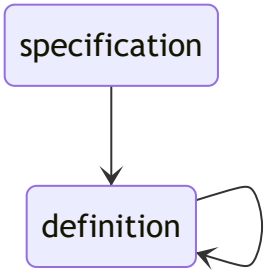
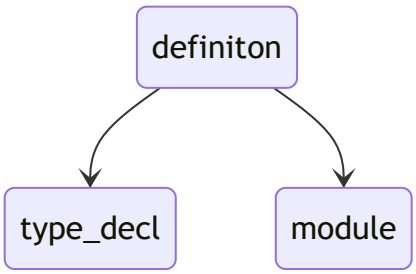
语法规则每一句对应的抽象语法树

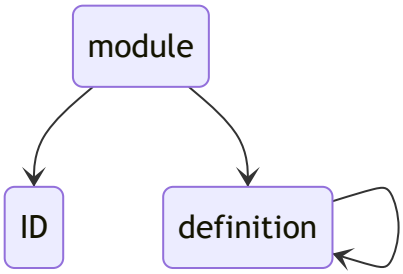
1.specification -> definition { definition }



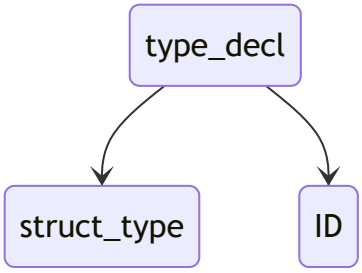
2.definiton -> type_decl “;”| module “;”



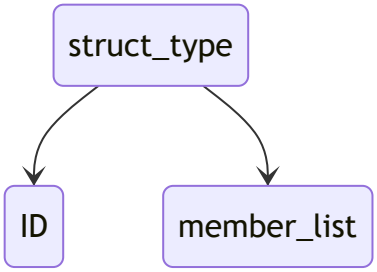
3.module -> “module”ID “{” definition { definition } “}”



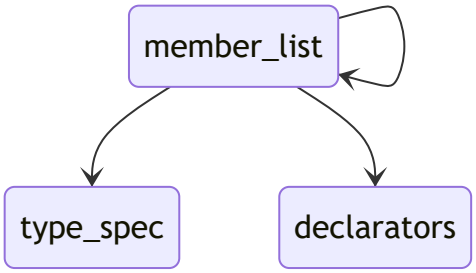
4.type_decl -> struct_type | “struct” ID



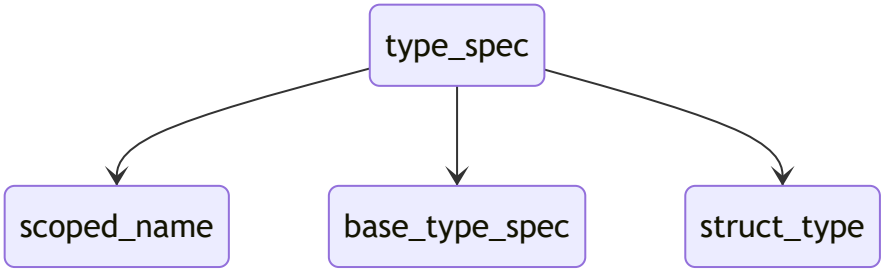
5.struct_type->“struct” ID “{” member_list “}”



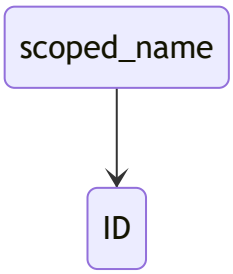
6.member_list-> { type_spec declarators “;” }



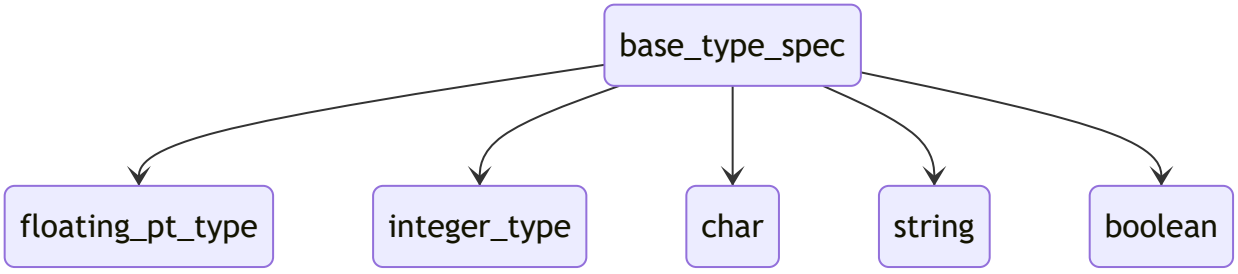
7.type_spec -> scoped_name | base_type_spec | struct_type



