Exame:

Projeto Analisador Léxico C-

Equipe:

Rafael Cassol

Alexandre Bernat

Erick Coelho

Turma 23

Objetivo do Trabalho

O objetivo deste trabalho é implementar o módulo de análise semântica de um compilador para a linguagem C-. Para isso foi usado o bison para a construção do parser, com a análise dos tokens sendo feita pelo módulo léxico, desenvolvido previamente usando a ferramenta flex. Com o resultado do parser, foi feita a análise semântica da árvore sintática gerada, apontando os erros sensíveis ao contexto.

Descrição do Analisador

O analisador semântico foi construído com base na confecção e análise da tabela de símbolos. Na análise da árvore sintática obtida do módulo de análise sintática, foram armazenadas as variáveis em uma hashTable para possibilitar uma análise sensível ao contexto. Nessa tabela, é armazenado o nome da variável, a localização na tabela hash, o tipo ID da variável (Var, função ou array), o tipo dado da variável (int, bool, void) e todos os números das linhas em que essas variáveis aparecem. Após a criação da tabela, são analisados os tipos das variáveis para identificar erros sensíveis ao contexto (Ex: chamada de função não declarada). Durante a construção da tabela também são localizados erros (Ex: variável declarada como void). Após a análise os erros semânticos e a tabela de símbolos é gerada, sendo armazenada na pasta results.

Resultados Obtidos

Ao executar os comandos descritos no readme, temos como output os arquivos txt correspondente à saída do analisador semântico.

Output do programa mdc.c

```
CMINUS COMPILATION: testfiles/mdc.c

Building Symbol Table...

Symbol table:

Variable Name Location Tipo ID Tipo Dado Line Numbers
------------- -------- ------- --------- ------------
gdc           0        FUNC    int         1    1
main          3        FUNC    void        8    8
u             1        VAR     int         1    1
v             2        VAR     int         1    1
x             4        VAR     int        10   10
y             5        VAR     int        11   11

Checking Types...

Type Checking Finished
```

Output do programa sort.c

No programa sort.c, o mesmo nome de variável é definido em escopos diferentes. Contudo, não contemplamos esse caso de erro no analisador semântico e ele não está diferenciando os escopos e então aponta erro, como mostrado abaixo. O correto seria não mostrar esses erros.

```
CMINUS COMPILATION: testfiles/sort.c

Building Symbol Table...
Erro semantico na linha 19: variavel ja declarada anteriormente
Erro semantico na linha 19: variavel ja declarada anteriormente
Erro semantico na linha 20: variavel ja declarada anteriormente
Erro semantico na linha 20: variavel ja declarada anteriormente
Erro semantico na linha 33: variavel ja declarada anteriormente

Symbol table:

Variable Name Location Tipo ID Tipo Dado Line Numbers
------------- -------- ------- --------- ------------
main          8        FUNC    void       32   32
sort          6        FUNC    void       19   19
minloc        1        FUNC    int         4    4
a             2        ARR     int         4    4   19
i             4        VAR     int         5    5   20   33
k             5        VAR     int         5    5   20
t             7        VAR     int        23   23
x             0        ARR     int         2    2
high          3        VAR     int         4    4   19

Checking Types...

Type Checking Finished
```

Output do programa simple.c

O programa simple.c só possui uma declaração de main vazia. O esperado era não haver erro semântico nesse caso, como se observa abaixo.

```
CMINUS COMPILATION: testfiles/simple.c

Building Symbol Table...

Symbol table:

Variable Name Location Tipo ID Tipo Dado Line Numbers
------------- -------- ------- --------- ------------
main          0        FUNC    void      1    1

Checking Types...

Type Checking Finished
```

Output do programa varvoid.c

No arquivo varvoid.c, não há main, e uma variável é declarada com o mesmo nome de uma função (função func). O analisador semântico identificou corretamente o erro, apontando que o identificador func já estava sendo usado. Além disso, uma variável foi declarada como tipo void, o que não é permitido. Novamente, o analisador identificou na checagem de tipos.

```
CMINUS COMPILATION: testfiles/varvoid.c

Building Symbol Table...
Erro semantico na linha 7: variavel ja declarada anteriormente
Erro semantico: funcao main() não declarada

Symbol table:

Variable Name Location Tipo ID Tipo Dado Line Numbers
------------- -------- ------- --------- ------------
a             0        VAR     void      1    1
func          1        FUNC    int       3    3    7

Checking Types...
Type error at line 1: declaracao invalida de variavel

Type Checking Finished
```

Output do programa void.c

O programa possui uma declaração de variável e uma de função repetidas - o analisador semântico identificou corretamente e apontou esses erros.

```
CMINUS COMPILATION: testfiles/void.c

Building Symbol Table...
Erro semantico na linha 9: funcao ja declarada anteriormente
Erro semantico na linha 16: variavel ja declarada anteriormente

Symbol table:

Variable Name Location Tipo ID Tipo Dado Line Numbers
------------- -------- ------- --------- ------------
outrofun      1        FUNC    void       5   5   9
main          2        FUNC    void      13   13
fun           0        FUNC    void       1   1
a             3        VAR     int       14   14  16

Checking Types...
Type error at line 19: chamada de função não declarada

Type Checking Finished
```

Output do programa nomain.c

O erro esperado era a função main não estar declarada, como de fato está contemplado no resultado gerado.

```
CMINUS COMPILATION: testfiles/nomain.c

Building Symbol Table...
Erro semantico: funcao main() não declarada

Symbol table:

Variable Name Location Tipo ID Tipo Dado Line Numbers
------------- -------- ------- --------- ------------
a             0        VAR     int       1   1

Checking Types...

Type Checking Finished
```

Casos não contemplados:

- “Variável não declarada” quando a variável não é global e está em um escopo.
- “Atribuição inválida”, quando a variável é do tipo int e o que está sendo atribuído é uma função que nada retorna.
- Análise de escopo

Gerador de Código

Utilizou-se como base para o gerador de código a geração de código para o TINY, disponibilizado pela professora.

Descrição do Gerador de Código

Não foi possível finalizar a implementação do gerador de código.

Resultados Obtidos

Não foi possível finalizar a implementação do gerador de código.