모듈 12

String 문자열 들

문자열을 원하는 데로 비교하고 조작할 수 있게 됩니다.

우리는 이미 델파이에서 *String*을 경험해보았다(한글로는 '문자열'이라고 하겠다). 예를 들어, *Caption*이나 *Text* 같은 프로퍼티에는 문자열 값만 들어갈 수 있다. 문자열이란 정확히 무엇이고 어떻게 사용할까?

문자열이란 기호(문자 등)를 나열하여 작은따옴표 안에 넣어 하나로 만든 것을 말한다. 문자열 변수를 선언하려면 *String* 타입을 사용해야 한다.

```
var
   s: String;
```

위 정의는 길이 제한이 정해지지 않은 문자열을 프로그램에서 사용할 것이라는 의미이다. 문자열은 이어 붙일 수 있다. 문자열 연결은 더하기 기호를 사용한다. 예를 들어,

```
var
   s, st: String;
begin
   s:= '델파이를'; //첫 번째 변수에 "델파이를"을 넣는다.
   st:= ' 배운다'; //두 번째 변수에 "배운다"를 넣는다.
   s:=s+st; //이 두 String을 이어 붙인다.
end;
```

프로그램을 실행하면 "델파이를 배운다"라는 문자열이 변수 s 안에 들어간다.

문자열은 서로 비교할 수도 있다.

문자열 안의 각 기호(문자 등)가 하나씩 비교된다(내부 표현 방식을 비교한다). 영문자에는 알파벳 순서가 있고 숫자에는 0 〈1 〈…〈9 처럼 순서가 있다.

예문

'AB' 〉 'A A'
'A' 〈 'AB'
'DC' 〉 'ABCDE'
'ABCE' 〉 'ABCD'

문자열을 다루는 표준 함수들과 프로시저들 몇 가지를 요약표에서 살펴보자.

표준 문자열 프로시저들

String 프로시저들		
이름 (파라미터, Parameters)	파라미터 타입	의미
Delete(St, Pos, N)	St: String; Pos, N: Integer;	문자열 St에서 Pos 지점부터 N개의 기호(문자 등)를 삭제
Insert(St1, St2, Pos)	St1, St2: String; Pos: Integer;	문자열 St1을 문자열 St2의 Pos 지점에 삽입

표준 문자열 함수들

String 함수들		
이름 (파라미터, Parameters)	파라미터 타입	의미 (Semantics)
Copy(St, Pos, N)	반환 타입: String; St: String; Pos, N: Integer;	문자열 St에서 Pos 지점부터 N개의 기호(문자 등)을 복사한 값을 반환
Length(St)	반환 타입: Integer; St: String;	문자열 St의 길이 (문자 등 기호 전체의 개수)를 반환
Pos(St1, St2)	반환 타입: Integer; St1, St2: String;	문자열 St1 안에 있는 문자열 St2 부분을 찾아서 그 첫 번째 위치를 반환. St2가 없으면 0을 반환

예문

– Delete 프로시저

St 의 값	구문	구문이 실행된 후의 St 의 값
'abcdef'	Delete(St,4,2)	'abcf'
'Turbo–Pascal'	Delete(St,1,6)	'Pascal'

– Insert 프로시저

St1 의 값	St2 의 값	구문	구문이 실행된 후의 St2 의 값
'Turbo'	'–Pascal'	Insert(St1, St2,1)	'Turbo–Pascal'
'–Pascal'	'Turbo'	Insert(St1, St2,6)	'Turbo–Pascal'

– Copy 함수

St 의 값	구문	구문이 실행된 후의 Str 의 값
'abcdefg'	Str:=Copy(St,2,3);	'bcd'
'abcdefg'	Str:=Copy(St,4,4);	'defg'

– Length 함수

St 의 값	구문	구문이 실행된 후의 N 의 값
'abcdefg'	N:=Length(St);	7
'Turbo-Pascal'	N:=Length(St);	12

– Pos 함수

St2 의 값	구문	구문이 실행된 후의 N 의 값
'abcdef'	N:=Pos('de' , St2);	4
'abcdef'	N:=Pos('r' , St2);	0

응용 예문:

두 단어가 공백 하나로 구분되어 있는 문자열 하나를 입력 받는 프로시저가 있다. 이 프로시저는 입력 받은 문자열의 두 단어의 순서를 바꾼다.

```
Procedure Change(var s: String);
var
   s1: String;
begin
   s1:= copy(s,1, pos('␣', s)-1); //문자열 s 에 있는 첫 번째 단어(즉, 첫 번째 공백 앞에 있는 모든 문자)를 복사한다.
   delete(s,1, pos('␣', s)); //첫 번째 단어와 공백을 삭제한다.(s에는 두 번째 단어만 남는다)
   s:= s+'␣'+s1; //s 뒤에 공백과 복사해 둔 첫 번째 단어를 붙인다.
end;
```

NOTE | 기호 ␣ 는 실제로는 공백이다. 알아보기 쉽도록 표시한 것일 뿐이다.

실습

Exercise 1.

텍스트박스에 세 단어가 들어가는 문장을 입력하고 버튼을 누르면 문장에서 두 번째 단어와 세 번째 단어의 위치를 서로 바꾸는 프로그램을 작성해보자. 텍스트박스에는 세 단어만 들어가고 단어 사이는 공백 한 칸으로 되어야 한다.

Exercise 2.

버튼을 누르면 텍스트박스의 모든 공백이 느낌표로 바뀌도록 해보자.

Exercise 3.

텍스트박스 안에 있는 문자열에서 마침표의 개수를 알아내자.

Exercise 4.

텍스트박스 안에 있는 문자열에서 문자열 "가나다"가 몇 개 들어있는지를 알아내자.

Exercise 5.

텍스트박스 안에 괄호로 감싸진 문자열을 넣고 버튼을 누르면, 괄호를 빼고 나머지 안에 있는 문자열만 옆에 있는 또 다른 텍스트박스 안으로 들어가도록 해보자.

Exercise 6.

텍스트박스 안에 입력된 문자열에서 공백으로 구분된 단어의 개수를 알아내자.

Exercise 7.

텍스트박스 안에 입력된 문자열에서 "강아지"라는 단어는 모두 "고양이"로 바꿔보자.

Exercise 8.

텍스트박스 안에 있는 문자열을 뒤에서 앞으로 뒤집어 적어보자.

Exercise 9.

텍스트박스에 입력된 문자열에서 공백의 숫자를 세어보자. 그리고 앞에서 두 번째 단어와 뒤에서 두 번째 단어를 서로 바꿔보자.