

Trabalho   
 Algoritmia e Programação   
2017/18

**“GAU - Gestão da Avaliação na UAL”**

**FASE I + FASE II Partes A&B**

**FASE II Parte B**

A última parte do presente Trabalho tem em conta o arquivo serializado dos objectos/dados que permitem guardar e ler a informação desejada em ficheiro binário, mantendo-se a integridade da estrutura interna dos objectos/dados.

Neste contexto, pretende-se serializar os objectos listas que foram criados deste Trabalho, para que possam gravá-los em ficheiro binário utilizando o módulo Pickle do Python. Para além do indicado na aula teórica, consultar para melhor estudo e esclarecimento:

Python 3.6.4 - 21.1 pickle – Python object serialization <https://docs.python.org/3/library/pickle.html>

(desta extensa exposição apenas utilizámos a versão simplificada de serialização com os métodos dump() e load() exemplificados no ponto 12.1.8 Examples, suficiente para o arquivo em questão)

Deve considerar os requisitos:

1. Guardar em ficheiros binários, com serialização Pickle, as várias listas que o GAU utiliza, quando se abandona o programa;
2. Uma vez que existam estes ficheiros serializados, ao iniciar a execução do GAU, este deve realizar a leitura dos ficheiros referidos no ponto anterior, repondo os objectos listas acessíveis no estado em que foram guardados.

**Instruções sobre a entrega do Trabalho**

**Datas:**

• Data limite de entrega da Fase I do Projecto: 5 de Janeiro de 2018.

• Data limite de entrega da Fase II do Projecto:

Turno Diurno: 22 de Janeiro de 2018

Turno Pós-laboral: 30 de Janeiro de 2018

**Procedimentos de entrega**

**Material a entregar na Fase II:**

**a)** Um relatório escrito do Trabalho-FASE I&II com número de páginas até 30, mais um anexo com o código Python desenvolvido, com letra de tamanho 12, e os seguintes pontos:

1. (\*) Uma capa com a identificação do Trabalho: “Trabalho de Algoritmia e Programação”; título do trabalho; data; Universidade; curso; **turno (Diurno ou Pós-laboral); turma**; número e nome dos alunos/as;

As indicações do turno e turma devem obrigatoriamente aparecer em destaque:

**Turno Diurno, Turma A (ou B) ou**

**Turno em Pós-laboral Turma A (ou B).**

**Caso hajam alunos que pertençam a turnos ou turmas diferentes, no Relatório deve aparecer o turno correspondente ao das suas aulas práticas habituais e a indicação de cada aluno a que turno/turma pertence.**

2. Introdução (problema e objectivos a atingir – pode ser baseado no enunciado);

3. Descrição e explicação das estruturas de dados utilizadas;

4. Descrição sumária do código implementado, indicando o que faz cada um dos módulos (funções / procedimentos);

5. Manual do Utilizador, com imagens dos menus e dos “outputs” com as respetivas explicações;

6. Conclusões, onde deve realçar os aspectos relacionados com a implementação do Problema:

* 1. Dificuldades sentidas e forma de as ultrapassar, se aplicável;
  2. Indicação clara do que foi e não foi feito em relação ao enunciado do Projecto;
  3. Funções e tarefas que cada elemento do grupo desempenhou ao desenvolver este Projecto.

7. Uma listagem do programa fonte, comentado, em anexo (não conta para as 30 páginas).

Esta listagem deve ser com letras com fonte de tamanho 8.

O Relatório deve ser entregue em formato digital.

**O docente das aulas práticas indicará aos alunos se deseja ou não receber uma cópia do Relatório em papel.**

No caso da entrega do Relatório em papel, este deve ser entregue na Sala dos Livros de Sumários, até à sua hora de fecho do dia limite de entrega.

**b)** Por *up-load*, no e-L Moodle de AP, até ao dia limite de entrega:

- O Relatório do Trabalho;

- Uma pasta com o Projecto Python.

**IMPORTANTE:**

O relatório e o projecto Python, devem ser designados segundo a seguinte sintaxe, para a Fase I:

**D**iurno A e B:

AP17DA <RF2|PyF2>\_{<InicialPrimeiroNome><UmNomeDeFamília>}

AP17DB <RF2|PyF2>\_{<\_{<InicialPrimeiroNome><UmNomeDeFamília>}

**P**ós-laboral:

AP16PL\_<RF2|PyF2>\_{<\_{<InicialPrimeiroNome><UmNomeDeFamília>}

Exemplo:

Relatório: AP17DA\_RF2\_NFerreiraDGoncaloJBalagoes

Ou AP17DA\_RF2\_NF&MS&JB (no caso de ser extensa a designação)

Python: AP17DA\_PyF2\_NFerreiraDGoncaloJBalagoes

**Sobre a avaliação do Trabalho:**

Para que os trabalhos sejam avaliados têm de cumprir os seguintes requisitos:

R1 - Deve ser possível compilar e executar o programa;

R2 - O programa terá que cumprir os requisitos de cada FASE, pelo menos parcialmente;

R3 - Os grupos são constituídos por 2 a 4 pessoas;

R4 - O Projecto será desenvolvido em Python (Anaconda3), no ambiente do Spyder.

