Sample: Pascal - Variable Length String Data Type

```
unit vstrings;

interface

//Write the Pascal variable length string abstract data type (ADT)

const
  MAX_LENGTH = 100; // arbitrary

type
  vstring = array [0..MAX_LENGTH] of char;
  // your definition goes here; you cannot use the built-in
// string type in your definition of vstring

procedure vwrite(str: vstring);
// Write the vstring to standard output with no new line

procedure vread(var str: vstring);
// Reads an entire line from standard input and returns it as a
// vstring. End-of-line characters are not inserted in the returned
// vstring.

function vconvert(str: string): vstring;
// Returns a vstring equivalent of str; if necessary,
// the returned string is truncated so it is not longer
// than MAX_LENGTH

function vlength(str: vstring): integer;
// Returns the length of str, i.e. the number of characters in the vstring
```

```
function vconcat(str1, str2: vstring): vstring;
// Returns a vstring which is the concatenation of its arguments
// in the order in which they are specified. The returned vstring cannot
// exceed MAX_LENGTH - ignore all characters in str2 that might cause this
// limit to be exceeded

function vequal(str1: vstring; str2: vstring): boolean;
// Returns true if two strings contain exactly the same characters in
// the same order, false otherwise.

function vcharAt(str: vstring; n: integer): char;
// Returns the character specific index n where 1 <= n <= length(str);
// the first character in the string is at index 1; returns the empty
// character (ASCII 0) if n is outside this range

function vcopy(str: vstring; pos: integer; num: integer): vstring;
// Returns a substring containing num characters from str starting at
// position pos. If pos < 1 or pos > vlength(str) or num < 1, the empty
// vstring is returned. If an attempt is made to get characters beyond
// the end of the vstring (i.e. (pos + num) > vlength(str)), only the
// characters within the vstring are returned

procedure vdelete(var str: vstring; pos: integer; num: integer);
// Removes a substring containing num characters from str starting at
// position pos (where the first character in str is at position 1).
// If pos < 1 or pos > vlength(str), no characters are removed. If an
// attempt is made to delete characters beyond the end of the vstring
// (i.e. (pos + num) > vlength(str)), only characters within the vstring
```

```
// are deleted.

procedure vinsert(target, source: vstring; var result: vstring; pos: integer);
// Inserts the vstring source into the vstring result at position pos.
// If pos > vlength(result), source is concatenated to result. If
// (vlength(source) + vlength(target)) > MAX_LENGTH, excess characters
// are truncated and result will only contain the leftmost characters
// of source at position pos. If pos < 1, no characters are inserted.

implementation

procedure vwrite(str: vstring);
var i:Integer;
begin
  for i:=1 to MAX_LENGTH do
    if Ord(str[i])<>0 then
       write(str[i])
     else Break;
end;

procedure vread(var str: vstring);
var s:string;
begin
readln(s);
if Length(s)=1 then
str[1]:=char(s[1])
else
str:=vconvert(s);
end;
```

```
function vconvert(str: string): vstring;
var i:Integer;
begin
 for i:=1 to MAX_LENGTH do
   if Ord(str[i])<>0 then
      Result[i]:=Char(str[i])
   else Break
end;

function vlength(str: vstring): integer;
var i:Integer;
begin
 for i:=1 to MAX_LENGTH do
   if Ord(str[i])<>0 then
      Result:=Result+1
   else Break;
end;

function vconcat(str1, str2: vstring): vstring;
var i,k:Integer;
begin
 Result:=str1;
 k:=vlength(Result)+1;
 for i:=1 to vlength(str2) do
   if (Ord(str2[i])<>0) and (k<=MAX_LENGTH) then
   begin
    Result[k]:=char(str2[i]);
    k:=k+1;
```

```
   end else Break;
end;

function vequal(str1: vstring; str2: vstring): boolean;
var i:Integer;
begin
Result:=True;
 for i:=1 to MAX_LENGTH do
   if Ord(str1[i])<>Ord(str2[i]) then
   begin
     Result:=False;
     Break;
   end;
end;

function vcharAt(str: vstring; n: integer): char;
begin
if (n>=1) and (n<=vlength(str)) then
Result:=str[n]
else
Result:=chr(0);
end;

function vcopy(str: vstring; pos: integer; num: integer): vstring;
var k,i:Integer;
begin
if (num > 1) and ( pos >= 1 ) and ( pos <= vlength(str) ) then
begin
 if (pos + num) > vlength(str) then num:=(vlength(str)-pos)+1;
```

```
 k:=1;
 for i:=pos to vlength(str) do
   if (Ord(str[i])<>0) and (num>0) and (k<=MAX_LENGTH) then
   begin
     Result[k]:=str[i];
     k:=k+1;
     num:=num-1;
   end else Break;
end else Result:=vconvert('');
end;

procedure vdelete(var str: vstring; pos: integer; num: integer);
var k,i:Integer; str2:vstring;
begin
if (num > 1) and ( pos >= 1 ) and ( pos <= vlength(str) ) then
begin
 if (pos + num) > vlength(str) then num:=vlength(str)-pos+1;
 k:=1;
 str2:=str;
 for i:= 1 to vlength(str) do str[i]:=chr(0);
 for i:=1 to vlength(str2) do
   if Ord(str2[i])<>0 then
   begin
     if (i<pos) or (i>(num+pos)-1) then
     begin
       str[k]:=str2[i];
       k:=k+1;
     end;
   end else Break
```

```
end;
end;

procedure vinsert(target, source: vstring; var result: vstring; pos: integer);
var k,i:Integer;
begin
if pos < 1 then
begin
result:=source;
Exit;
end;
if pos > vlength(source) then
begin
 result:=vconcat(source,target)
end else
begin
 result:=source;
 vdelete(result,pos,((vlength(result)-pos)+1));
 k:=pos;
 for i:=1 to vlength(target) do
  if (Ord(target[i])<>0) and (vlength(result)<=MAX_LENGTH) then
  begin
   result[k]:=target[i];
   k:=k+1;
  end else Break;
 if vlength(result) < MAX_LENGTH then
  begin
  result:=vconcat(result,vcopy(source,pos,((vlength(source)-pos)+1)));
  end;
```

```
end;
end;

end.
```