```
PROGRAM travesty (input, output);

CONST
   ArraySize = 3000;
   MaxPat = 9;

VAR
  BigArray : PACKED ARRAY [1..ArraySize] of CHAR;
  FreqArray, StartSkip : ARRAY[''..'|'] of INTEGER;
  Pattern : PACKED ARRAY [1..MaxPat] of CHAR;
  SkipArray : ARRAY [1..ArraySize] of INTEGER;
  OutChars : INTEGER;
  PatLength : INTEGER;
  f : TEXT;
  CharCount : INTEGER;
  Verse, NearEnd : BOOLEAN;
  NewChar : CHAR;
  TotalChars : INTEGER;
  Seed : INTEGER;

FUNCTION Random (VAR RandInt : INTEGER) : REAL;
BEGIN
  Random := RandInt / 1009;
  RandInt := (31 * RandInt + 11) MOD 1009
END;

PROCEDURE InParams;
( * Obtains user's instructions * )
VAR
  InFile : STRING [12];
  Response : CHAR;
BEGIN
  WRITELN ('Enter a Seed (1..1000) for the randomizer');
  READLN (Seed);
  WRITELN ('Number of characters to be output?');
  READLN (OutChars);
  REPEAT
    WRITELN ('What order? <2-', MaxPat,'>');
    READLN (PatLength)
  UNTIL (PatLength IN [2..Maxpat]);
  PatLength := PatLength - 1;
  WRITELN ('Name of input file?');
  READLN (InFile);
  ASSIGN(f, InFile);
  RESET (f);
  WRITELN ('Prose or Verse? <p/v>');
  READLN (Response);
  IF (Response = 'V') OR (Response = 'v') THEN
     Verse := true
  ELSE Verse := false
END; {Procedure InParams}

PROCEDURE ClearFreq;
(* FreqArray is indexed by 93 probable ASCII characters,         *)
(* from "" to "|". Its elements are all set to zero.             *)
VAR
  ch : CHAR;
BEGIN
  FOR ch := '' TO '|' DO
   FreqArray[ch] := 0
END; {Procedure ClearFreq}

PROCEDURE NullArrays;
(*  Fill BigArray and Pattern with nulls   *)
VAR
 j : INTEGER;
BEGIN
  FOR j:= 1 TO ArraySize DO
    BigArray[j] := CHR(0);
  FOR j := 1 TO MaxPat DO
    Pattern[j] := CHR(0)
END; {Procedure NullArrays}

PROCEDURE FillArray;
(*  Moves textfile from disk into BigArray, cleaning it         *)
(*  up and reducing any run of blanks to one blank.             *)
(*  Then copies to end of array a string of its opening         *)
(*  characters as long as the Pattern, in effect wrapping       *)
(*  the end to the beginning.                                   *)
VAR
  Blank : BOOLEAN;
  ch: CHAR;
  j : INTEGER;

  PROCEDURE Cleanup;
  (* Clears Carriage Returns, Linefeeds, and Tabs out of        *)
  (* input stream. All are changed to blanks.                   *)
BEGIN
 IF (( ch = CHR(13))      {CR}
     OR (ch = CHR(10))    {LF}
     OR (ch = CHR(9))     {TAB}
 THEN ch := ''
END;

BEGIN {Procedure FillArray}
 j := 1;
 Blank := false;
 WHILE (NOT EOF(f)) AND (j <= (ArraySize-MaxPat)) DO
 BEGIN {While not EOF}
  READ (f, ch);
  Cleanup;
  BigArray[j] := ch;               {Place character in BigArray}
  IF ch = '' THEN Blank := true;
  j := j + 1;
  WHILE (Blank AND (NOT EOF(f))
    AND (j <= (ArraySize-MaxPat))) DO
  BEGIN {While Blank}              {When a blank has just been}
    READ (f, ch);                  {printed, Blank is true,}
    Cleanup;                       {so succeeding blanks are skipped,}
    IF ch <> '' THEN               {thus stopping runs.}
