Programação Imperativa
Departamento de Informática
Universidade do Minho

1º Ano LMCC

2004/2005

Ficha Teórico-Prática Nº 1
Ambiente (Unix) de Programação em C

24 de Fevereiro de 2005

1 Funcionamento das aulas Teórico-Práticas e Práticas

As aulas teórico-práticas da diciplina Programação Imperativa são baseadas na realização de uma ficha teórico-prática por semana, previamente disponível no site da disciplina, em `http://wiki.di.uminho.pt/wiki/bin/view/ProgramacaoImperativa/WebHome`.

Na aula teórico-prática semanal, a ficha será abordada sob o ponto de vista algorítmico. Espera-se que o aluno pense no problema e especifique uma possível solução. Na aula prática, os alunos deverão desenvolver os seus algoritmos em C e colocá-los a funcionar no computador. No fim, cada grupo deverá submeter ao sistema de submissões electrónico a ficha com as suas resoluções.

2 Ambiente de Programação em C

Para a resolução dos programas apresentados nas fichas de Programação Imperativa, aconselham-se todos os alunos a criar um ambiente de trabalho no computador que permita editar, compilar e executar os programas C, bem como preparar a documentação associada aos programas desenvolvidos.

2.1 Comentários em C

Exercício 1: Comentários

Diga quais as formas permitidas para misturar comentários com o texto dos programas, na linguagem C.

2.2 Primeiro programa C

Exercício 2: "Hello world!"

Desenvolva um programa C que escreve a string "O meu primeiro programa"no écran do computador.

2.3 Compilação separada em C

Exercício 3: Opções do gcc

Diga, sucintamente, o que faz cada uma das seguintes opções do CC:

1. `gcc -E`

   Invoca o pré-processador; o resultado é um novo texto resultante da aplicação do pré-processador. Defina a frase do primeiro programa como uma constante. Execute o compilador com esta opção e analise o resultado.

2. `gcc -S`

   Compila o programa, resultando um ficheiro ainda em assembly (ou seja, por assemblar). Analise o resultado.

3. `gcc -c`

   Compila e converte o assembly em código máquina, mas não invoca o linker. Experimente e veja que é gerado um ficheiro com extensão ".o".

4. `gcc -o` ou só `gcc`

   Compila, assembla e linka. Gera um arquivo pronto a executar de nome "a.out".

Exercício 4: Soma de 2 números inteiros

Escreva um novo programa C, que soma dois números inteiros, previamente declarados e inicializados respectivamente com 7 e 9, e escreve o resultado da soma.

Exercício 5: Utilização do gcc

Escreva os comandos para realizar as acções seguintes:

1. Comando para compilar o programa C (sem gerar o executável).

2. Comando para gerar um executável `soma.exe`

---

3 Makefiles

Analise a seguinte Makefile:

```
soma: soma.c
        gcc -o soma.exe soma.c

ppp: soma.c
        gcc -E soma.c

asm: soma.c
        gcc -S soma.c
```

Exercício 6: Utilização de makefiles

Considerando a Makefile apresentada, e que a mesma está guardada no arquivo Makefile, execute os seguintes comandos:

1. Comando para gerar o executável.
2. Comando para correr apenas o pré-processador.
3. Comando para gerar o código Assembley.

Exercício 7: Tratamento de excepções em makefiles

Estude o funcionamento do comando make, e diga:

1. Qual a diferença entre os comandos:

   ```
   limpa:
   rm *.log *.dvi *.aux
   ```

   ```
   limpa:
   -rm *.log *.dvi *.aux
   ```

   Quando a execução de um comando retorna erro, o make pára. Com o prefixo '-', o make não pára se o comando retornar um erro.

---

4 LaTeX

Exercício 8: Um pequeno relatório

Com a ajuda do professor, elabore em LaTeX, um pequeno relatório do trabalho realizado nesta ficha. Para tal edite o ficheiro TeXdesta ficha, acrescentando as soluções que desenvolveu na aula. No fim, gere um documento em PDF.