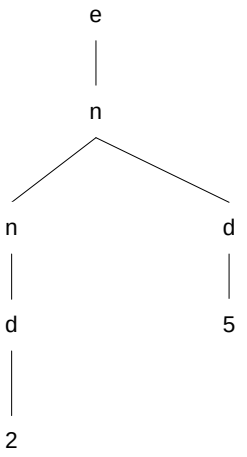
INF 3110

Solutions **7.9 - 11.9, 2009**

Problem 1

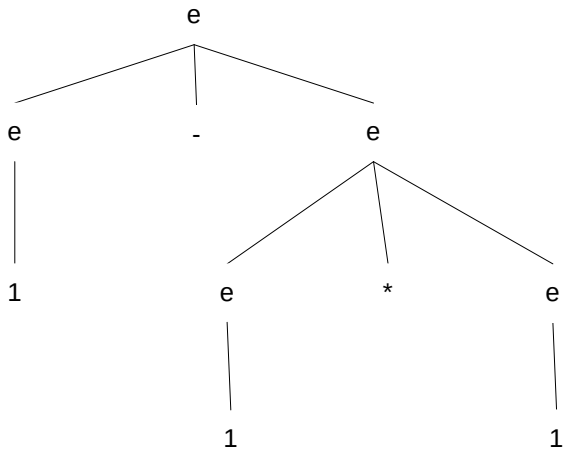
Mitchell 4.1



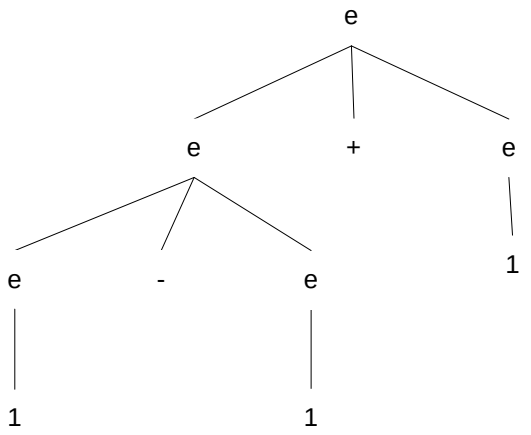
Another derivation: e -> n -> nd -> n5 -> d5 -> 25

Mitchell 4.2

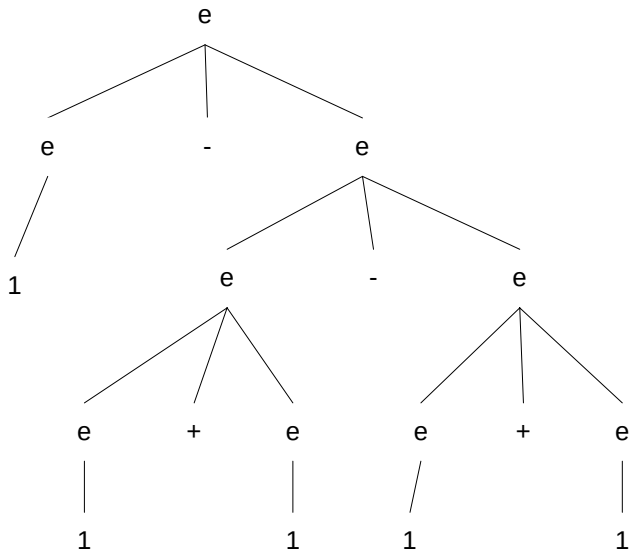
(a)



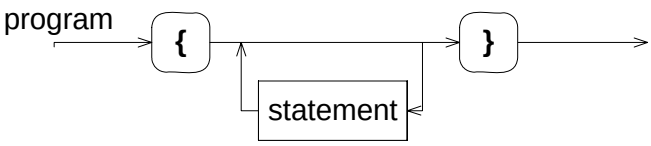
(b)