    BEGIN {If}
      Blank := false;
      BigArray[j] := ch;
      j := j + 1
    END {If}
  END {While Blank}
END; {While Not EOF}
TotalChars := j - 1;
IF BidArray[TotalChars] <> '' THEN
BEGIN                              {If no Blank at end of text, append one}
  TotalChars := TotalChars + 1;
  BigArray[TotalChars] := ''
END;
{Copy front of array to back to simulate wraparound.}
FOR j := 1 TO PatLength DO
  BigArray[TotalChars+j] := BigArray[j];
 TotalChars := TotalChars + PatLength;
 WRITELN('Characters read, plus wraparound = ',TotalChars:4)
END; {Procedure FillArray}

PROCEDURE FirstPattern;
(*   User selects "order" of operation, an integer, n, in the *)
(*   range 1..9. The input text will henceforth be scanned *)
(*   in n-sized chunks. The first n-1 characters of the input *)
(*   file are placed in the "Pattern" Array. The Pattern is *)
(*   written at the head of output. *)
VAR
  j:INTEGER;
BEGIN
  FOR j := 1 TO PatLength DO                {Put opening chars into Pattern}
    Pattern[j] := BigArray[j];
  CharCount := PatLength;
  NearEnd := false;
  IF Verse THEN ('');                       {Align first line}
  FOR j := 1 TO PatLength DO
   WRITE (Pattern[j])
  END; {Procedure FirstPattern}

PROCEDURE InitSkip;
VAR
  ch : CHAR;
  j : INTEGER;
BEGIN
 FOR ch := '' TO '|' DO
```

```
  StartSkip[ch] := TotalChars + 1;
 FOR j := TotalChars DOWNTO 1 DO
 BEGIN
  ch := BigArray[j];
  SkipArray[j] := StartSkip[ch];
  StartSkip[ch] := j
  END
END;  {Procedure InitSkip}

PROCEDURE Match;
VAR
  i : INTEGER;
  j : INTEGER;
  Found : BOOLEAN;
  ch1 : CHAR;
  NxtCh : CHAR;
BEGIN
  ch1 := Pattern[1];
  i := StartSkip[ch1] - 1;
  WHILE (i <= TotalChars-PatLength-1) DO
  BEGIN
   j := 1;
   Found := true;
   WHILE (Found AND (j <= PatLength)) DO
     IF BigArray[i+j] <> Pattern[j]
       THEN Found:= false
       ELSE j := j + 1;
  IF Found THEN
  BEGIN
    NxtCh := BigArray[i + PatLength +1];
    FreqArray[NxtCh] := FreqArray[NxtCh] +1
  END;
  i := SkipArray[i+1] - 1
 END
END;

PROCEDURE WriteCharacter;
VAR
   Counter, Total, Toss : INTEGER;
   ch : CHAR;
BEGIN
  Total := 0;
  FOR ch := '' TO '|' DO
  Total := Total + FreqArray[ch];
  Toss := TRUNC (Total * Random(Seed)) +1;
  Counter := 31;
  REPEAT
    Counter := Counter +1;
    Toss := Toss-FreqArray[CHR(Counter)]
    until Toss <= 0;
  NewChar := CHR(Counter);
  IF NewChar <> '|' THEN
    Write (NewChar);
  CharCount := CharCount +1;
  IF CharCount MOD 50 = 0 THEN NearEnd := true;
  IF ((Verse) AND (NewChar = '|')) THEN WRITELN;
  IF ((NearEnd) AND (NewChar = '')) THEN
  BEGIN
   WRITELN;
   IF Verse THEN WRITE ('   ');
   NearEnd := false
  END
END;

PROCEDURE NewPattern;
VAR
  j : INTEGER;
BEGIN
  FOR j := 1 to PatLength - 1 DO
    Pattern[j] := Pattern[j+1];
  Pattern[PatLength] := NewChar;
  ClearFreq
END;

BEGIN
  ClearFreq;
  NullArrays;
  InParams;
  FillArray;
  FirstPattern;
  InitSkip;
  REPEAT
    Match;
    WriteCharacter;
    NewPattern
  UNTIL CharCount >= OutChars;
END. {Main Program}
```