7 Abril 2019 – Fabio Silva | Nuno Bernardes

RELATÓRIO TRABALHO 1

1. INTRODUÇÃO

Neste trabalho foi-nos pedido para escrever um interpretador para uma linguagem de programação que é definida por um conjunto de instruções básicas. O objetivo deste trabalho era reforçar o que foi lecionado nas aulas práticas de forma a fundamentar os conhecimentos, então, para além de usarmos as estruturas fornecidas pelo professor, usamos:

i. Técnicas de referenciação;
ii. estruturas e unions;
iii. gestão de memória;
iv. construtores;
v. listas e tabela de hash;
vi. estruturas de dados;
vii. makefiles e
viii. estrutura bem definida de programas (que será explicada de forma sucinta mais para a frente).

2. FUNCIONAMENTO DO PROGRAMA

Não fugindo do enunciado para o problema proposto, o nosso programa recebe um ficheiro (em formato texto) com um conjunto de instruções bem definidas, guarda todas elas numa lista de instruções em que cada elemento dessa lista é um quádruplo, cria uma tabela de hash e, do início até ao fim da fila de instruções, analisa o quádruplo e realiza a devida operação.

3. ESTRUTURAÇÃO DO TRABALHO

Como foi anteriormente dito, o nosso trabalho segue uma estrutura bem definida de programas para que seja possível a cada membro do grupo trabalhar de forma autónoma em cada um dos programas sem atrapalhar o trabalho um do outro, seja mais fácil detetar erros e bugs que poderão aparecer no processo de compilação e seja possível promover um código limpo e conciso sem grandes repetições. O trabalho é então constituído por sete componentes:

A. "*estruturas.h*" e "*estruturas.c*"

   Nesta componente é onde realizamos a construção das instruções que são recebidas do ficheiro "input.t*xt*" e onde é adicionada a instrução ao fim da lista de instruções.

B. "*hash.h*" e "*hash.c*"

   Onde guardamos todas as funções relativas à tabela de hash tais como procura de valores na hash, inserção, inicialização da hash e retorno de valores da hash.

C. "*hashOperators.h*" e "*hashOperators.c*"

   Recebida uma instrução em formato de quádruplo, esta componente vai realizar todas as operações necessárias para que sejam adicionadas as instruções na tabela de hash, seja retornado os valores corretos da tabela de hash e todas as outras operações que impliquem ligações entre os quádruplos e a tabela de hash.

D. "*lista.h*" e "*lista.c*"

   Sendo possível a existência de variáveis com o mesmo valor de *key*, quando for procedida a inserção dos valores na tabela de hash é necessário guardar os valores distintos para cada variável distinta. Para isso usamos esta componente de lista, que trata de fazer as corretas alterações às listas que estão definidas em cada *key* da tabela de hash. Funções como criar uma nova lista, verificar a cabeça da lista, o fim, o tamanho, o elemento numa dada posição, concatenação, impressão da lista e adição de um novo valor ao fim da lista.

E. "*load.h*" e "*load.c*"

   Nesta componente é onde tratamos de analisar cada linha do ficheiro "*input.txt*", verificamos a palavras chave de cada instrução (por exemplo, verificamos se existe a palavra "MUL", "ADD", "IF_I", ...) e encaminha para as devidas funções cada instrução para que seja criado o quádruplo correto. Esta componente é a maior de todas (em termos de linhas de código) pois têm de ser analisados todos os padrões para cada instrução recebida pelo *loader*.

F. "*main.c*"

   Onde verificamos se os quádruplos foram bem criados (na fase de *debugging* do programa) e onde é percorrida toda a lista de instruções e chamada a componente das operações com a tabela de hash, realizando assim todas as operações pedidas pelo ficheiro "*input.txt*".

G. "*input.txt*""

Neste ficheiro é onde se escreve as instruções no formato enunciado (exemplo : "y = x + 2;"), tendo a palavra chave "*quit*" no fim do ficheiro para que o *loader* saiba quando parar de analisar mais instruções. O ficheiro é passado como argumento a compilar o "*main.c*".

4. CONCLUSÃO E COMENTÁRIOS FINAIS

Este trabalho deixou-nos mais à vontade com o que foi exposto nas aulas lecionadas até ser anunciado o trabalho. Não fugimos muito do que foi pedido no enunciado, mas foi-nos proposto para incluirmos um *parser* que analisa se as instruções no ficheiro "*input.txt*" estão bem construídas, e criar uma forma de escrever instruções e ir realizando as operações à medida que se fosse escrevendo mais instruções. Tentamos introduzir um pouco destes dois temas, mas não foi possível abordar os dois de forma profunda para que fosse mais um elemento forte do nosso trabalho. Concordamos em grupo que não fugimos às expectativas do trabalho que era proposto, e de que todo o processo e elaboração do mesmo foi sempre bem delineado com objetivos semanais.