 **Instituto Politecnico de Beja**

**Escola Superior de Tecnologia e Gestão**

**Engenharia Informatica**

**Arquitetura de Computadores**

**Trabalho prático 1**

**Aluno:** Pedro Pita nº19933 e Tomas Ramos nº19934

**Docentes:** João Carlos Martins e Gonçalo Fontes

Índice

Introdução ---------------------------------------------------------------------------------------- 3

Desenvolvimento ----------------------------------------------------------------------------- 4-6

1ª Parte ---------------------------------------------------------------------------------- 4

2ª Parte ---------------------------------------------------------------------------------- 5

3ª Parte ---------------------------------------------------------------------------------- 6

Conclusão ----------------------------------------------------------------------------------------- 7

Webgrafia---------------------------------------------------------------------------------------- 8

**Introdução**

Este trabalho consiste na implementação de quinze funcionalidades existentes na biblioteca **string.h** [(**clique** aqui)](https://www.tutorialspoint.com/c_standard_library/string_h.htm) da linguagem de programação C, diretamente em assembler para a arquitetura ARM sobre a plataforma Raspberry Pi.

O trabalho será dividido em três partes:

**1º parte (implementação das funcionalidades):** implementar as quinze funcionalidades;

**2º parte (teste das funcionalidades):** testar as quinze funcionalidades individualmente de forma estática, com valores pré-definidos no código e sem qualquer possibilidade de alteração pelo utilizador;

**3ª parte (implementação do menu):** criar um menu em que o utilizador tem a possibilidade de escolher qual é a funcionalidade que deseja testar, e em seguida pode inserir os valores necessários para esse teste (valores das strings, n, etc..), recebendo logo de seguida o respetivo resultado e podendo ter a oportunidade de repetir o processo e escolher outra funcionalidade para testar.

**Desenvolvimento**

**1ª parte (implementação das funcionalidades)**

Está primeira parte consiste na implementação de quinze funcionalidades da biblioteca string.h.

As funções implementadas foram as seguintes:

1. [memcpy](https://www.tutorialspoint.com/c_standard_library/c_function_memcpy.htm)
2. [strlen](https://www.tutorialspoint.com/c_standard_library/c_function_strlen.htm)
3. [memchr](https://www.tutorialspoint.com/c_standard_library/c_function_memchr.htm)
4. [memmove](https://www.tutorialspoint.com/c_standard_library/c_function_memmove.htm)
5. [strcpy](https://www.tutorialspoint.com/c_standard_library/c_function_strcpy.htm)
6. [strncpy](https://www.tutorialspoint.com/c_standard_library/c_function_strncpy.htm)
7. [strncmp](https://www.tutorialspoint.com/c_standard_library/c_function_strncmp.htm)
8. [strcmp](https://www.tutorialspoint.com/c_standard_library/c_function_strcmp.htm)
9. [strspn](https://www.tutorialspoint.com/c_standard_library/c_function_strspn.htm)
10. [strrchr](https://www.tutorialspoint.com/c_standard_library/c_function_strrchr.htm)
11. [strcspn](https://www.tutorialspoint.com/c_standard_library/c_function_strcspn.htm)
12. [strpbrk](https://www.tutorialspoint.com/c_standard_library/c_function_strpbrk.htm)
13. [strxfrm](https://www.tutorialspoint.com/c_standard_library/c_function_strxfrm.htm)
14. [memset](https://www.tutorialspoint.com/c_standard_library/c_function_memset.htm)
15. [strstr](https://www.tutorialspoint.com/c_standard_library/c_function_strstr.htm)

(clicar no nome da função para mais informações)

**Observações**

No decorrer do desenvolvimento das funcionalidades detetamos alguns problemas de manipulação das strings, sendo que o primeiro exemplo revela-se o mais complicado. Após o finalizarmos percebemos ter ficado um pouco mais familiarizados com a linguagem e com a manipulação das strings. Por essa razão, foi-nos de certa forma "facilitado" o desenvolvimento das funcionalidades restantes.

**Desenvolvimento (continuação)**

**2ª parte (teste das funcionalidades)**

Para testarmos todas as funcionalidades criamos um programa com as funcionalidades e ao ser executado testa todas as funcionalidades individualmente de forma estática, com valores pré-definidos no código e sem qualquer possibilidade de alteração pelo utilizador.

**Observações**

Esta parte do trabalho foi provavelmente a mais importante. Nela pudemos perceber vários erros que não tínhamos dado conta na parte 1. A presente revelou-se mais demorada em comparação à anterior, uma vez que não contou apenas com o desenvolvimento dela, mas também como o melhoramento também da parte.

**Desenvolvimento (continuação)**

**3ª parte (implementação do menu)**

Nesta parte adicionamos o menu e além disso modificamos os testes das funções, que antes estavam estáticos, para uma forma permite o utilizador inserir os dados necessários para testar a função.

**Observações**

Nesta parte começamos inicialmente por desenvolver o menu e a chamada das funcionalidades de forma estática (reaproveitando os testes da 2ª parte).

Após o menu estar a funcionar perfeitamente utilizado os testes estáticos, começamos a desenvolver os testes utilizando a função scanf para permitir ao utilizador a inserção dos dados necessários para o funcionamento das funções.

E finalmente, desenvolvemos a opção de o utilizador poder repetir um teste, onde após mostrar o resultado da funcionalidade testada perguntamos ao utilizador se deseja testar uma nova funcionalidade, se o utilizador desejar retornamos o programa para o main onde será mostrado novamente as funcionalidades disponíveis para teste.

**Conclusão**

Ao longo deste trabalho deparamo-nos com algumas dificuldades, visto não estarmos familiarizados com uma linguagem tão próxima à linguagem máquina. Porém, apesar dessa difícil adaptação, foi interessante percebermos o modo como a máquina processa certas instruções. Em outras linguagens como por exemplo C, muitas das vezes aplicamos certas instruções apenas com uma chamada a uma função, sem sequer sabermos o que está por detrás da mesma e este trabalho fez nos perceber um pouco mais acerca desse processo interno e acerca de como os dados são armazenados.

**Webgrafia**

[RASPBERRY PI ASSEMBLER](https://personal.utdallas.edu/~pervin/RPiA/RPiA.pdf)

[ARM assembler in Raspberry Pi](https://thinkingeek.com/arm-assembler-raspberry-pi/)

[String.h](https://www.tutorialspoint.com/c_standard_library/string_h.htm)