Linguagens Formais e Autómatos

(Ano letivo de 2015/2016)

Guiões das aulas práticas

Guião #01 Exercícios em C

Sumário

Construção de programas em C/C++ em ambiente GNU/Linux.

Exercício 1 *Sobre o conjunto de caracteres* $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$*, considere o conjunto*

$$L_1 = \{u \in A^* \ : \ \exists_i \ x_i = x_{i-1} = x_{i-2}\}$$

onde A^* *representa o conjunto de todas as* strings *que se podem construir usando os caracteres de* A*.*

(a) O módulo `m1` *(ficheiros* `m1.h` *e* `m1.cpp`*) é uma implementação em C/C++ de uma função* `M`*, que recebe uma* string *como argumento e devolve* `true` *ou* `false` *consoante essa string pertence ou não a* L_1*. Analise o código desse módulo.*

(b) O programa `main-1.cpp` *processa as palavras passadas na linha de comando, indicando as que pertencem à linguagem* L_1*. Usa a função anterior para o fazer. Analise esse programa.*

(c) O programa `main-2.cpp` *lê palavras do* stdin *e processa-as, indicando as que pertencem à linguagem* L_1*. Usa a função anterior para o fazer. Analise esse programa.*

(d) Gere os programas executáveis `main-1` *e* `main-2`*. A* `Makefile` *fornecida facilita esse trabalho.*

(e) Teste os dois programas.

Exercício 2 *Sobre o conjunto* $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$*, considere as seguintes conjuntos:*

$$L_2 = \{u \in A^* \ : \ \#(\mathtt{ab}, u) > 1\}$$

$$L_3 = \{u \in A^* \ : \ \#(\mathtt{aba}, u) > 1\}$$

$$L_4 = \{u \in A^* \ : \ \#(\mathtt{b}, u) = 0 \ \vee \ \#(\mathtt{a}, u)\%2 = 0\}$$

$$L_5 = \{u \in A^* \ : \ \#(\mathtt{abc}, u) \geq (\#(\mathtt{ca}, u) + \#(\mathtt{cb}, u))\}$$

onde $\#(v, u)$ *representa o número de ocorrências da sub-*string v *na* string u*. Note que* $\#(aa, aaa) = 2$*.*
Para cada um dos conjuntos L_i*, com* $i = 1, 2, \cdots, 4$*,*

(a) *Construa um módulo em C/C++, chamado* `mi`*, que implemente uma função* `M`*,* `bool M(char *u);` *que recebe uma* string *como argumento e devolve* `true` *ou* `false` *consoante essa string pertence ou não a* L_i.

(b) *Altere a* `Makefile` *de modo a usar a função* `M` *anterior.*

(c) *Teste.*

Exercício 3 *Considere a linguagem* L_6 *definida sobre o conjunto* $A = \{\texttt{a}, \texttt{b}, \texttt{c}\}$ *cujas* strings *são tais que quaisquer duas ocorrências consecutivas da letra* `b` *apenas têm entre si 0 ou mais ocorrências da letra* `a` *ou 0 ou mais ocorrências da letra* `c`*, mas nunca simultaneamente letras* `a` *e* `c`.

(a) *Construa um módulo em C/C++, chamado* `mi`*, que implemente uma função* `M`*,* `bool M(char *u);` *que recebe uma* string *como argumento e devolve* `true` *ou* `false` *consoante essa string pertence ou não a* L_6.

(b) *Altere a* `Makefile` *de modo a usar a função* `M` *anterior.*

(c) *Teste.*

Exercício 4 *Considere que se pretende desenvolver uma aplicação que implemente uma máquina de calcular funcionalmente equivalente ao comando "bc -l". A interação com o utilizador é feita através dos* standard input *e* output. *Desenvolva a aplicação pretendida seguindo o faseamento definido a seguir.*

(a) *Comece por apenas considerar expressões aritméticas com as operações de adição (*`+`*), subtração (*`-`*), multiplicação (*`*`*), divisão (*`/`*) e potência (*`^`*), respeitando a precedência e associatividade habituais destas operações e aceitando qualquer número de parênteses. (Sugestão: use uma pilha (std::stack, boost::stack, ...), convertendo a expressão para notação polaca invertida.)*

(b) *Acrescente o uso de variáveis definidas por uma única letra. Terá de acrescentar a operação de atribuição (*`=`*). Uma variável usada numa expressão antes de lhe ter sido atribuído um valor assume o valor zero (0).*

(c) *Expanda de modo a aceitar variáveis com nomes mais abrangentes. Considere que estes nomes devem começar por uma letra, seguindo-se-lhe zero ou mais letras ou algarismos, sendo minúsculas e maísculas são letras distintas. Recorra ao uso das classes/templates* `std::map` *ou* `boost::map` *para implementar a tabela de variáveis.*

(d) *Expanda de modo a incluir as funções* `sqrt`*,* `sin`*,* `cos`*,* `tan` *e outras que considere pertinente.*
