Algoritmos e Tipos Abstratos de Dados

Git Classroom (Projeto)

Ler com atenção os pré-requisitos e regras antes de avançar.

Durante este documento a sintaxe `<foo>` representa um *placeholder* que deverá ser substituído apropriadamente, de acordo com a descrição de "foo".

Pré-requisitos:

- Cada aluno deverá ter uma conta registada no GitHub, seja com email pessoal ou institucional.
- Deverá conhecer o *username* e *email* associado a esta conta.

Regras:

- Os grupos são formados no mínimo por 3 alunos e no máximo por 4;
  - Se o grupo for de 3 alunos, terão de ser obrigatoriamente do mesmo docente de laboratório;
  - Se o grupo for de 4 elementos, poderão os 4 ser do mesmo docente de laboratório **ou** 2 de um docente e os outros 2 de outro docente.
- Cada grupo designa internamente o **leader** do grupo; este será responsável por:
  - Fazer a **primeira aceitação** do *assignment* e criar a *team* do grupo;
  - Inscrever o grupo no questionário Moodle respetivo, e;
  - Submeter o projeto até à data limite no Moodle.

Passos a seguir

Nota 1: Será mais fácil fazer preliminarmente *sign in* no *GitHub*.

Nota 2: Apenas o **leader** cria uma nova *team* (equipa); os restantes membros associam-se (fazem *join*) à equipa criada pelo leader).

Os seguintes passos deverão ser seguidos pela ordem apresentada:

1. O **leader** do grupo inicia a aceitação do *assignment* através do endereço respetivo disponibilizado no Moodle, e.g., `PNormal assignment`.
2. O **leader** cria uma nova *team* cujo nome deverá ter o seguinte formato:

   `<sigla(s)_docente(s)>_<numero_aluno_leader>`, e.g., `AP_201234567`.

   - Se o grupo tiver 2 alunos de docentes distintos, coloque ambas as siglas, e.g., `AP_ML_201234567`.
3. Finalmente, o **leader** aceita o *assignment*. Daqui resulta uma *team* (com o nome especificado) e um endereço de um *repositório* **privado** para utilização do grupo – **tome nota do nome do grupo e endereço do repositório**.
4. Os restantes elementos do grupo repetem o passo **1**, no passo **2** escolhem a team criada pelo leader e comunicada por este e fazem a aceitação no passo **3**. Ficarão todos com acesso ao repositório do grupo.

Considerações

- Cada *team*/leader é responsável pela gestão da mesma; se detetar que algum colega se associou indevidamente à equipa, deverá removê-lo;
- O repositório contém inicialmente código fornecido e considerações importantes descritas no `README.md`;
  - Este repositório é **privado** e só é acessível pelos membros da equipa e pelos docentes da UC;
- Excepto nas funcionalidades de importação iniciais, as restantes deverão ser distribuídas equititativamente pelos elementos do grupo; os *commits* individuais serão utilizados para avaliar a participação individual dos membros do grupo.

Utilização do repositório

É recomendado que os membros do grupo façam algumas "experiências" iniciais antes de iniciarem o desenvolvimento do projeto, no que toca à utilização do *versionamento* do projeto.

Nota: Não é aconselhável que trabalhem simultaneamente no desenvolvimento das mesmas funções (será difícil sincronizar o código).

Para os seguintes comandos, utilize o *terminal* integrado do VS Code.

Git Clone

Cada elemento do grupo faz uma cópia local do seu repositório através do comando `git clone`, e.g.:

```
$> git clone <endereco_repositorio_grupo> <nome_pasta_local>
```

Todas as alterações locais (na máquina), continuarão a ser locais até que sejam "empurradas" para o repositório central.

Garanta que, posteriormente, está sempre a trabalhar dentro da pasta `<nome_pasta_local>`, dado que os comandos seguintes só fazem sentido dentro desta.

- Exemplo para abrir a pasta local do projeto no VS Code através da linha de comandos:

  ```
  $> code <nome_pasta_local>
  ```

- Alternativamente, utilize o explorador de ficheiros para abrir esta pasta com o VS Code (i.e., "Abrir Com…" e escolher o VS Code)

Git Push

Quando um elemento do grupo tiver código *finalizado* para partilhar com os restantes colegas, deverá seguir a seguinte sequência de passos:

```
$> git add .
$> git commit -m "<muito breve descricao do que foi feito>"
$> git push -u origin main
```

Git Pull

Sempre que (re)iniciar o desenvolvimento, cada aluno deverá garantir que tem a "última versão" do código da equipa.

As alterações que tenham sido feitas e "commited" para o servidor central serão sincronizadas com a sua cópia local.

Isto é feito com:

```
$> git pull
```

Será apresentado:

- O código que foi modificado, relativamente ao repositório local, ou;
- `"Already up to date"`, se não existe nenhuma diferença entre a cópia local do repositório e o repositório central.

Se foram feitas alterações paralelas às mesmas funções, terá de ser dito como resolver essas *colisões*. Evite isto, através de planeamento e sincronização das atividades do grupo de trabalho.

Integração Git e VS Code

O VS Code possui integração direta com repositórios *git*; pode utilizar essas funcionalidades (barra esquerda) em vez dos comandos no terminal.

Suporte e ajuda

Peça suporte ao seu docente de laboratório.