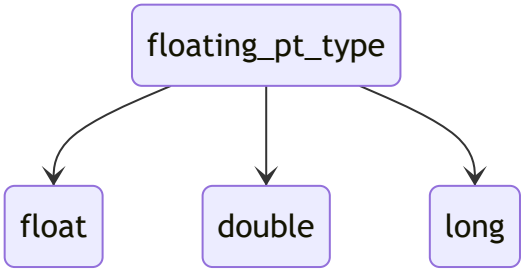
8.scoped_name -> [“::”] ID {“::” ID }



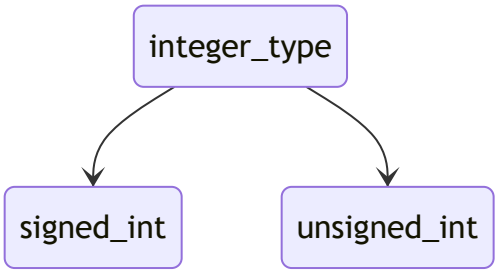
9.base_type_spec->floating_pt_type|integer_type|“char”|“string”|“boolean”



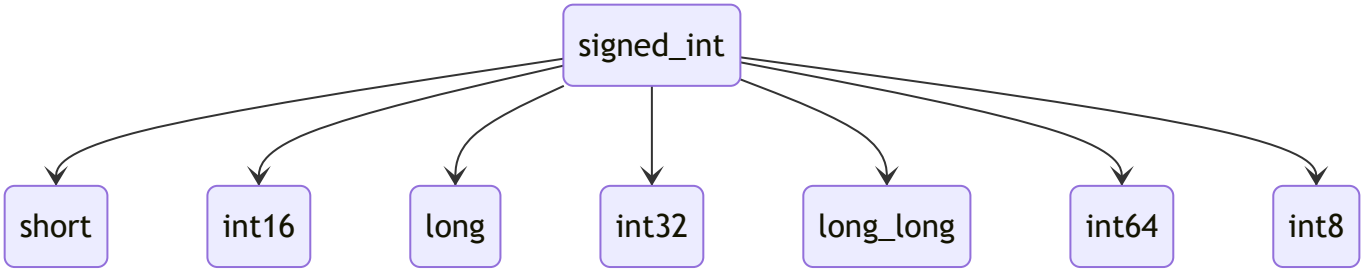
10.floating_pt_type -> “float” | “double” | “long double”



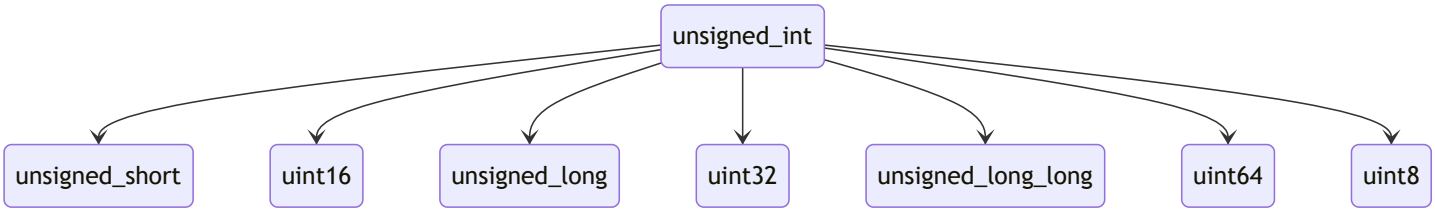
11.integer_type -> signed_int | unsigned_int



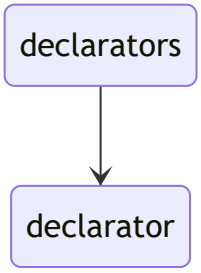
12.signed_int->(“short”|“int16”)|(“long”|“int32”)|(“long” “long”|“int64”)|“int8”



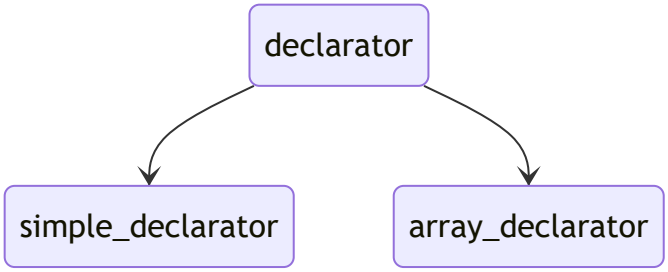
13.unsigned_int -> (“unsigned”“short”| “uint16”)| (“unsigned”“long”| “uint32”)| (“unsigned” “long” “long” | “uint64”)| “uint8”



