Compiler Project 4

compile process

Scanner->parser(build AST)->semantic Check->code generation

Semantic Check(run the AST) :

After semantic check, each node that contains a identifier has an entry to get information of the variable, and each factor/expression has its value type and array dimension(0 if its value is a constant,1 if its value is 1-dimensional array and so on)

1.Duplicate declaration(when add new variable)

2.Type mismatch(check when running to node_op/assign , check if valuetypes and arrdimensions are same in both sides/ constant int num is regarded as real in operation if there is a real num/var in operation)

3.Undeclared variable

4.Variable that hasn't been initialized

Global variable, parameters are set to be initialized in subprogram declaration part

If the variable is assigned successfully in assign statement, then it becomes initialized

Otherwise, an uninitialized variable can't be used as a factor

5.Procedure cannot be used as a factor

6.Function/procedure parameter construction check

7.The legality of Array use (after checking this ,the node will get array dimension of this factor/variable if there is [] follow behind it ,o.w ,arrdimension = 0)

8.function/procedure can't be LHS(can't be assigned) (check when runs to node_assign_statement)

**I set scopenum of loacal variables as -1 in symboltable after subprogram ends rather than delete them because I want to keep entry reference of each idNode until code generation done

Code generation :

I write another function called codegeneration to generate java byte code .

If there is no error in semantic check ,call codegeneration function.

it runs the AST that has been run in semantic check.

Therefore, we can know the information of each node and generate the code.

do

printInt ,printString(without second line) ,printReal

by

get expression/factor

invokestatic java/lang/String/valueOf(I/F)Ljava/lang/String;(convert int/real to string)

invokevirtual java/io/PrintStream/println(Ljava/lang/String;)V

Subprogram part:

.method function type(in head ,get function info by entry of node)

i/fload #localnum(localnum is stored in entry->val when semantic check)

i/f/areturn if function, return if procedure

Assign node:

leftright = 1 (left part of assign)

if it has tail([])behind it, get/load the variable and a/f/laload the reference in sym_ref

generate put/ i/f(a)store code in the end of assignment

leftright = 0(right part of assign)

could be a function =>generate invoke function in node_sym_ref

else generate get/load,ldc num code in sym_ref

Procedure:

Invoke procedure when running to node_sym_ref

Operation node:

If node(op)->valuetype = real

=>if there is Node_Int(integer num) in any of side, add .0 in its instruction, ex. ldc 3->ldc 3.0 (recognized as a real num)

Get both sides of code(leftright = 0 get/load/invoke/ldc instruction) first

Then generate code of operation(i/f add/sub/div/mul)

For relational operation(>/</=/…)

While => generate the code if_cmp(inverse relation) (ex. gt->le ,eq->ne)

If=> if_cmpge/if_cmplt….

Labelnum+2(both if/while need to create 2 label at a time)

If statement node:

If_cmpge L#labelnum (true ->jump to label/false-> next line)

1. generate the code for condition(node_op usually)
2. generate the code for else part (in the end goto label of remaining part)
3. generate the code for then part(new label in first line)
4. generate new label for remaining part

while statement node:

1. create new label for begin ,and if_cmp** L# (in nodeop) (when condition is false(inverse cond is true)->jump outside the loop)
2. generate the code for do part (in the end goto label of beginning label)
3. generate new label for remaining part(outside the loop)

at the same time ,deal with the labelnum in order to avoid duplicate label

./myrun.sh->generate testfile.j and use jasmine & java make the code run, out of bound error can be detected in running time.