CSE220 Class# 12166 Lab3 Team18
Nicholas Murray
Vivian Vinh
Timothy Zamora

Test Case for System Test

System test description:

The program reads a line from a Pascal source code file (NEWTON.PAS), output that to the output file (ActualOutput.txt) with the line number and then proceeds to move through the line character by character building tokens. Each token must be identified as what it is; a keyword (reserved word) in Pascal, a literal (which is an identifier, a number, or a string), or one of many special characters in Pascal.

System test table:

Function to be test	Procedure name	Input parameter	Output or Return value	Expected result	Test command line
Extract the content of an input file (NEWTON.PAS) and produce an identical output (ActualOutput.txt) with line numbers	int main(int argc, const char * argv[])	NEWTON.PAS	ActualOutput.txt	Expected output (ActualOutput.txt) is identical to input (sample_output.txt) with each line numbered.	./lab3 NEWTON.PAS > ActualOutput.txt

Input data (NEWTON.PAS):

```
PROGRAM newton (input, output);

CONST
  epsilon = 1e-6;

VAR
  number, root, sqroot : real;

BEGIN
  REPEAT
        writeln;
        write('Enter new number (0 to quit): ');
        read(number);

        IF number = 0 THEN BEGIN
          writeln(number:12:6, 0.0:12:6);
        END
        ELSE IF number < 0 THEN BEGIN
          writeln('*** ERROR:  number < 0');
        END
```

```
        ELSE BEGIN
          sqroot := sqrt(number);
          writeln(number:12:6, sqroot:12:6);
          writeln;

          root := 1;
          REPEAT
                root := (number/root + root)/2;
                writeln(root:24:6,
                        100*abs(root - sqroot)/sqroot:12:2,
                        '%')
          UNTIL abs(number/sqr(root) - 1) < epsilon;
        END
  UNTIL number = 0
END.
```

Expected Output data (sample_output.txt):

```
Page 1    /Users/bholto/Desktop/CourseFolder_Bryce/Labs/Lab2/NEWTON.PAS Thu Mar  4 08:51:35 2014

  1: PROGRAM newton (input, output);

   >> PROGRAM        program
   >> <IDENTIFIER>   newton
   >> (              (
   >> <IDENTIFIER>   input
   >> ,              ,
   >> <IDENTIFIER>   output
   >> )              )
   >> ;              ;
  2:

  3: CONST

   >> CONST          const
  4:    epsilon = 1e-6;

   >> <IDENTIFIER>   epsilon
   >> =              =
   >> <NUMBER>       1e-06
   >> ;              ;
  5:

  6: VAR
```

```
  >> VAR            var
 7:    number, root, sqroot : real;

  >> <IDENTIFIER>    number
  >> ,              ,
  >> <IDENTIFIER>    root
  >> ,              ,
  >> <IDENTIFIER>    sqroot
  >> :              :
  >> <IDENTIFIER>    real
  >> ;              ;
 8:

 9: BEGIN

  >> BEGIN          begin
10:    REPEAT

  >> REPEAT         repeat
11:    writeln;

  >> <IDENTIFIER>    writeln
  >> ;              ;
12:    write('Enter new number (0 to quit): ');

  >> <IDENTIFIER>    write
  >> (              (
  >> <STRING>        Enter new number (0 to quit):
  >> )              )
  >> ;              ;
13:    read(number);

  >> <IDENTIFIER>    read
  >> (              (
  >> <IDENTIFIER>    number
  >> )              )
  >> ;              ;
14:
```



```
15:    IF number = 0 THEN BEGIN

  >> IF             if
  >> <IDENTIFIER>    number
  >> =              =
```

```
  >> <NUMBER>       0
  >> THEN          then
  >> BEGIN         begin
16:       writeln(number:12:6, 0.0:12:6);

  >> <IDENTIFIER>    writeln
  >> (            (
  >> <IDENTIFIER>    number
  >> :            :
  >> <NUMBER>       12
  >> :            :
  >> <NUMBER>       6
  >> ,            ,
  >> <NUMBER>       0
  >> :            :
  >> <NUMBER>       12
  >> :            :
  >> <NUMBER>       6
  >> )            )
  >> ;            ;
17:    END

  >> END          end
18:    ELSE IF number < 0 THEN BEGIN

  >> ELSE          else
  >> IF            if
  >> <IDENTIFIER>    number
  >> <             <
  >> <NUMBER>       0
  >> THEN          then
  >> BEGIN         begin
19:       writeln('*** ERROR:  number < 0');

  >> <IDENTIFIER>    writeln
  >> (            (
  >> <STRING>       *** ERROR:  number < 0
  >> )            )
  >> ;            ;
20:    END

  >> END          end
21:    ELSE BEGIN

  >> ELSE          else
  >> BEGIN         begin
```

```
22:       sqroot := sqrt(number);

  >> <IDENTIFIER>    sqroot
  >> :=              :=
  >> <IDENTIFIER>    sqrt
  >> (               (
  >> <IDENTIFIER>    number

Page 3    /Users/bholto/Desktop/CourseFolder_Bryce/Labs/Lab2/NEWTON.PAS Thu Mar  4 08:51:35 2014

  >> )               )
  >> ;               ;
23:       writeln(number:12:6, sqroot:12:6);

  >> <IDENTIFIER>    writeln
  >> (               (
  >> <IDENTIFIER>    number
  >> :               :
  >> <NUMBER>        12
  >> :               :
  >> <NUMBER>        6
  >> ,               ,
  >> <IDENTIFIER>    sqroot
  >> :               :
  >> <NUMBER>        12
  >> :               :
  >> <NUMBER>        6
  >> )               )
  >> ;               ;
24:       writeln;

  >> <IDENTIFIER>    writeln
  >> ;               ;
25:

26:       root := 1;

  >> <IDENTIFIER>    root
  >> :=              :=
  >> <NUMBER>        1
  >> ;               ;
27:       REPEAT

  >> REPEAT          repeat
28:               root := (number/root + root)/2;
```

```
  >> <IDENTIFIER>    root
  >> :=              :=
  >> (               (
  >> <IDENTIFIER>    number
  >> /               /
  >> <IDENTIFIER>    root
  >> +               +
  >> <IDENTIFIER>    root
  >> )               )
  >> /               /
  >> <NUMBER>        2
  >> ;               ;
 29:                 writeln(root:24:6,

  >> <IDENTIFIER>    writeln
  >> (               (
  >> <IDENTIFIER>    root
  >> :               :
  >> <NUMBER>        24
  >> :               :
  >> <NUMBER>        6

Page 4    /Users/bholto/Desktop/CourseFolder_Bryce/Labs/Lab2/NEWTON.PAS Thu Mar  e 08:51:35 2014

  >> ,               ,
 30:                 100*abs(root - sqroot)/sqroot:12:2,

  >> <NUMBER>        100
  >> *               *
  >> <IDENTIFIER>    abs
  >> (               (
  >> <IDENTIFIER>    root
  >> -               -
  >> <IDENTIFIER>    sqroot
  >> )               )
  >> /               /
  >> <IDENTIFIER>    sqroot
  >> :               :
  >> <NUMBER>        12
  >> :               :
  >> <NUMBER>        2
  >> ,               ,
 31:                 '%')

  >> <STRING>        %
  >> )               )
```

```
 32:        UNTIL abs(number/sqr(root) - 1) < epsilon;

   >> UNTIL          until
   >> <IDENTIFIER>    abs
   >> (              (
   >> <IDENTIFIER>    number
   >> /              /
   >> <IDENTIFIER>    sqr
   >> (              (
   >> <IDENTIFIER>    root
   >> )              )
   >> -              -
   >> <NUMBER>        1
   >> )              )
   >> <              <
   >> <IDENTIFIER>    epsilon
   >> ;              ;
 33:     END

   >> END            end
 34:     UNTIL number = 0

   >> UNTIL          until
   >> <IDENTIFIER>    number
   >> =              =
   >> <NUMBER>        0
 35: END.

   >> END            end
   >> .              .
```