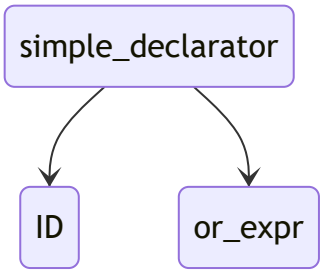
14.declarators -> declarator {“,” declarator }



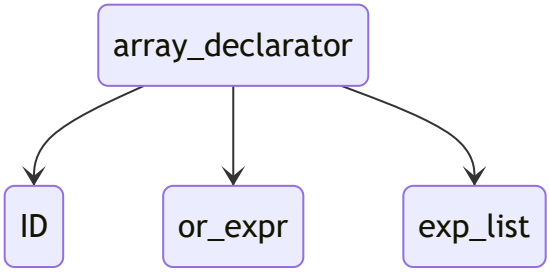
15.declarator -> simple_declarator | array_declarator



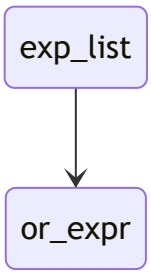
16.simple_declarator -> ID [“=” or_expr]



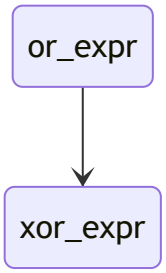
17.array_declarator -> ID “[” or_expr “]” [“=” exp_list]



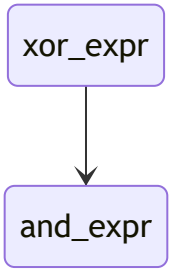
18.exp_list -> “[” or_expr { “,”or_expr } “]”



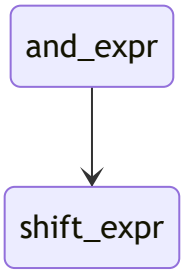
19.or_expr -> xor_expr {“|” xor_expr }



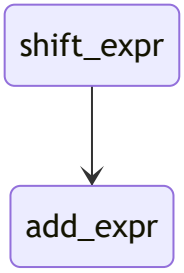
20.xor_expr -> and_expr {“^” and_expr }



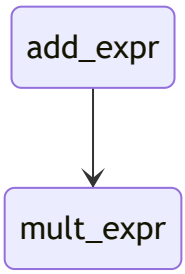
21.and_expr -> shift_expr {“&”shift_expr }



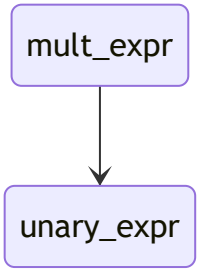
22.shift_expr -> add_expr { (“>>” | “<<”) add_expr }



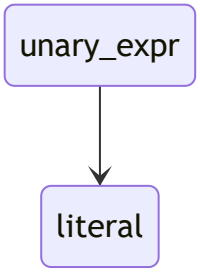
23.add_expr -> mult_expr { (“+” | “-”) mult_expr }



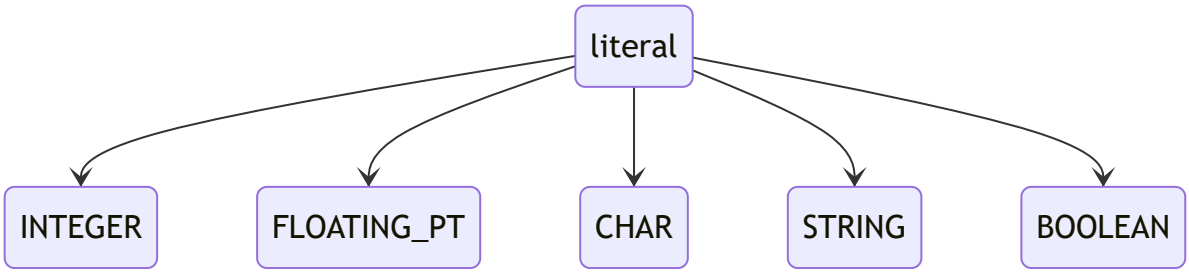
24.mult_expr -> unary_expr { (“*” |“/”|“%”) unary_expr }



25.unary_expr -> [“-”| “+” | “~”] literal



26.literal -> INTEGER | FLOATING_PT | CHAR | STRING | BOOLEAN



全部的整合