Tapaus 8

Tarkastellaan seuraavaa yksinkertaista ohjelmointikieltä, jonka kaikki ohjemat ovat muotoa *begin LAUSE; end.* *LAUSE* voi olla yksinkertainen sijoitus $MUUTTUJA := MUUTTUJA$;, $MUUTTUJA := KLUKU$;, $MUUTTUJA := MUUTTUJA+KLUKU$; tai $MUUTTUJA := MUUTTUJA-KLUKU$;, missä muuttujat ovat kirjaimia $a, b, ..., z$ ja $KLUKU$ vastaa etumerkitöntä kokonaislukua. Tai *LAUSE* on while-rakenne: *while (EHTO) do begin LAUSE end;*. *EHTO* puolestaan vertailee kahta tekijää, jotka voivat olla muuttujia tai kokonaislukuja, operaatioilla $<, >, =, <=, >=, <>$.

Muodosta ohjelmointikielen kuvaava kontekstiton kielioppi ja laadi sille (pseudokoodilla) rekursiivinen jäsentäjä, joka tutkii, onko annettu koodi syntaksin mukainen. Esim. allaoleva ohjelma on syntaksiltaan laillinen. Huomaa, että sisäkkäisiä while-silmukoita saa olla kuinka paljon tahansa!

Voisitko laatia samaan tapaan toimivan jäsenytimen Pascal- tai C-kielille?

Let's consider the following simple programming language, in which all programs are of form *begin STATEMENT; end.* *STATEMENT* can be a simple assignment $VARIABLE := VARIABLE$;, $VARIABLE := INT$;, $VARIABLE := VARIABLE + INT$; or $VARIABLE := VARIABLE - INT$;, in which variables are letters $a, b, ..., z$ and INT corresponds an unsigned integer. Or *STATEMENT* is a while structure: *while (CONDITION) do begin STATEMENT end;*. *CONDITION* compares two factors, which can be either variables or integers, with operations $<, >, =, <=, >=, <>$.

Construct a context-free grammar, which describes the language and give it a recursive (pseudocode) parser, which checks whether a given program code is syntactically correct. E.g. the program below has a correct syntax. Notice that there can be any number of nested while-loops!

Could you construct a parser for Pascal or C language in the same way?

```
begin
    a:=1; b:=10;
    while (a<b) do
    begin
        a:=a+1;
        b:=b-1;
    end;
    c:=a+b;
end;
```