Actual program output (ActualOutput.txt):

```
Page    1  NEWTON.PAS  Sun Mar 16 00:49:11 2014

   1: PROGRAM newton (input, output);
>> PROGRAM      program
>> <IDENTIFIER>           newton
>> (         (
>> <IDENTIFIER>           input
>> ,         ,
>> <IDENTIFIER>           output
>> )         )
>> ;         ;
   2:
```

```
  3: CONST
>> CONST          const
  4:    epsilon = 1e-6;
>> <IDENTIFIER>          epsilon
>> =        =
>> <NUMBER>    1e-6
>> ;        ;
  5:

  6: VAR
>> VAR    var
  7:    number, root, sqroot : real;
>> <IDENTIFIER>          number
>> ,        ,
>> <IDENTIFIER>          root
>> ,        ,
>> <IDENTIFIER>          sqroot
>> :        :
>> <IDENTIFIER>          real
>> ;        ;
  8:

  9: BEGIN
>> BEGIN          begin
 10:    REPEAT
>> REPEAT         repeat
 11:    writeln;
>> <IDENTIFIER>          writeln
>> ;        ;
 12:    write('Enter new number (0 to quit): ');
>> <IDENTIFIER>          write
>> (        (
>> <STRING>      Enter new number (0 to quit):
>> )        )
>> ;        ;
 13:    read(number);
>> <IDENTIFIER>          read
>> (        (
>> <IDENTIFIER>          number

Page    2  NEWTON.PAS  Sun Mar 16 00:49:11 2014

>> )        )
>> ;        ;
 14:
```

```
 15:    IF number = 0 THEN BEGIN
>> IF       if
>> <IDENTIFIER>          number
>> =       =
>> <NUMBER>   0
>> THEN          then
>> BEGIN         begin
 16:       writeln(number:12:6, 0.0:12:6);
>> <IDENTIFIER>          writeln
>> (       (
>> <IDENTIFIER>          number
>> :       :
>> <NUMBER>   12
>> :       :
>> <NUMBER>   6
>> ,       ,
>> <NUMBER>   0.0
>> :       :
>> <NUMBER>   12
>> :       :
>> <NUMBER>   6
>> )       )
>> ;       ;
 17:    END
>> END    end
 18:    ELSE IF number < 0 THEN BEGIN
>> ELSE          else
>> IF       if
>> <IDENTIFIER>          number
>> <       <
>> <NUMBER>   0
>> THEN          then
>> BEGIN         begin
 19:       writeln('*** ERROR:  number < 0');
>> <IDENTIFIER>          writeln
>> (       (
>> <STRING>      *** ERROR:  number < 0
>> )       )
>> ;       ;
 20:    END
>> END    end
 21:    ELSE BEGIN
>> ELSE          else
>> BEGIN         begin
 22:       sqroot := sqrt(number);
```

```
>> <IDENTIFIER>          sqroot

Page    3  NEWTON.PAS  Sun Mar 16 00:49:11 2014

>> :        :
>> =        =
>> <IDENTIFIER>          sqrt
>> (        (
>> <IDENTIFIER>          number
>> )        )
>> ;        ;
  23:       writeln(number:12:6, sqroot:12:6);
>> <IDENTIFIER>          writeln
>> (        (
>> <IDENTIFIER>          number
>> :        :
>> <NUMBER>    12
>> :        :
>> <NUMBER>    6
>> ,        ,
>> <IDENTIFIER>          sqroot
>> :        :
>> <NUMBER>    12
>> :        :
>> <NUMBER>    6
>> )        )
>> ;        ;
  24:       writeln;
>> <IDENTIFIER>          writeln
>> ;        ;
  25:

  26:       root := 1;
>> <IDENTIFIER>          root
>> :        :
>> =        =
>> <NUMBER>    1
>> ;        ;
  27:       REPEAT
>> REPEAT         repeat
  28:               root := (number/root + root)/2;
>> <IDENTIFIER>          root
>> :        :
>> =        =
>> (        (
>> <IDENTIFIER>          number
```

```
>> /        /
>> <IDENTIFIER>          root
>> +        +
>> <IDENTIFIER>          root
>> )        )
>> /        /
>> <NUMBER>    2
>> ;        ;

Page    4  NEWTON.PAS  Sun Mar 16 00:49:11 2014

  29:               writeln(root:24:6,
>> <IDENTIFIER>          writeln
>> (        (
>> <IDENTIFIER>          root
>> :        :
>> <NUMBER>    24
>> :        :
>> <NUMBER>    6
>> ,        ,
  30:                       100*abs(root - sqroot)/sqroot:12:2,
>> <NUMBER>    100*abs(root
>> -        -
>> <IDENTIFIER>          sqroot
>> )        )
>> /        /
>> <IDENTIFIER>          sqroot
>> :        :
>> <NUMBER>    12
>> :        :
>> <NUMBER>    2
>> ,        ,
  31:                       '%')
>> <STRING>      %
>> )        )
  32:       UNTIL abs(number/sqr(root) - 1) < epsilon;
>> UNTIL          until
>> <IDENTIFIER>          abs
>> (        (
>> <IDENTIFIER>          number
>> /        /
>> <IDENTIFIER>          sqr
>> (        (
>> <IDENTIFIER>          root
>> )        )
>> -        -
```

```
>> <NUMBER>    1
>> )         )
>> <         <
>> <IDENTIFIER>            epsilon
>> ;         ;
 33:     END
>> END     end
 34:     UNTIL number = 0
>> UNTIL             until
>> <IDENTIFIER>            number
>> =         =
>> <NUMBER>    0

>> )         )
>> -         -
>> <NUMBER>    1

Page    5  NEWTON.PAS  Sun Mar 16 00:49:11 2014

>> )         )
>> <         <
>> <IDENTIFIER>            epsilon
>> ;         ;
 35: END.
>> END     end
>> .         .
```