

# TP 1 de Algoritmos e Estruturas de Dados

## Introdução

1. [Modulos e funções](#)
2. [Como compilar o projeto](#)
3. [Tipos abstratos](#)
  - [Processo](#)
  - [Cursor](#)
  - [Celula](#)
  - [Lista](#)
4. [Como compilar a documentação](#)

---

## Módulos

### UTILS

| Função   | Tipo   | Argumentos | Descrição                                                 |
|----------|--------|------------|-----------------------------------------------------------|
| menu()   | void   |            | Imprime na saída padrão as opções de interação do sistema |
| cls()    | void   |            | Limpa a saída padrão                                      |
| pid(int) | char * |            | Gera um id único do processo de tamanho 20                |

---

## Comando para compilar o projeto

```
make
```

# Tipos abstratos

## Processo

- Tipo

| Campo           | Tipo       | Descrição                                                                               |
|-----------------|------------|-----------------------------------------------------------------------------------------|
| PID             | int        | Um indentifiador único de cada processo ( PID ). Usado para ordenar os processos        |
| prioridade      | int        | A prioridade do processo ( de 1 a 5 ).( Gerado aleatoriamente na criação do processo ). |
| horario_criacao | struct tm* | A hora em que o processo foi criado, alocado na memória e inicializado.                 |

```
typedef struct Tprocesso
{
    int PID; // identificador do processo
    int prioridade; //prioridade do processo
    struct tm* horario_criacao; // horario de criacao do processo
} Processo;
```

- Funções

| Função              | Tipo       | Argumentos               | Descrição                                                                     |
|---------------------|------------|--------------------------|-------------------------------------------------------------------------------|
| inicializa_processo | void       | (Processo* )             | Inicializa o processo com os valores de PID, prioridade e horario de criacao. |
| imprime_processo    | void       | (Processo)               | imprime os campos do processo.                                                |
| get_PID             | int        | (Processo*)              | Retorna o valor do campo PID.                                                 |
| set_PID             | void       | (Processo* , int)        | Define o valor do campo PID.                                                  |
| get_prioridade      | int        | (Processo*)              | Retorna o valor do campo prioridade.                                          |
| set_prioridade      | void       | (Processo* , int)        | Define o valor do campo prioridade.                                           |
| get_horario_criacao | struct tm* | (Processo*)              | Retorna o valor do campo horario_criacao.                                     |
| set_horario_criacao | void       | (Processo* , struct tm*) | Define o valor do campo horario_criacao.                                      |

# Cursor

- Tipo

Cursor representa um apontador (índice) para um elemento de uma lista.

```
typedef int cursor;
```

# Celula

- Tipo

Celula é um tipo abstrato que representa um elemento de uma lista.

| Campo    | tipo       | Descrição                                      |
|----------|------------|------------------------------------------------|
| processo | Processo * | O processo que está armazenado na celula.      |
| prox     | cursor     | O apontador ( índice ) para a próxima celula.  |
| ant      | cursor     | O apontador ( índice ) para a celula anterior. |

```
typedef struct Tcelula {
    Processo *processo;
    cursor ant;
    cursor prox;
} Celula;
```

- Funções

| Função                 | Tipo | Argumentos               | Descrição                                                                                        |
|------------------------|------|--------------------------|--------------------------------------------------------------------------------------------------|
| inicializa_celula_nula | void | (Celula* ,<br>int)       | Inicializa a celula com o valor de prox passado como parametro, ant igual a -1 e processo NULL . |
| inicializa_celula      | void | (Celula* ,<br>Processo*) | Inicializa a celula com prox e ant como -1, e o campo processo passado como parametro.           |

# Lista

- Tipo

| Campo               | tipo    | Descrição                                                 |
|---------------------|---------|-----------------------------------------------------------|
| primeiro            | cursor  | O apontador ( indice ) para a primeira celula.            |
| ultimo              | cursor  | O apontador ( indice ) para a ultima celula.              |
| primeira_disponivel | cursor  | O apontador ( indice ) para a primeira celula disponivel. |
| numCelOcupadas      | int     | O numero de celulas ocupadas.                             |
| plista              | Celula* | O ponteiro para a lista.                                  |
| tamanho             | int     | O tamanho da lista.                                       |

```
typedef struct Tlista {
    cursor primeiro;
    cursor primeira_disponivel;
    cursor ultimo;
    int numCelOcupadas;
    Celula *plista;
} Lista;
```

- Funções

| Função           | Tipo | Argumento         | Descrição                                                                                                                                      |
|------------------|------|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| inicializa_lista | void | (Lista*, int)     | Aloca um espaço de memória para a lista, inicializa o primeiro e ultimo como -1 e o primeira_disponivel e o numero de celulas ocupadas como 0. |
| adiciona_celula  | void | (Lista*, Celula*) | Adiciona celulas ordenadamente à lista.                                                                                                        |
| remove_celula    | void | (Lista*, int)     | Remove uma celula da lista em determinado índice.                                                                                              |
| remove_primeiro  | void | (Lista*)          | Remove a primeira celula da lista.                                                                                                             |
| remove_ultimo    | void | (Lista*)          | Remove a ultima celula da lista.                                                                                                               |
| imprime_lista    | void | (Lista*)          | Imprime todos os processos da lista.                                                                                                           |

---

## Comandos principais

- adiciona alteração para ser enviado

```
git add <nome_do_arquivo>
```

- adiciona todas alterações

```
git add .
```

- commitar as alterações

```
git commit -m "mensagem"
```

- enviar as alterações para o repositório remoto

```
git push origin master
```

- baixar as alterações do repositório remoto

```
git pull <nome_do_repositorio> <branch>
```

- verificar o status do repositório

```
git status
```

- verificar o log do repositório

```
git log
```

- busca alterações no repositório

```
git fetch -p -t
```

---

## Compilar a documentação

```
make doc
```

- Gera arquivo PDF com o mesmo nome do arquivo markdown (não mudar nome do arquivo .md, crie uma cópia com outro nome e compile a cópia, ou mude o nome do arquivo resultado)

```
md-to-pdf readme.md
```

- Para gerar o arquivo PDF, é necessário ter instalado o programa `md-to-pdf`, caso não tenha a ferramenta de build de documentação instalada

```
pip install md-to-pdf
--- ou ---
npm install md-to-pdf
```