Homework#3: Semantic

2018008168 김지효

# Environment

* macOS Monterey 12.4 (M1)
* gcc Apple clang version 13.1.6

# Implementation

symtab.h에서 symbol table의 각 라인을 저장하는 BucketList, 각 스코프를 저장하는 ScopeList 구조체를 선언하였다. 또한 extern 변수 currScope를 선언하여 각 단계마다 현재의 scope를 저장하도록 하였다.

symtab.c에서 st\_insert 함수는 해당 scope에 해당 treeNode를 저장하도록 하였고, st\_lookup 함수는 해당 scope 및 parent scopes의 해당 name을, st\_lookup\_exluding\_parents 함수는 해당 scope의 해당 name을 갖는 BucketList를 찾도록 구현하였다.

analyze.c에서 insertNode 함수를 구현하였다. 각 treeNode의 type에 따라서 st\_insert 함수를 호출한다. 이 때 노드가 선언 노드라면 st\_lookup\_exluding\_parents 함수를 호출하여 error를 확인하고, 함수 선언이라면 새로운 스코프를 생성한다. 또한 새로운 블럭이 생성될 때도 새로운 스코프를 생성한다.

Anlyze.c에서 checkNode 함수를 구현하였다. 각 treeNode의 type에 따라서 type check를 한다.

위의 수정사항 등을 바탕으로 buildSymTab, typeCheck에서 올바른 symbol table을 생성하고 type checking을 하도록 하였다.

# Result

1. type\_error.cm

테스트 소스 및 결과는 다음과 같다.

int main(void)

{

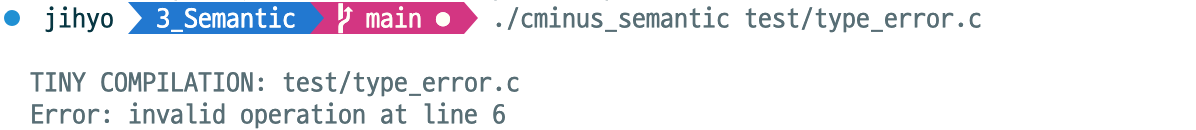
int x;

int y[3];

x + y;

return 0;

}



1. void\_var.cm

테스트 소스 및 결과는 다음과 같다.

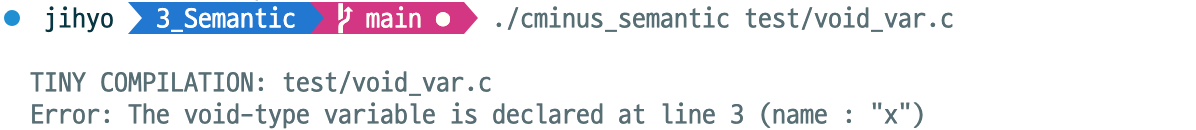
int main(void)

{

void x;

return 0;

}



1. invalid\_func.cm

테스트 소스 및 결과는 다음과 같다.

int x(int y)

{

return y + 1;

}

int main(void)

{

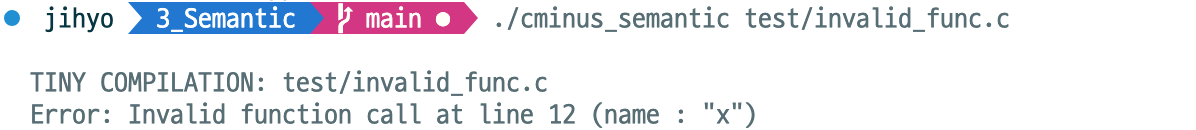
int a;

int b;

int c;

return x(a, b, c);

}



1. undeclared\_var.cm

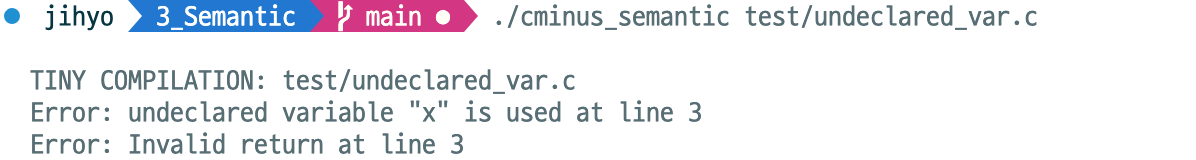
테스트 소스 및 결과는 다음과 같다.

int main(void)

{

return x;

}



1. invalid\_index.cm

테스트 소스 및 결과는 다음과 같다.

int main(void)

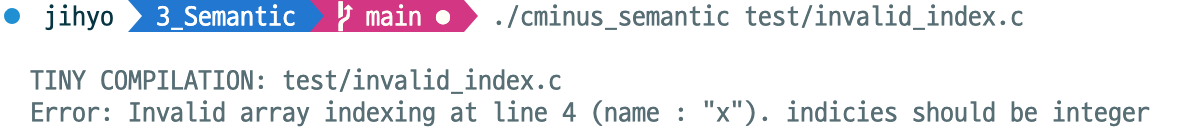
{

int x[5];

x[output(5)] = 3 + 5;

return 0;

}



1. invalid\_condition.cm

테스트 소스 및 결과는 다음과 같다.

int main(void)

{

if (output(5)) { }

return 0;

}

