Problem Set 1

OUT: 1 SEPTEMBER, 1994
DUE: 13 SEPTEMBER, 1994

Goals

Introduction to Loop Invariants and Data Objects.

Reading Assignment

Review chapters 5, 7 and 8 in *Condensed Pascal.*

Assignment

Supplement the procedures `writeln_myString` and `readln_myString` with the following functions. Document your loops with Preconditions, Loop Invariants and Postconditions.

```
procedure copy_myString( var dest : myString; src : myString ) ;
(*   makes a copy of src in dest. src is unchanged. *)
procedure append_myString( var dest : myString; src : myString ) ;
(*   appends src to the end of dest, src is unchanged. *)
function length_myString( s : myString ) : integer ;
(*   returns integer length of s. *)
function compare_myString( s1 : myString ; s2 : myString ) : boolean ;
(*   returns TRUE iff s1=s2. Two myStrings are equal iff they
     are the same length and have the same character in each
     position. *)
```

Write a procedure to take the first full word off of a myString, placing it in a second user provided myString. Words are delimited by white-space and all punctuation except the hyphen.

```
procedure parse_myString( var src : myString ; var oneWord : myString ) ;
(*    the first word is taken off of src and placed in oneWord.
      if there is no more words in src, oneWord is returned with
      the empty string. src is updated, removing oneWord, i.e.
      successive calls to parse_myString will give successive
      words from src. *)
```

Combine the procedures into a test program which rewrites the input to the output, one word on a line, staring the line if the word appears just after a copy of itself. Spawn a subprocess to accept input from a file named *test.run.*

```
$ type test.run
run test.exe
  Romeo, Romeo!   wherefore
  art    thou,    Romeo   ?
$ spawn/input=test.run
%DCL-S-SPAWNED, process BURT_1 spawned
%DCL-S-ATTACHED, terminal now attached to process BURT_1
 Romeo
*Romeo
 wherefore
 art
 thou
 Romeo
%DCL-S-RETURNED, control returned to process BURT
$
```

Create subdirectories `INBOX` and `OUTBOX` on your VMS accounts. Place the program, test.run files and additional test programs in `OUTBOX` before class on the due date.