O Projecto terá avaliação final em prova de apresentação e discussão após a entrega da Fase II, com a presença de todos os alunos de cada Grupo, no período de 16 a 30 de Janeiro de 2018.

**Note bem:**

O Projecto deve ser efectuado em Grupo de 2 a 4 alunos.

Os Projectos devem ser originais.

Cópias de outros programas levarão à anulação do Projecto.

Todos os alunos de cada Grupo devem conhecer bem o código de Python desenvolvido.

**FASE II Parte A**

A FASE II do Trabalho prático de Algoritmia e Programação continua a FASE I, considerando a utilização de ficheiros para guardar os dados necessários.

Considerando a matéria já lecionada, a “FASE II Parte A” diz respeito a Ficheiros-Texto. Seguir-se-á uma Parte B.

O GAU deverá arquivar em Ficheiro-Texto os dados relativos às disciplinas e sua avaliação contínua.

Considere os seguintes requisitos:

1. O Ficheiro-Texto deve conter um cabeçalho que contenha algo semelhante a

“AVALIAÇÃO CONTÍNUA 1.º ANO/1.º SEM <curso>”

e apresentar, segundo um formato esteticamente a gosto dos programadores, as informações decorrentes das várias disciplinas, nomeadamente para cada uma: disciplina, docente, elementos de avaliação, ponderações, datas e classificações.

1. Deve considerar procedimentos ou funções que permitam gravar e afixar o conteúdo do Ficheiro-Texto.
2. Terminadas as modificações decorrentes da inicialização ou actualização de disciplinas e seus dados relacionados, o programa GAU deve proceder à gravação do Ficheiro-Texto.
3. O utilizador deve dispor de uma opção no menu inicial que permita listar o conteúdo do Ficheiro

-Texto.

**Actualização de Datas:**

• Data de publicação do enunciado da Fase II do Projecto: 7 de Janeiro de 2018.

• Data limite de entrega da Fase II do Projecto: 22 de Janeiro de 2018.

**FASE I**

**Descrição:**

O Trabalho prático de Algoritmia e Programação, a implementar com a Linguagem de Programação Python (Anaconda3), intitula-se “GAU - Gestão da Avaliação na Universidade”. Irá ser desenvolvido em duas FASES (FASE I e FASE II).

O GAU permitirá a um aluno gerir a sua avaliação nas várias disciplinas do curso, na componente de Avaliação Contínua, com as seguintes funcionalidades:

* Criar as disciplinas do curso
* Criar o sistema de avaliação para cada disciplina
* Registar as classificações obtidas
* Verificar o seu estado de avaliação
* Obter informações úteis, tais como: datas de avaliação; classificação mínima necessária na última prova
* Gráficos informativos

A FASE I deve implementar um programa que siga as seguintes indicações:

1. Considerar os seguintes dados:

- Curso

- Lista de disciplinas do primeiro ano, primeiro semestre

- Docentes das disciplinas

- Em cada disciplina: Elementos de avaliação, sua ponderação e datas

- Classificações obtidas

SUGESTÕES: Utilize vectores e matrizes, com recurso a listas em Python, para representar a lista de disciplinas, a lista de respectivos docentes, uma lista de listas para a avaliação (os elementos de avaliação, a sua ponderação, as suas datas e a suas classificações).

Exemplo para AP e EDC:

AP:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Teste1 | Teste2 | Trabalho | Presenças | Final |
| 30% | 30% | 35% | 5% |  |
| 20171105 | 20171120 | 2071216 |  |  |
| 8 | 14 | 13 | 1 | 12 |

EDC:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Trab1 | Teste1 | Trab2 | Teste2 | Presenças | Final |
| 10% | 10% | 35% | 35% | 10% |  |
| 20171010 | 20171105 | 20171120 | 2071216 |  |  |
| 15 | 8 | 14 | 13 | 2 | 14 |

Na implementação, comece por atribuir conteúdos a estas listas de listas, localmente no próprio código: Posteriormente, quando o programa estiver a executar-se satisfatoriamente, considere a alternativa de introduzir os dados via teclado.

As primeiras três linhas de AP e EDC são atribuídas logo de início. Na quarta linha as classificações vão-se inserindo ao longo do tempo, quando a classificação correspondente é conhecida, devendo existir uma opção no programa, para este efeito. À medida que as classificações parcelares são atribuídas, a nota Final, vai-se construindo automaticamente com as várias parcelas.

1. Resultados a obter

- Afixar a tabela de avaliação de uma dada disciplina e o nome do docente respectivo;

- Consultar as próximas datas de avaliação (Trabalhos ou Testes);

- Quando só faltar um elemento de avaliação, calcular qual a nota mínima necessária para obter aprovação com 10 valores. Considerar também a hipótese de desejar obter uma determinada nota final (indicada via teclado), e para esta, calcular qual a nota mínima necessária para obter aprovação com esta determinada nota final;

- Apresentar um gráfico de barras verticais com as classificações obtidas em uma disciplina, representando no eixo dos X - os elementos de avaliação, e no eixo dos Y – as notas obtidas;

- Apresentar um gráfico de barras horizontal que compare as classificações finais de todas as disciplinas, representando no eixo dos X – as notas finais e no eixo dos Y – os nomes das disciplinas.