Lab01 – Fundamentos, Arquitetura de Computadores e Sistemas Operacionais

(★★) Defina em termos próprios os seguintes conceitos, apontando exemplos do seu uso na prática, em até cinco frase por definição.

- Sistema Operacional
- BIOS
- API
- Chamada de sistema
- Kernel

(★★★) Descreva a **funcionalidade** das cinco principais estruturas de dados utilizadas na implementação dos sistemas operacionais, isto é, listas, pilhas, filas, função hash e bitmap.

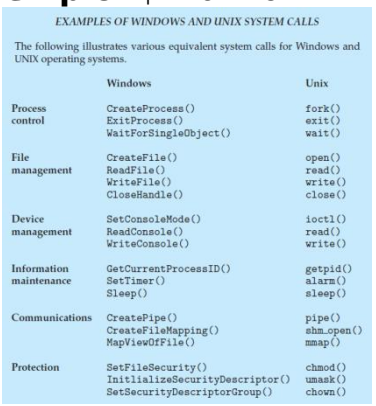
- Escolha duas opções dentre as cinco e escreva um **exemplo** do uso destas estruturas em código. A escolha da linguagem fica ao seu critério (*e.g.*, C/C++, Java, Python, Rush, Go, etc.)

(★) O que é uma API? qual é a sua utilidade dentro do sistema operacional? que relação que guarda com as chamadas de sistema?

(★) Qual é o nome da API que utilizam os sistemas Windows, Unix (*e.g.* Solaris, Linux, MacOS X) e Java Virtual Machine (JVM) e quais são as suas principais características?

(★) Descreva cinco chamadas de sistema utilizadas nos sistemas Unix detalhando sua sintaxe e a operação que executam. Cada função deverá pertencer a uma categoria diferente, sejam elas “Process control”, “File management”, “Device management”, “Information Maintenance”, “Communication” e “Protection”.

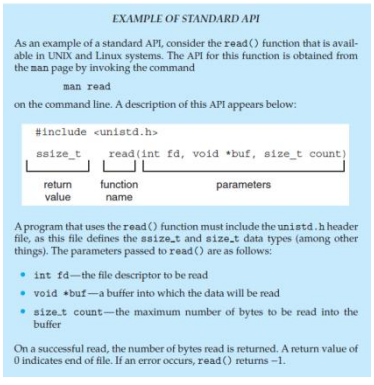
- **Sugestão:** Leia a documentação na “man page” de cada função Unix e o livro-texto Seção 2.3.3 Types of System Calls.
  - **Exemplo:** $ man fork *# fornece informações da função ‘fork’*



EXAMPLES OF WINDOWS AND UNIX SYSTEM CALLS

The following illustrates various equivalent system calls for Windows and UNIX operating systems.

	Windows	Unix
Process control	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
File management	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
Device management	SetConsoleMode() ReadConsole() WriteConsole()	ioctl() read() write()
Information maintenance	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
Communications	CreatePipe() CreateFileMapping() MapViewOfFile()	pipe() shm_open() mmap()
Protection	SetFileSecurity() InitlializeSecurityDescriptor() SetSecurityDescriptorGroup()	chmod() umask() chown()



EXAMPLE OF STANDARD API

As an example of a standard API, consider the `read()` function that is available in UNIX and Linux systems. The API for this function is obtained from the `man` page by invoking the command

```
man read
```

on the command line. A description of this API appears below:

```
#include <unistd.h>

ssize_t    read(int fd, void *buf, size_t count)
```

return value — function name — parameters

A program that uses the `read()` function must include the `unistd.h` header file, as this file defines the `ssize_t` and `size_t` data types (among other things). The parameters passed to `read()` are as follows:

- `int fd`—the file descriptor to be read
- `void *buf`—a buffer into which the data will be read
- `size_t count`—the maximum number of bytes to be read into the buffer

On a successful read, the number of bytes read is returned. A return value of 0 indicates end of file. If an error occurs, `read()` returns −1.

(★) O que é uma máquina virtual? e um container? Cite dois exemplos do seu uso na atualidade.

(★★) Descreva o **princípio de funcionamento** do programa bootstrap e da memória cache.

(★★) Quando trabalhamos com Gestão de Projetos e Processos definimos as metodologias de desenvolvimento de projetos de software mais importantes utilizadas na atualidade. Defina em até cinco frases por item uma metodologia utilizada para:

- Gestão de projetos
- Gestão de processos