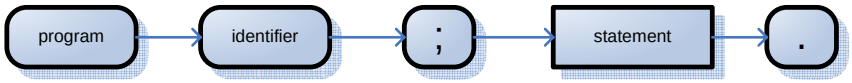
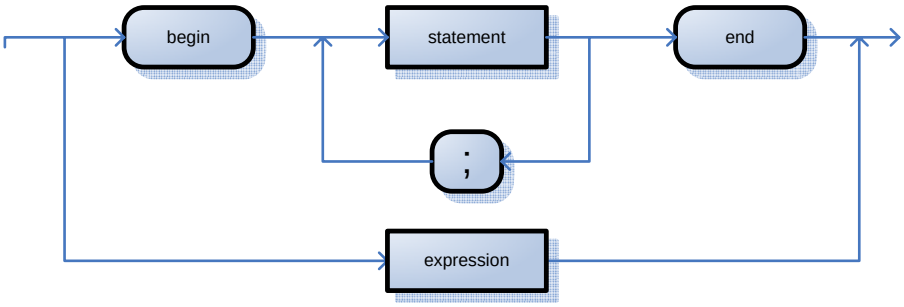
Buatlah parser untuk syntax graph seperti yang terdeskripsikan dibawah ini !

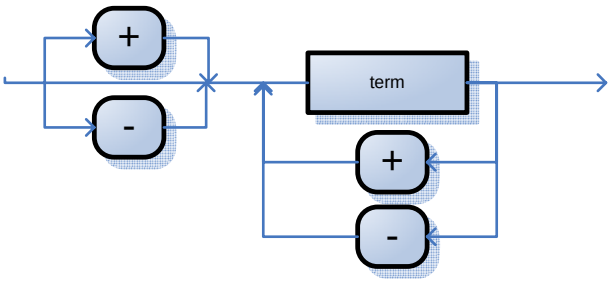
program :



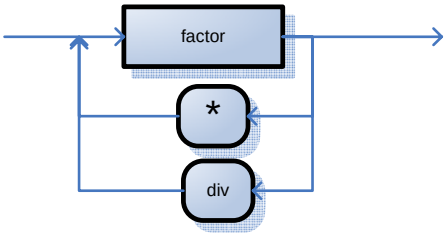
statement :



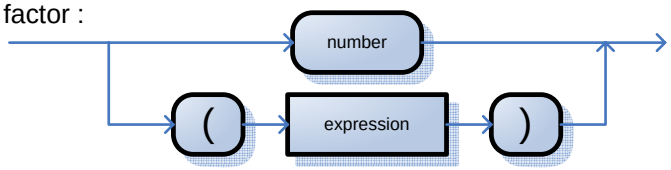
expression :



term :



factor :



Testing Code :

Input Source
`Program example41;` `  321.`

`Program example42;`
`Begin`
`  376 + 4 * 6;`
`  (4+2)*8;`
`  10 * 32`
`End.`

`Program example43;`
`Begin`
`  -(1+(-2))`
`End.`

Lampiran :

```
/*   Pseudo code for expression parsing */

#define isplus     (token.attr == SYMBOL && token.value == PLUS)
#define isminus    (token.attr == SYMBOL && token.value == MINUS)
#define istimes    (token.attr == SYMBOL && token.value == TIMES)
#define isdiv      (token.attr == RWORD  && token.value == DIV)


void expression(void)
{
   if (isplus || isminus) get_token();

   term();
   while (isplus || isminus) {
     get_token();
     term();
}

void term(void)
{
    factor();
    while (istimes || isdiv) {
       get_token();
       factor();
    }
}

void factor(void)
{
    if (isnumber) {
       get_token();
       return;
    } else if (islparen) {
       get_token();
       expression();
       if (isrparen) {
          get_token();
          return;
       } else error();
    } else error();
}
```