Programação e Algoritmia 2024-2025 - Introdução

Carlos Tavares

September 24, 2024

Table of Contents

Carlos Tavares (https://www.linkedin.com/in/tavarescarlos)

- ctavares.teaching@gmail.com
- d7039@di.uminho.pt
- Enviem-me e-mails com [Eng.Aero] no cabeçalho

Programação e Algoritmia

O que significam estas palavras?

Algoritmo:

A palavra algoritmo tem sua origem no nome do matemático persa Muhamamad ibn Musa al-Khwarizmi, que viveu no século IX.

al-Khwarizmi => Algoritmi

Al-Kitāb al-Mukhtaṣar fī ḥisāb al-jabr wa'l-muqābala ("O Livro Conciso sobre Cálculo por Completamento e Balanço"),

O livro era sobre a resolução de equações e definia receitas muito precisas para o fazer.

Programação e Algoritmia ii

Receitas => **Algoritmo.** Um algoritmo é exactamente uma **receita** muito precisa para fazer uma tarefa.

Definição mais formal: O algoritmo é uma sequência finita de passos lógicos, completamente não-ambíguos, para realizar uma tarefa

Examplos:

Mudar o pneu do carro, uma receita de culininária, Somar dois números, triagem de Manchester

Algoritmos são um tema central na ciência de computadores e na ciência em geral!

Um algoritmo é algo abstrato! Podemos exprimi-lo através de uma infinidade de linguagens.

Normalmente queremos que sejam computadores a executar os nossos algoritmos.

Os computadores só "dialogam" através de programas (sequências de instruções) expressos em linguagens próprias capazes de lidar com os seus componentes (memória, periféricos). Estas linguagens denominam-se **linguagens de programação**.

Um pequeno exemplo: Como encontrar o nome de uma pessoa numa lista eleitoral?

- Se a lista não estivesse ordenada ?
- Aproveitando o facto da lista estar ordenada ?

O curso

Objectivos

Ser capaz de resolver problemas simples utilizando computadores:

- Aprender a **"programar"** (utilizar uma linguagem de programação para exprimir programas correctos)
- Conhecer estratégias algoritmicas e estruturas de dados comuns para resolver problemas computacionais

A linguagem de programação utilizada vai ser o **Python**!

Estrutura do curso

- **Módulo 1** - Introdução ao Python, funções, matemática e input/output básico (Setembro)
  - Avaliação: ~~8 Outubro~~ 14 Outubro
- **Módulo 2** - Estruturas de controlo: instruções **if ... then ... else**, funções recursivas, ciclos **while** e **for**. Estruturas de dados: tuplos (Outubro)
  - Avaliação: 5 Novembro
- **Módulo 3** - Estruturas de dados: listas, strings, dicionários e conjuntos. Definições em extensão, funções de ordem superior e programação "preguiçosa" (Outubro e Novembro)
  - Avaliação: 3 Dezembro
- **Módulo 4** - Pesquisa força-bruta, pesquisa em espaços ordenados, programação dinâmica (Novembro)
  - Avaliação: 17 Dezembro
- **Módulo 5** - Objectos, Ficheiros, Engenharia de Software (Dezembro)

Como irá funcionar o curso? i

2 hrs **Aulas Teóricas (T)** (Segunda-feira)
2 hrs **Aulas Teórico-Práticas (TP)** (Terça-feira)

Aulas Teórico-praticas - Obrigatória a presença em 70% (máximo 5 faltas).

