Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e de Computadores

Programação de Sistemas Computacionais

Verão de 2020/2021

Série de Exercícios 1

Realize os exercícios seguintes usando a linguagem C. Não se esqueça de testar devidamente o código desenvolvido, bem como de o apresentar de forma cuidada, apropriadamente indentado e comentado, não sendo necessário relatório. Assegure-se de que o compilador não emite qualquer aviso sobre o seu código com a opção `-Wall`. As resoluções dos exercícios 1 a 4 devem integrar o respectivo programa de teste. Tenha em consideração que os exercícios que não forem demonstrados a funcionar serão considerados como não tendo sido realizados. Contacte o docente se tiver dúvidas. Encoraja-se a discussão de problemas e soluções com outros colegas, mas a partilha directa de soluções leva à anulação das entregas de todos os estudantes envolvidos.

1. Programe a função **`print_integer`** que representa em texto, na base de numeração **`base`**, o valor inteiro contido em **`value`**. O texto é depositado no *array* de caracteres indicado pelo ponteiro **`str`**, no formato de *string*. A função retorna a dimensão da *string* produzida.

   ```
   int print_integer(char *str, int value, int base);
   ```

2. Programe a função **`print_float`** que representa em texto, na base decimal, o valor real contido em **`value`**. O texto é depositado no *array* de caracteres indicado pelo ponteiro **`str`**, no formato de *string*. Na programação desta função não deve utilizar operações sobre valores do tipo *float*. A função retorna a dimensão da *string* produzida.

   ```
   int print_float(char *str, float value);
   ```

3. Programe a função **`mini_sprintf`** segundo a definição da função **`sprintf`** da biblioteca normalizada da linguagem C, limitada aos especificadores de conversão **`c`**, **`s`**, **`d`**, **`x`** e **`f`**, sem adornos. Na programação desta função não deve utilizar a função **`sprintf`**. Consulte o exemplo da página 156 do livro The C Programming Language.

   ```
   int mini_sprintf(char *str, const char *format, ...);
   ```

4. Programe a função **`extract_data`** que extrai informação sobre pessoas de um texto. O texto é constituído por uma sequência de linhas, terminadas por '\n', em que cada linha contém os dados de uma pessoa. Estes dados são o nome, a data de nascimento e o número fiscal. O parâmetro **`people`** é um *array* de ponteiros para as *structs* que vão receber a informação, o parâmetro **`people_size`** indica a dimensão máxima do *array* **`people`** e o valor de retorno da função é o número de pessoas processadas.

   ```
   struct date {
       int day, month, year;
   };

   typedef struct person {
       char name[100];
       struct date date;
       int nif;
   } Person;

   size_t extract_data(Person *people[], size_t people_size, char *text);
   ```

   Exemplo de texto com a informação de duas pessoas:

   "Manuel,15,3,2002,125745045\nJoaquim,25,4,1974,122003088\n"

5. Realize um programa para procurar uma palavra num ficheiro de texto. As linhas que contêm a palavra são escritas no ficheiro de saída, prefixadas com o número de ordem da linha no ficheiro de entrada.

   `$ find_in_file <opções> <palavra>`

   O campo **`<opções>`** pode conter qualquer sequência das seguintes opções, inclusivé ser vazio:

   `-o <ficheiro>` indica o nome do ficheiro de saída com o resultado;
   em caso de omissão usar **`stdout`**;

   `-i <ficheiro>` indica o nome do ficheiro de entrada, com o texto a processar;
   em caso de omissão usar **`stdin`**;

   `-c` indica se a comparação léxica é sensível a maiúsculas e minúsculas;
   em caso de omissão é insensível;

   `-w` indica se se deve considerar apenas palavras completas;
   em caso de omissão considera também subpalavras.

   Valorizam-se soluções que usem a função getopt no processamento das opções.

Data recomendada de entrega: 25 de abril de 2021

ISEL, 24 de março de 2021