Uma imagem com texto

Descrição gerada automaticamente

Sistemas Operativos

2022/2023

**Taxas de Leitura/Escrita de processos em bash**

**Hugo Correia** 108215 P6

**Sara Almeida** 108796 P6

**Professor:**

Nuno Lau (nunolau@ua.pt)

**Índice**

**Introdução3**

**Estrutura do código4**

Declaração de variáveis globais4

Funções4

num\_Int()3

validate\_date()3

usage()3

arguments\_validation()3

listar\_processos()3

PID\_filter()3

sort\_process()3

**Escrever título do capítulo (nível 1)4**

Escrever título do capítulo (nível 2)5

Escrever título do capítulo (nível 3)6

**1 – Introdução**

No contexto da disciplina de Sistemas Operativos, foi-nos proposta a realização deste projeto que tem como objetivo a visualização e tratamento dos processos a decorrer atualmente no nosso computador, mais especificamente, a taxa de leitura/escrita dos mesmos em bash.

A ferramenta utilizada para a realização deste trabalho foi o VS CODE, um editor conhecido por ambos os elementos deste grupo e com o qual estamos habituados a trabalhar no seguimento de outras disciplinas.

#ideia base ou wtv asdkahdka

A realização deste projeto permitiu claramente o alargamento do nosso conhecimento com a linguagem de bash e clarificou a nossa perceção dos processos do sistema operativo Linux.

**2 – Estrutura do código**

2.1. – Declaração de variáveis globais

Primeiramente, decidimos usar arrays associativos, para o tratamento da informação, por estes serem estruturas de dados que permitem associar determinados valores a “keys” facilitando a manipulação dos dados. Sendo assim, começámos por inicializar os arrays *READBI*, *READBF*, *WRITEBI*, *WRITEBF* e *PID\_array*.

#imagem

READBI - …

READBF - …

WRITEBI - …

WRITEBF - …

PID\_array - …

2.2. – Funções

Grande parte do nosso código fonte da solução encontrada é baseado em funções, por questões de organização e de mais fácil manipulação de dados.

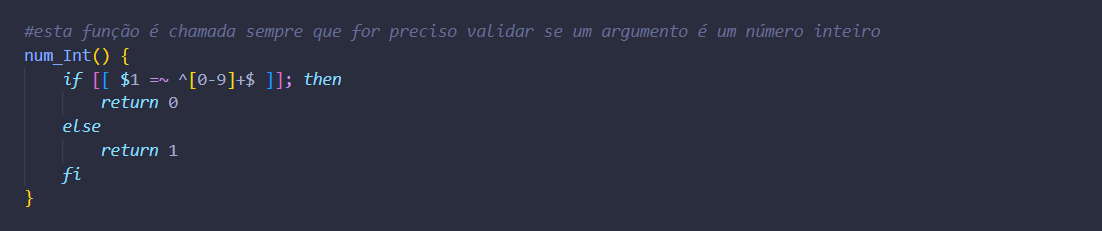
2.2.1 – Função num\_Int()

Figura - Função num\_Int()

Neste campo, começámos por criar uma função que será chamada sempre que for necessário validar se um argumento é ou não um número inteiro, pois vimos ser útil ao longo da resolução do guião.

Uma imagem com texto

Descrição gerada automaticamente2.2.2 – Função validate\_date()

Figura - Função validate\_date()

Criámos também esta função que será utilizada mais tarde, no tratamento das opções, para validar o formato das datas possivelmente passadas como argumento, mais especificamente, às opções *-s* e *-e*, pois queríamos que apenas datas com o formato apresentado no guião pudessem ser passadas como argumento.

Uma imagem com texto

Descrição gerada automaticamente2.2.3 – Função usage()

Figura - Função usage()

Aqui, criámos a função usage que imprime no terminal todas as opções válidas e a sua finalidade, sendo estas : *-c, -s, -e, -u, -m, -M, -p, -r e -w*. Esta função será chamada aquando da validação dos argumentos e opções caso algum seja inválido, como iremos aprofundar mais à frente.

* Opção *-c* : Permite filtrar os processos através de uma expressão *regex* passada como argumento à mesma, imprimindo no terminal apenas os processos que obedeçam a essa expressão.
* Opções *-s* e *-e* : Permitem a seleção dos processos através da especificação de um período temporal com data mínima e data máxima para o início do processo.
* Opção *-u* : A utilização desta opção permite a impressão no terminal unicamente dos processos que estejam a ser realizados pelo utilizador passado como argumento à mesma.
* Opções *-m* e *-M* : Permitem a filtração dos processos consoante uma gama mínima e máxima de pids, respetivamente.
* Opção *-p* : Esta opção permite ao utilizador escolher a quantidade de processos a visualizar no terminal.
* Opção *-r* : Esta opção tem como finalidade imprimir no terminal todos os processos, ou os filtrados pelo utilizador através das outras opções, por ordem inversa à ordem default.
* Opção *-w* : Por outro lado, esta opção irá ordenar os processos a ser visualizados consoante os *write values.*