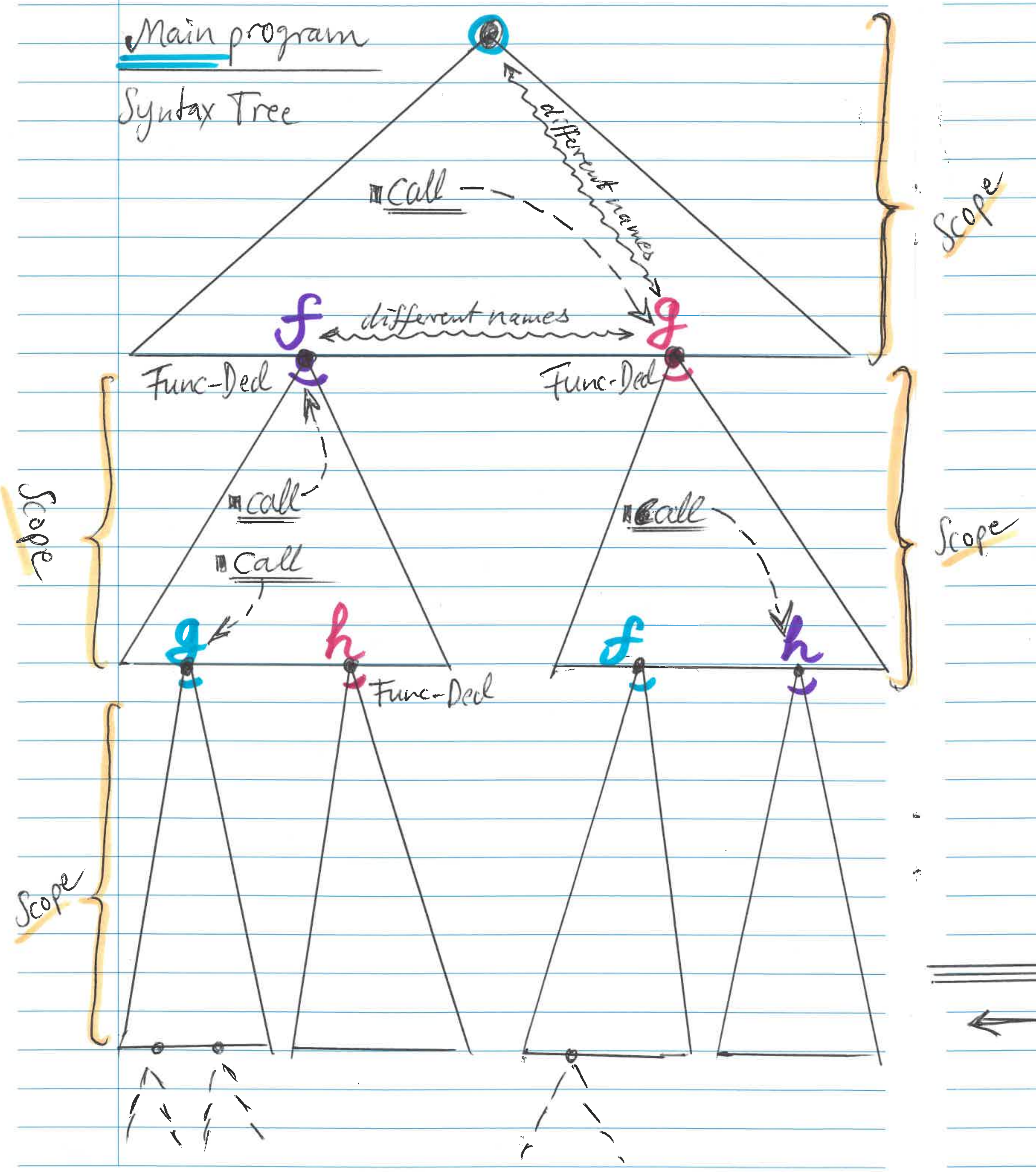
SEMESTER PROJECT : SEMANTIC



RULES CONCERNING FUNCTION NAMES

* The main program forms the highest-level scope, with no "parent".

* Every function declaration opens its own scope.

* A child scope may not have the same name as its immediate parent scope.