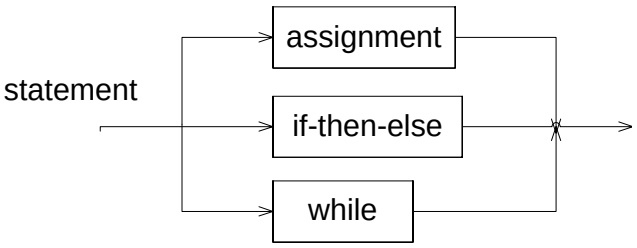


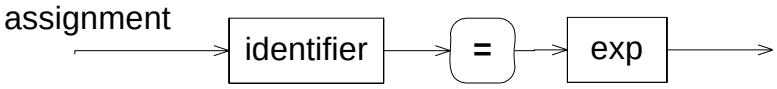
(c)

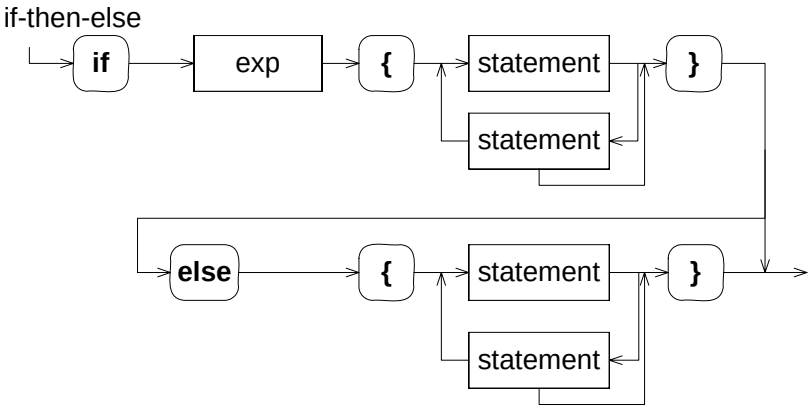


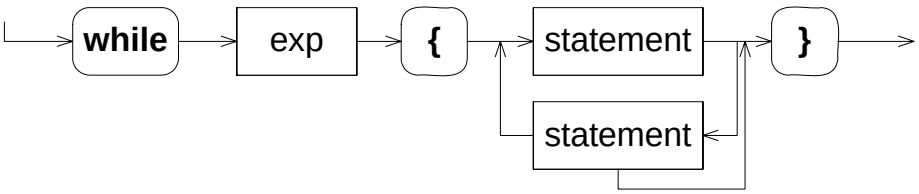
Problem 2