Aulas Teóricas - Presença não obrigatória, no entanto há uma bonificação de 0,1 pela presença (máximo de 1,5v). No entanto, depende da nota final *NF*:

- Se *NF* <= 8: Bonificação total
- Se *NF* > 8 e < 18: Bonificação * 2/3
- Se *NF* >= 18: Sem bonificação

Como irá funcionar o curso? ii

Avaliação:

$$\text{Nota final} = A_1 * 0,15 + A_2 * 0,35 + A_3 * 0,35 + A_4 * 0,15$$

- A_1 - 1ª avaliação (sem nota mínima).
- A_2 - 2ª avaliação (nota mínima: 9,5)
- A_3 - 3ª avaliação (nota mínima: 9,5)
- A_4 - 4ª avaliação (sem nota mínima)

Ou

Exame:

$$\text{Nota final} = \text{Nota do Exame} \tag{1}$$

O exame só é acessível a quem tiver frequência, ou justificação legalmente aceite pela Universidade do Minho, e.g. doença.

Alguns conselhos...

É recomendado o uso de portátil/tablet durante as aulas teóricas. "Escuta activa", tentar replicar o que está a ser feito nas aulas teóricas

As aulas teórico práticas devem ser utilizadas para fazer os exercícios

Não é obrigatório ter portátil para fazer o curso, as avaliações são todas escritas.

É recomendado treino.

Recursos e Bibliografia

Principal:

- Lott, Steven F. "Building skills in python." Creative Commons, USA (2010).
- Guttag, John V. "Introduction to computation and programming using Python". Mit Press, 2013.

Auxiliar:

- Elkner, Jeffrey, Allen B. Downey, and Chris Meyers. "How to Think Like a Computer Scientist: Learning with Python Documentation.". 2010.
- Mertz, David. Functional Programming in Python. O'Reilly Media, 2015.

Resources and Bibliography

Alguns sites de treino em programação

Hacker rank: https://www.hackerrank.com

Leetcode: https://leetcode.com

Euler project: https://projecteuler.net

ChatGPT: https://chat.openai.com

Motivação

Porquê aprender programação?

A computação e a engenharia aeroespacial têm uma relação muito próxima desde o inicio!

Desde os anos 50 que os computadores são utilizados para o planeamento de missões (i.e. cálculo de trajectórias)

Mais tarde tornaram-se também importante na simulação e projecto dos veículos aeroespaciais e também nos sistemas de controlo das naves (a chamada aviónica).

Software no aero-espaço

video

Space Shuttle (1981-2011) – 400 mil linhas de código

(A person can do 100 lines of **good** code per day.)

Software no aero-espaço

O controlo da Apollo 11 foi feito por um computador.

Figure 1: Margaret Hamilton, NASA

Software disponível em:
https://qz.com/726338/the-code-that-took-america-to-the-moon-was-just-published-to-github-and-its-like-a-1960s-time-capsule

Software no aero-espaço

Hoje em dia o controlo de todo o tipo de aeronaves é em grande parte feito por computador (aviónica, pilotos automáticos):

Lines of code in aerospace applications:

- Hubble space telescope (1990) – 2 million;
- US Airforce F-22 Raptor (1996) – 1.7 million;
- Boeing 787 Dreamliner (2007) – 13.5 million;
- US Airforce F-35 Joint Strike Fighter (2008) – 23.4 million;
- Mars curiosity Rover (2011) – 5 million;
- US Military Drone (?) - 3.2 million

No futuro espera-se que o papel da computação seja cada vez mais preponderante na aviação e exploração espacial, e.g. aumento da autonomia das naves através de inteligência artificial.

O reverso da medalha i

Erros "famosos" de software:

(2018 e 2019) - Boeing 737 MAX - Dois acidentes $\sim$ 600 mortos

O reverso da medalha ii

(1996) - Foguetão Arianne 5 explode, levando consigo vários milhões em carga, por causa de uma má conversão de números.

(2014) - Voo da Malasian Airlines desaparece no mar de Java, devido aquilo que se julga ter sido um erro no piloto automático.

A engenharia aeroespacial tem o grau de criticidade máximo: falhas nos componentes podem significar perdas de vidas e/ou custos financeiros muito elevados. Os processo de desenvolvimento, testes e certificações, são extensivos, rigorosos e caros.

Questões