* A child scope may not have the same name as any of its sibling scopes under the same parent

* A call command may refer to an immediate Child-scope

* A call command may refer to its own scope : That is RECURSION

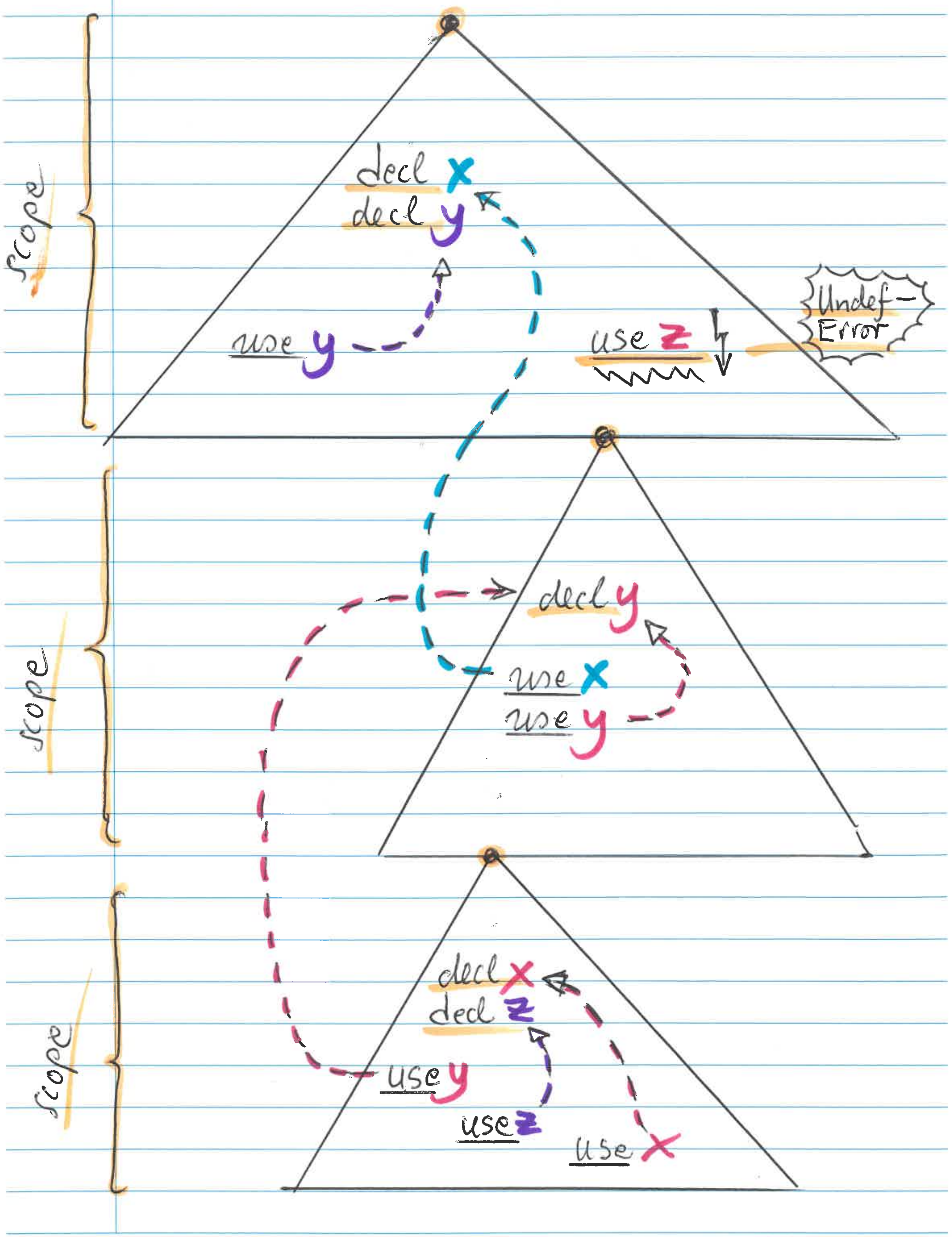
* There may be no recursive call to MAIN !

* Your Compiler's SEMANTIC ANALYSIS MODULE must throw an Error Report if any of the semantic rules of above are violated !

← in the figure on the left-side

- red g and blue g are different functions though they have the same NAME
- ditto for blue f, purple f, red h, purple h

★★★ SEMESTER PROJECT: SEMANTIC RULES



★★★ CONCERNING VARIABLE NAMES

- * No variable name may be double-declared (twice) in the same scope
  - for example: No string X and also number X
- * The declaration of a used variable name must be found either within that name's own scope, or in any higher ancestor scope.
- * If a used variable name has two declarations in two different scopes, then the "nearest" declaration is the relevant declaration for that variable.
- * Every used variable name must have a declaration.
- * No variable anywhere in the RecSPL program may have a name that is also used as a Function name anywhere in the program
- * No variable name anywhere in the RecSPL program may be identical with any Reserved Keyword
- * Two variables with the same name are different computational entities if they are rooted in different scopes
- * Your Compiler's SEMANTIC ANALYSIS MODULE must throw an Error Report (if) any of the semantic rules of above are violated