1 to n do
            frm2arr.sgdMy.Cells[i-1, j-1]:=IntToStr(arr[i][j]);
end;
```

우리가 이 배열의 행과 열을 서로 바꾸어서 *StringGrid*로 내보낸다면 어떻게 될까? 이경우, 행은 3개가 되고, 각 행마다 열 6개씩 들어간다. *StringGrid*의 셀을 지정할 때 앞의 숫자는 열의 인덱스번호이고, 뒤의 숫자는 행의 인덱스번호라는 점을 잊지 말자.

이 배열을 *StringGrid*로 내보내는 프로시저는 다음과 같다.

```
procedure TableOutput(arr: Array_2);
var
   i, j: Integer;
begin
   frm2arr.SgdMy.ColCount:= n; //n – 배열에 있는 행의 개수
   frm2arr.SgdMy.RowCount:= m; //m – 배열에 있는 열의 개수
   for i:=1 to n do
       for j:=1 to m do
           frm2arr.SgdMy.Cells[i-1, j-1]:= IntToStr(arr[j][i]);
end;
```

실습

Exercise 1.

M개의 열과 N개의 행으로 구성된 2차원 배열 하나를 무작위 숫자로 채운다.

텍스트박스 2개에서 각각 숫자 범위의 최소값과 최대값은 받는다.

만들어진 배열을 *StringGrid* 표로 내보낸다. 고정 셀인 열과 행의 개수는 0이다.

TMemo 컴포넌트에는 다음의 값들을 적어 넣는다.

- 배열에 있는 모든 요소들의 합
- 배열에 있는 값들 중에서 최소값과 최대값
- 각 열 별로 안에 있는 요소들의 합

모든 결과는 설명과 함께 적는다. 예를 들어,

- 배열에 있는 모든 요소들의 합 = 234.
- 최대값 요소 = 68.
- 최소값 요소 = 5.
- 1 번째 열에 있는 요소들의 합 = 94.
- 2 번째 열에 있는 요소들의 합 = 43.

Exercise 2.

2차원 배열을 무작위 *Real*실수들로 채운다. 이 배열은 N개의 행과 M개의 열을 가진다.

TEdit 컴포넌트 2개에서 무작위 숫자의 범위를 입력 받는다.

StringGrid 컴포넌트를 만든다. 고정 셀인 열과 행의 개수는 1로 설정한다.

만들어진 결과 배열을 이 *StringGrid* 컴포넌트로 내보낸다.

StringGrid 컴포넌트의 각 행마다 마지막 셀에 그 행의 요소들의 합계를 계산하여 넣는다.