Programação II

Introdução à cadeira

Programação II

- Chamo-me Bruno Loff
- Sou o novo regente da cadeira
- Fui contractado pela faculdade de ciências este ano, e é a minha primeira regência
- O meu website: https://brunoloff.wordpress.com

Porquê aprender a programar?

- Automatização de processos
- Geração de valor
- Extração de informação
- É divertido :-)

A cadeira de programação II no passado

- Fundamentos de estruturas de dados
- Algoritmos fundamentais de ciências dos computadores

A cadeira foi considerada inadequada para os cursos que servia

- A compreensão dos fundamentos em estruturas de dados é essencial para um cientista dos computadores
- Mas não para alguém que se limita a aplicar as linguagens de programação para automatização de certas tarefas.
- O exemplo mais relevante para esta audiência é a **análise de dados**, quer sejam numéricos, económicos, geográficos, em biologia, etc.

Análise de dados / Data analysis

- É uma fracção significativa de toda a programação que é feita hoje em dia.
- Pode ser aplicada em domínios muito variados. Exemplos: em finança, em biologia, em ciências sociais, em medicina, em geografia, em políticas públicas, recursos humanos, marketing, etc, etc, etc
- “Data-driven world”
- É a principal forma de programação feita por pessoas que não são programadores profissionais

Programação II

- O ideal, portanto, seria fazer desta cadeira uma cadeira de análise de dados.

  (Dado que a maioria de vocês são alunos de engenharia geográfica, faria sentido que a cadeira se focasse em dados geográficos.)
- Foi isso que eu pensei fazer quando conceptualizei a cadeira pela primeira vez.
- Mas foi-me feito entender que, infelizmente, com o nível de preparação com que os alunos chegam a esta cadeira, isto não é ainda possível!
- Foi-me dito que vos faltam princípios básicos de programação, e que vos falta o domínio necessário da actividade de programação.
- Porquê?

Porque é que não vamos ter uma cadeira de análise de dados

- A análise de dados é um tema extenso e difícil, que pode ser coberto razoávelmente num só semestre, apenas se os alunos tiverem um domínio pleno de programação.
- As ferramentas de análise de dados (e.g. a biblioteca *pandas* do python) são por si só de grande complexidade, e necessitam de um tratamento aprofundado.
- Não nos queria ver enterrados num mar de complexidade técnica para o qual vocês não estão preparados.

Programação II

- Assim sendo… O objectivo da cadeira será:
  - Treinar programação mais aprofundadamente que no semestre anterior, e desenvolver em vocês alguns princípios básicos da prática.
  - Desenvolver algumas noções básicas de análise de dados.
  - Aprender uma biblioteca simples de visualização de dados.
- Pensei nesta cadeira como uma cadeira profundamente prática. O objectivo é **exercitar** a prática de programação, e não de atingir uma compreensão muito elevada sobre esta actividade, como seria exigido de um especialista.

Avaliação

- A cadeira será avaliada da seguinte forma
- Todas as semanas eu vou passar um trabalho de casa
- Os vários trabalhos de casa resultarão numa série de “mini-projectos” (penso que 3-4, logo veremos).
- Cada pessoa deve resolver os trabalhos de casa **individualmente**.
- Ou seja, pode pedir ajuda aos colegas como quiser, mas depois deve escrever o programa individualmente.
- No fim do ano, vou fazer uma oral, em turnos de 5-10 pessoas de cada vez, para confirmar com cada pessoa que foram feitos **todos** os trabalhos de casa, e que foram de facto feitos pelo próprio.

Critérios de avaliação

- Desde que a funcionalidade dos vários mini-projectos esteja correctamente implementada, e que eu fique 100% convencido que foi implementada pelo próprio aluno, têm 14 pontos garantidos.
- Ademais têm o semestre inteiro para fazer os trabalhos de casa. Devem trabalhar todas as semanas, porque não vos vai sobrar tempo.
- A diferença entre o 14 e o 20 faz-se por:
  - Aplicação correcta de princípios de programação – abstracção, encapsulamento, e separação de conceitos, 3 pontos
  - Organização e documentação do código, 2 pontos, e
  - Capacidade de resolver uma pergunta “extra” durante a oral, 1 ponto.

Copianço

- Faço saber que é extremamente fácil testar se uma pessoa fez o próprio programa que diz ter feito. Basta fazer algumas perguntas simples sobre como funciona, e verificar que a pessoa consegue extender o programa com nova funcionalidade.
- Se o aluno não me convencer que fez o próprio programa, se não me conseguir convencer que compreende o código do programa que me está a apresentar, isso contará como nota negativa. Se for só uma falha aqui e ali, darei o benefício da dúvida, e poderão passar com nota entre 10 e 14.
- Mas em geral, repito o aviso, é **fácil** de perceber quando um aluno não fez o próprio trabalho.
- De resto, serei bastante generoso na atribuição de notas. A ideia é que voces compreendam como fazer programas, não é medir se são programadores absolutamente extraordinários versus programadores meramente razoáveis. Vocês não estão num curso de informática.
- Eu quero que todos os que passem a esta cadeira sejam depois capazes de fazer um programa simples de análise de dados.

Quais serão os mini-projectos?

- Mini-projecto 1 – análise simples de textos (contar palavras, etc).
- Mini-projecto 2 – Biblioteca “bebé” de análise de dados. Cada pessoa nesta cadeira vai fazer uma biblioteca de análise de dados (mini bébé), para perceber os conceitos base, e para utilizar nos outros projectos.
- Mini-projecto 3 – análise e visualização de dados eleitorais. Quem votou em quem, e aonde, nas legislativas portuguesas, ao longo dos anos?
- Mini-projecto 4 – Visualização de dados de incêndios em Portugal?
  O mini-projecto 4 ainda está por decidir, veremos.

Como serão organizadas as aulas

- Quem daqui não tem um computador portátil?
- Quero que TODOS os alunos tragam portátil para a aula, configurado com o pycharm. Os alunos que não têm portátil, sentam-se ao lado dos que têm.
- Nas aulas teóricas eu mostro-vos exemplos de programas (programo à vossa frente, e peço que me imitem).
- Cobrirei alguns princípios muito básicos e fundamentais de programação. Heurísticas. Modos de pensar.
- Nas aulas laboratoriais eu dou-vos exercícios para practicarem por vocês mesmos. Desses exercícios surgem depois os trabalhos de casa, que juntos formarão os mini-projectos.

Comunicação

- Consultar o email da universidade todos os dias (avisos etc vão para lá).
- Consultar as páginas da cadeira:
  - A página do sigarra tem horários, programa, etc
  - A minha página tem tudo o resto: informações, fichas, avisos de datas, *etc:*

    https://brunoloff.wordpress.com/programacao-ii-2020/
- A cadeira tem um **forum online** para resolver dúvidas:

  https://groups.google.com/forum/#!forum/programao-ii-2020--fcup/

  Já enviei um convite para o email universitário.
- Vamos decidir agora o meu horário de dúvidas.

Para a próxima aula (4ª feira)

- ***Preparar o vosso portatil.*** Instalar o python e o pycharm (community edition), e trazer para a proxima aula. Os que sabem fazer ajudam os que não sabem.

  `https://www.jetbrains.com/pycharm/`

- Façam já hoje, não se esqueçam...
- Tem de ter bateria para 1h de aula (duh!).
- Tragam os portáteis também para as aulas de laboratório!
- Quem não tiver portátil fica ao lado de quem tem. Vou tentar gravar a aula de modo a que quem não tem portátil possa fazer depois em casa.

Hoje já **há aula** de laboratório.
14h00-16h00
na sala 1.55 deste edifício.