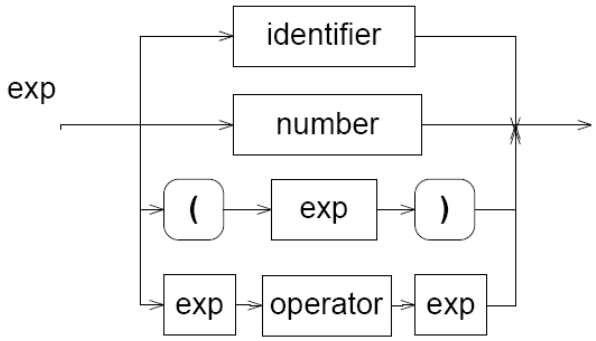

program
{
}
statement


statement
assignment
if-then-else
while


assignment
identifier
=
exp

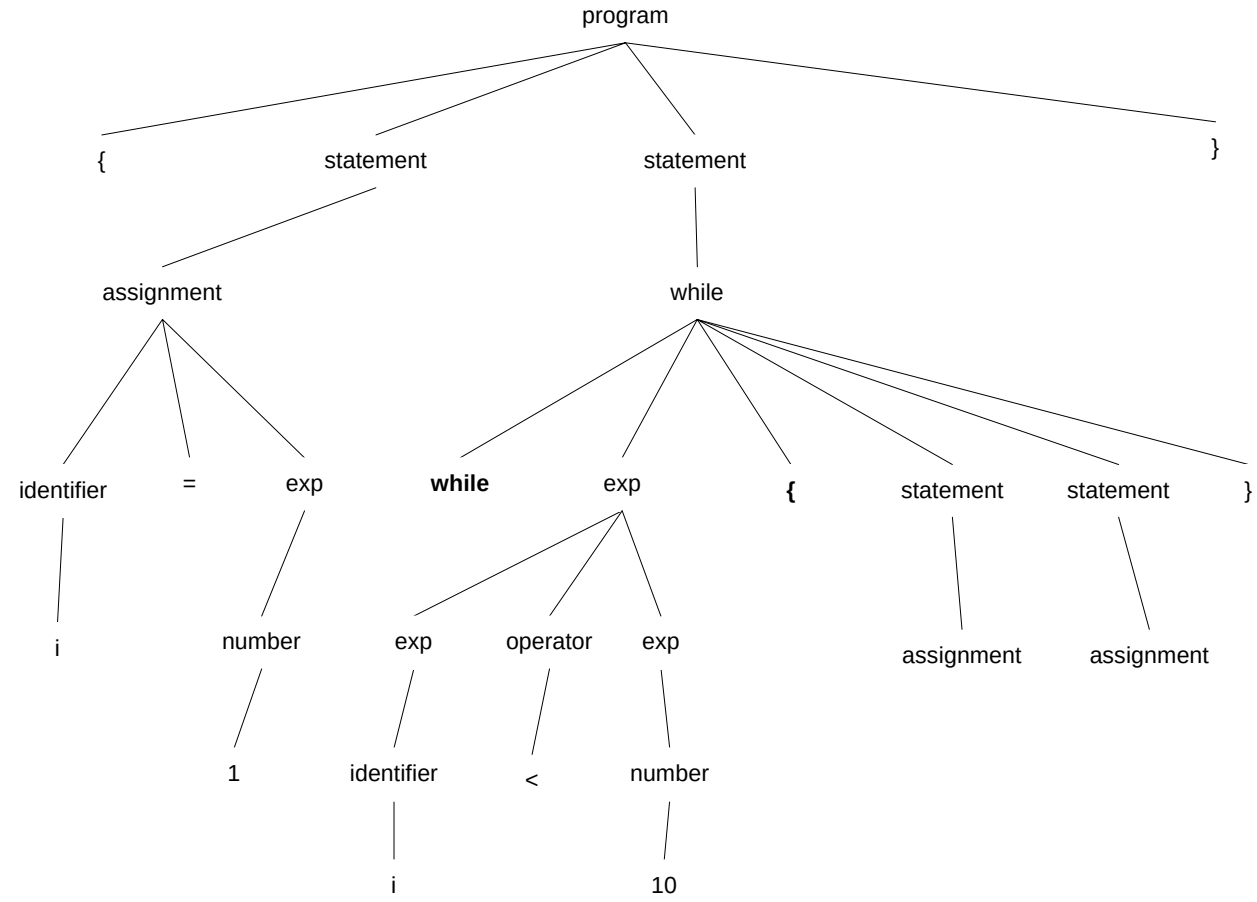

if-then-else
if
exp
{
statement
statement
}
else
{
statement
statement
}


while
exp
{
statement
statement
}

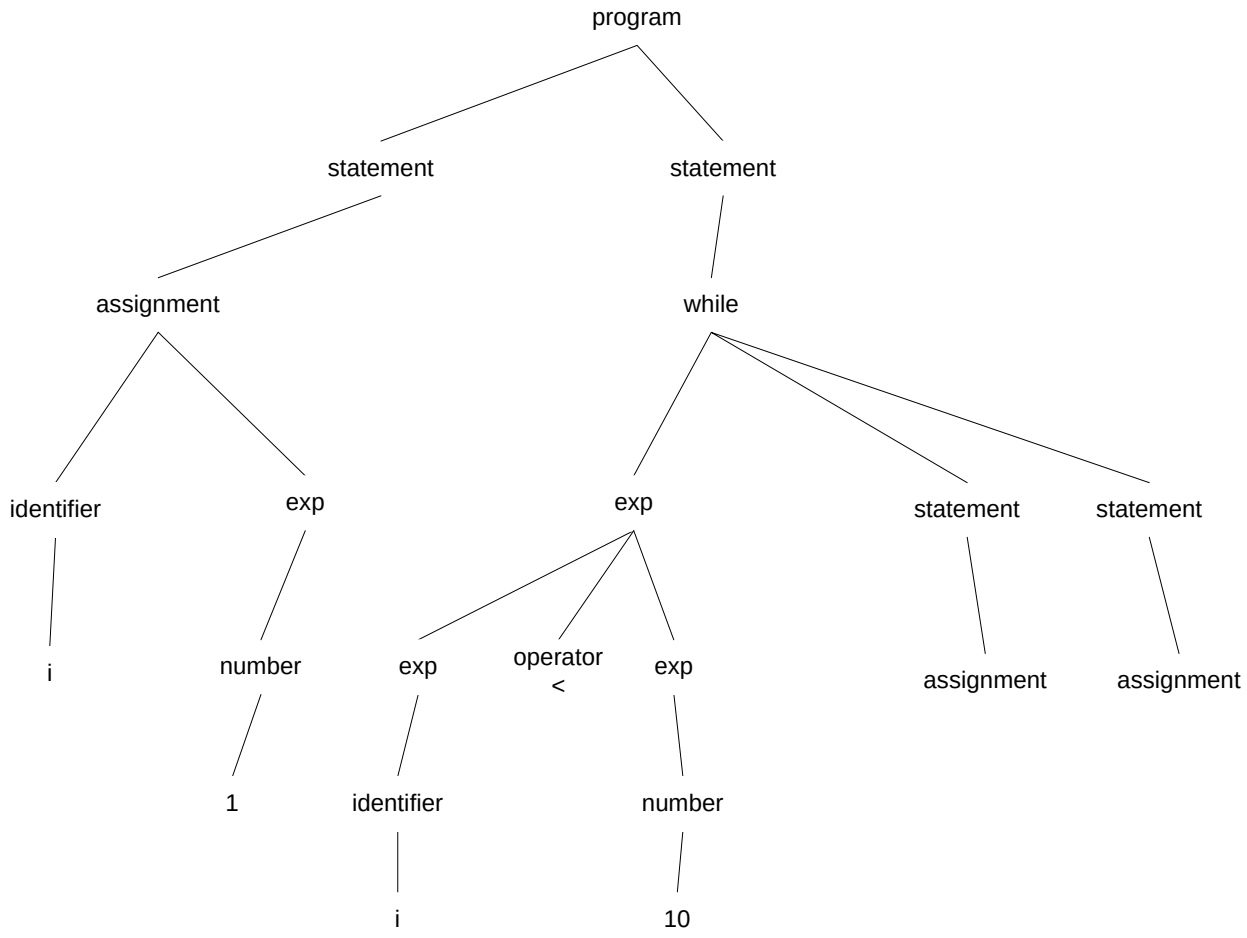


Problem 3

Parse tree

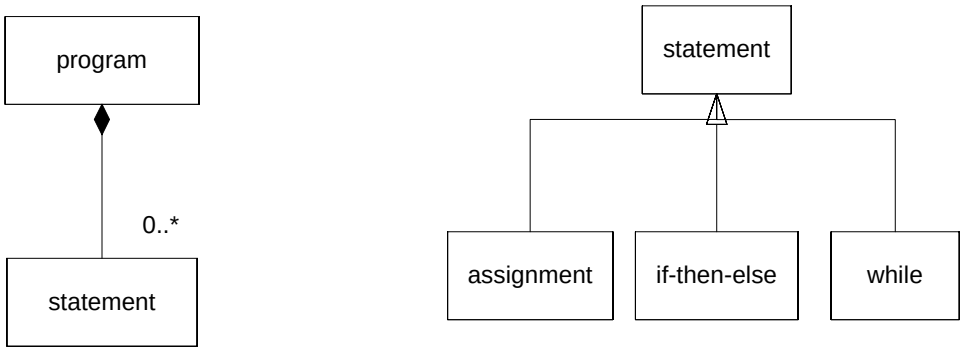


Abstract syntax tree

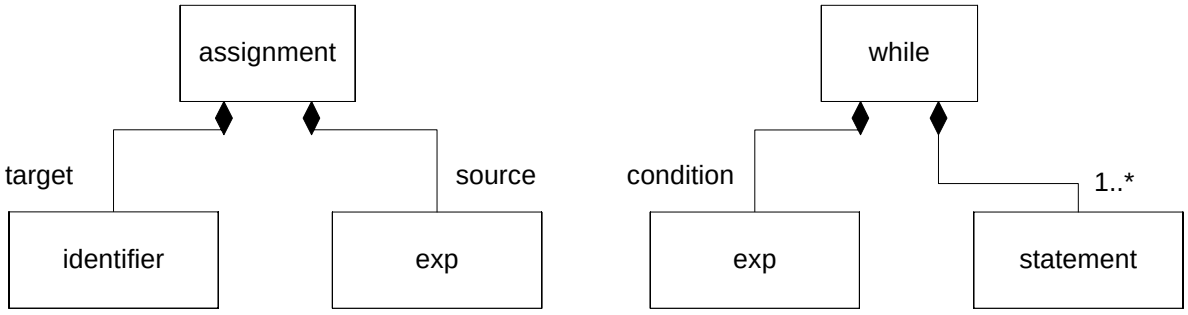


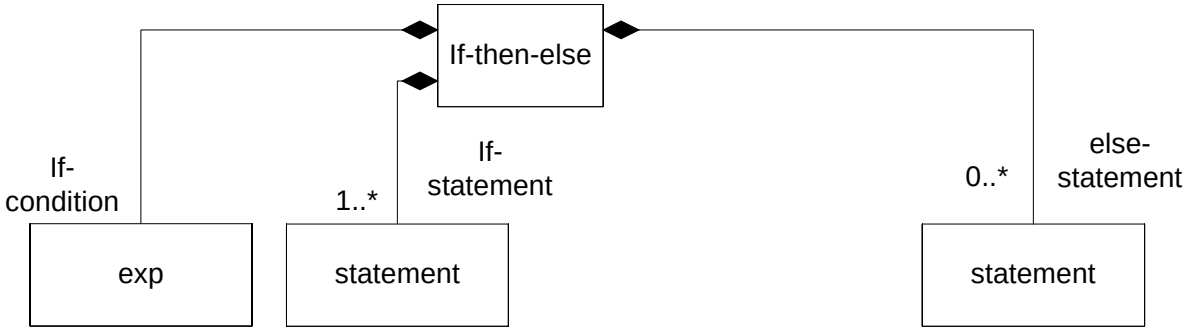
Problem 4

a) Metamodel corresponding to the grammar in Problem 2.

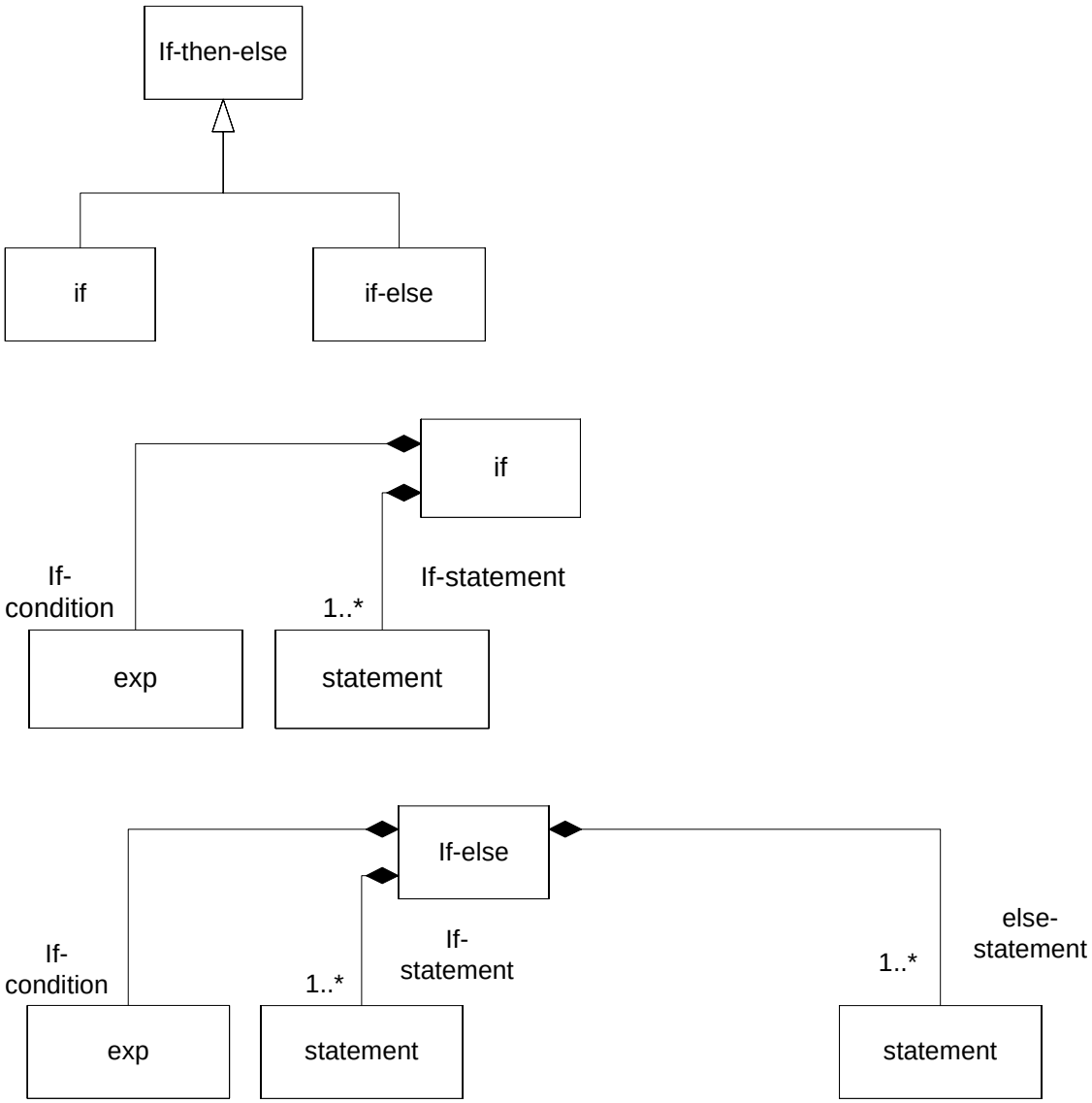


To be more precise we should really model the statements as a list of statements and not just as a set of statements, so in this metamodel we assume that compositions are made by means of lists.





Alternative if-then-else:

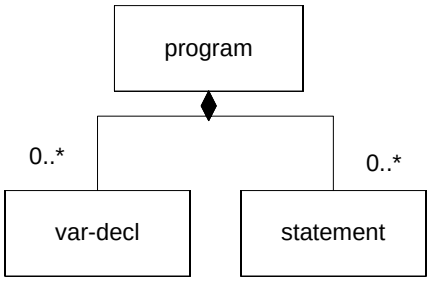


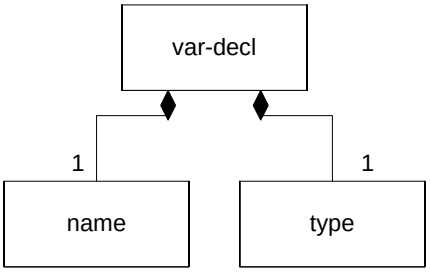
b)

The grammar is changed like this:

```
<program> ::= { <var-decl>* <statement>* }
<var-decl> ::= <name> <type>
```

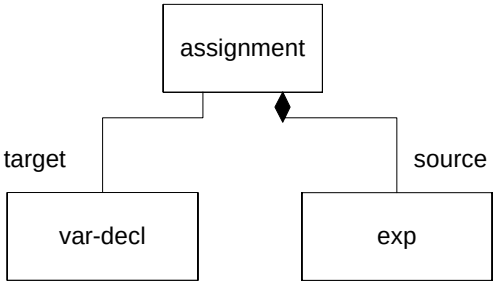
The metamodel is changed accordingly, by adding var-decl to program and define var-decl to have a name and a type:





We assume that all types will be modeled by subclasses of class ‘type’.

We also have to change the metamodel in places where we have ‘identifier’ so that it now links to the corresponding var-decl, e.g.:

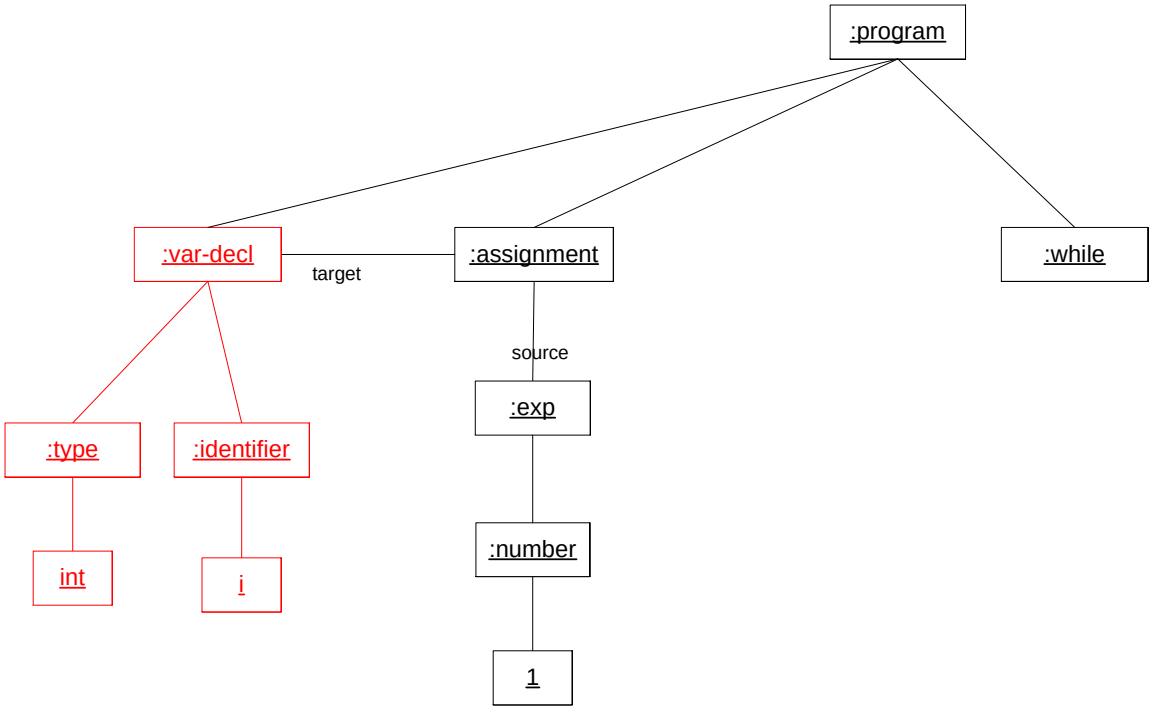


That is, an assignment consists of a link to the target variable and an expression.

Here is an example with a declaration of variable ‘i’:

```
{ int i;
  i=1
  while i<10 {
     i=i+1    j=j+i
  }
}
```

Here is part of the corresponding object model representing the program according to the metamodel above:


:program
:var-decl
target
:assignment
:while
source
:exp
:type
:identifier
:number
int
i
1