Aufgabe 3:

Zeitaufwand: 1h

KMP pattern match algorithmus verwenden um pattern zuerst vorzubearbeiten und danach zeichen für zeichen die eingabe durchgehen. Somit ist es egal wenn ein pattern nicht match und man nicht zurückspringen muss zu i – pLen sondern einfach zeichen für zeichen durchiteriert. Falls das pattern nicht gefunden wird aber man trotzdem das program beenden kann hab ich mich für ein terminate char entschieden, in dem Fall !.

Code:

program PatternSearch;

var

  pattern: String;

  pLen, i, j, pos: Integer;

  next: array of integer;

  c: Char;

begin

  Write('Enter pattern: ');

  ReadLn(pattern);

  pLen := Length(pattern);

  SetLength(next, pLen + 1);

  i := 1;

  j := 0;

  next[i] := 0;

  while(i < pLen) do

    if((j = 0) or (pattern[i] = pattern[j])) then

    begin

      Inc(i);

      Inc(j);

      if (pattern[i] <> pattern[j]) then next[i] := j else next[i] := next[j];

    end else j := next[j];

  i := 1;

  pos := 1;

  write('char > '); readln(c);

  while ((i <= pLen) and (c <> '!')) do // my terminate char is a !

    if((i = 0) or (c = pattern[i])) then

    begin

      Inc(pos);

      write('char > '); readln(c);

      Inc(i);

    end else i := next[i];

  WriteLn;

  if (i > pLen) then WriteLn('pattern was found on position: ', pos - pLen) else WriteLn('not found');

  WriteLn;

end.

Tests:

Ein Bild, das Text enthält.

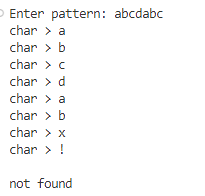
Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung



Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung