Exercise 1: Symbol Table with Simple Types

Given the following program:

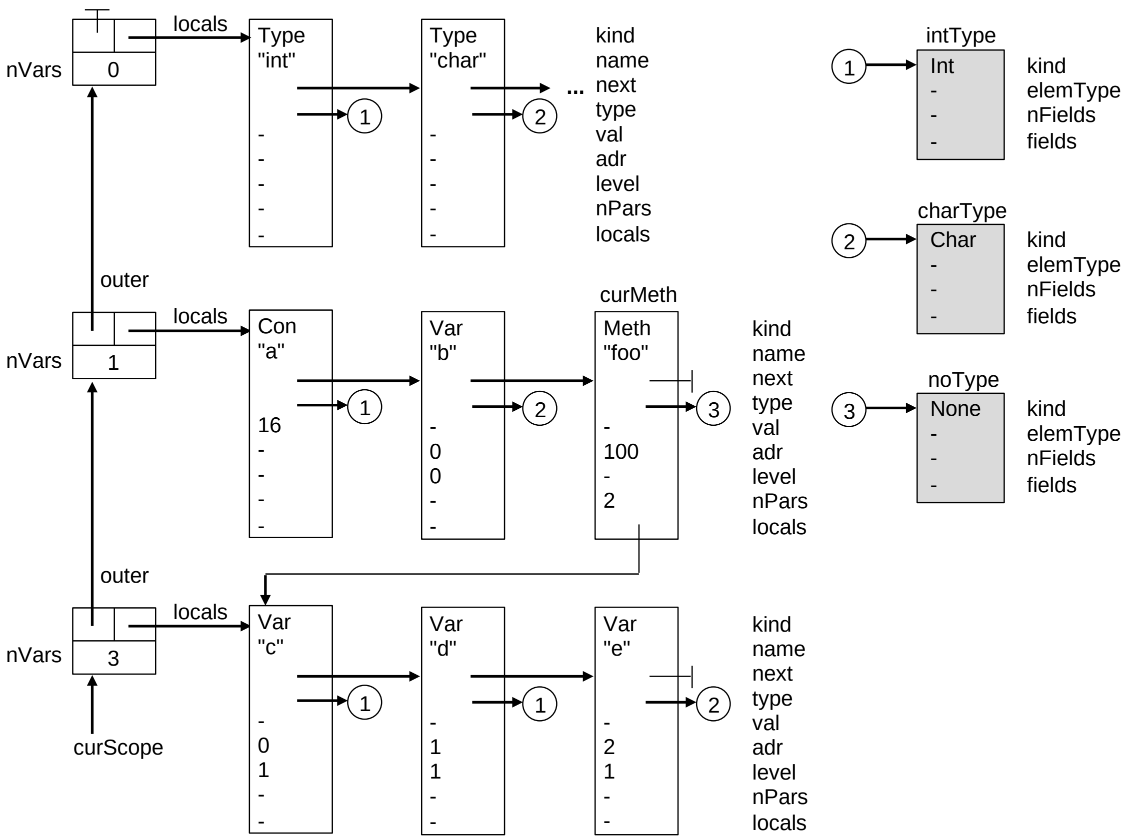
```
program P
   final int a = 16;
   char b;
{
   void foo (int c, int d)
     char e;
   { ...  // <== 1
   }

   void bar()
     int f, g;
   { ...  // <== 2
   }
}
```

a) Draw the symbol table (like in Fig. 5.7) at the point in time when the compiler is at position 1 (in foo()). Draw all scopes with their object and structure nodes. For the universe, it is sufficient to draw a few example nodes. Also specify the values of the fields in object and structure nodes.

b) Draw the symbol table at the point in time when the compiler is at position 2 (in bar()).

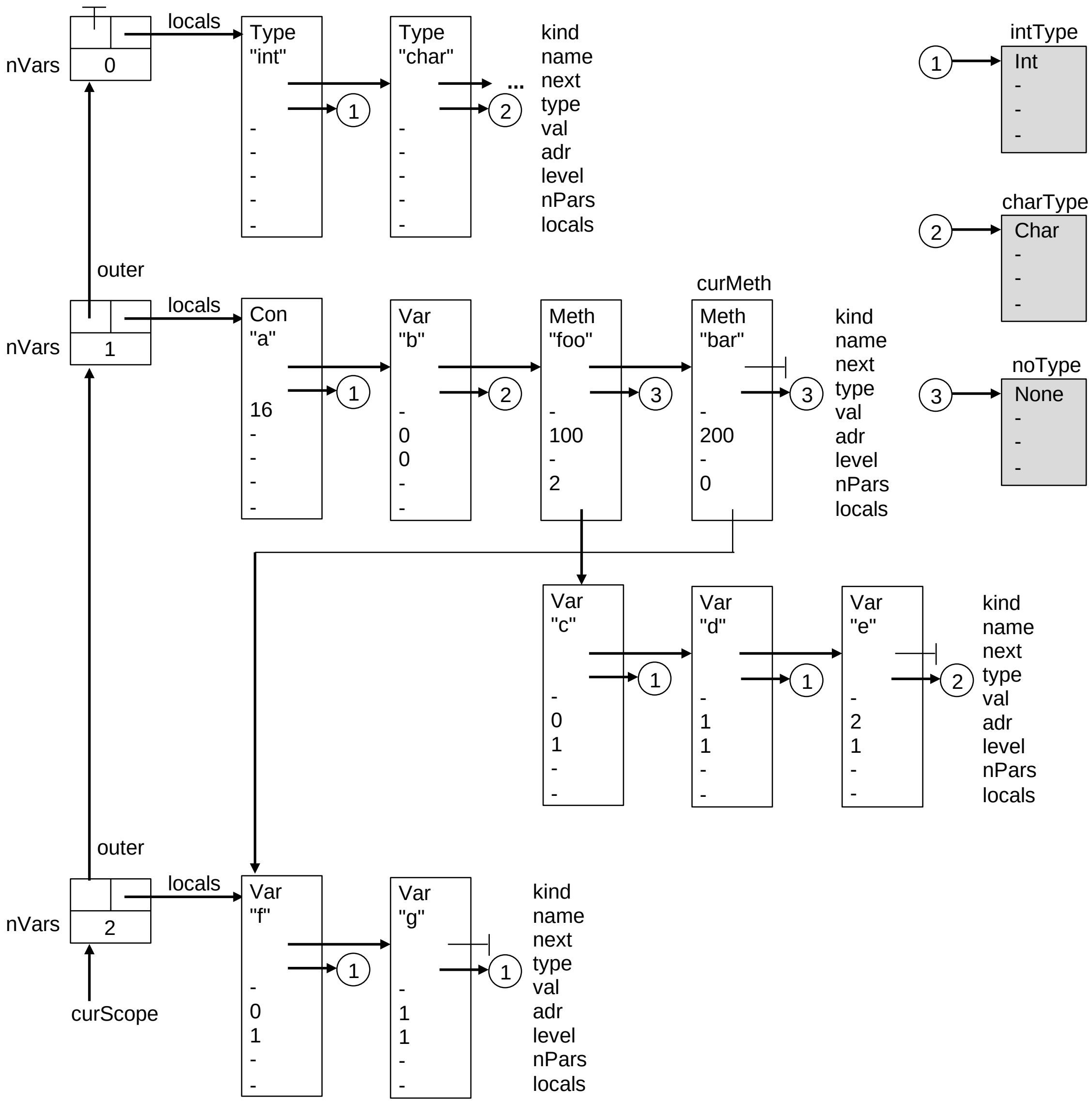
Solution

a) Symbol table at the position 1 (assume: foo() has the address 100 in the code buffer).



There is one scope each for the universe, for the program P and for the method foo() (object node curMeth). The variables of the program scope have level = 0, the variables of the method scope have level = 1. The addresses of the variables are assigned consecutively starting at 0.

b) Symbol table at the position 2 (assume: bar() has the address 200 in the code buffer).



The scope of foo() has been closed and a new scope for bar() has been opened. There is again a scope for the universe, one for the program P and one for the method bar() (object node curMethod). However, the local objects of the method foo() are still attached to the object node of foo().