Temario de Algoritmos Computacionales

Diego Alberto Barceló Nieves
Facultad de Ciencias
Universidad Nacional Autónoma de México

El siguiente temario está basado en el temario oficial del curso. Al elaborarlo, asumimos que quienes tomarán el curso no tienen experiencia previa con programación, pero sí tienen bases teóricas sólidas de álgebra superior y cálculo diferencial e integral de una variable, así como nociones básicas de ecuaciones diferenciales ordinarias, pues esto último será necesario en el módulo 3. Para los módulos 1, 2 y 3 utilizaremos el lenguaje de programación Julia. La duración aproximada de cada módulo se indica en paréntesis.

0. Introducción a la programación (2 semanas)

1. ¿Qué es un programa? (Paradigma imperativo de la programación)
2. ¿Cómo se ejecuta un programa? (Lenguaje de programación, código fuente y sintáxis, comentarios y mensajes de error)
3. Licencias: legalidad y ética. (*Software* de código abierto/cerrado y *software* privativo/libre)
4. ¿Cómo escribo y ejecuto un programa? (Editor de texto y terminal virtual, REPLs e IDEs)
5. ¿Cómo aprendo a programar? (Manuales, documentación y foros de preguntas)
6. Herramientas útiles para hacer programación. (Jupyter y Pluto, Git y GitHub/GitLab)

1. Estructura básica de la programación (6 semanas)

1. Operaciones aritméticas y tipos de datos numéricos. (Precedencia y asociatividad de operadores)
2. Sistemas numéricos de punto flotante y error numérico. (Épsilon de máquina y propagación de errores)
3. Operaciones lógicas y valores Booleanos.
4. Tipos de datos de texto, arreglos, variables y conversión de tipos implícita.
5. Algoritmos, diagramas de flujo y control de flujo.
6. Declaraciones condicionales. (Declaraciones *if*, *else* y *elseif*)
7. Ciclos iterativos. (ciclos *while* y *for*, rangos y arreglos por comprensión)
8. Funciones y ciclos recursivos.
9. Estructura de la programación modular. (Bibliotecas)

2. Representaciones visuales (2 semanas)

1. Visualización de datos con Plots.
2. Gráficación de funciones y animaciones con Plots.
3. Manipulación de imágenes digitales con JuliaImages.

3. Cómputo científico (5 semanas)

1. Métodos numéricos. (Estabilidad y convergencia).
2. Solución de sistemas lineales de ecuaciones algebráicas. (Método de eliminación Gaussiana)
3. Aproximación de raíces. (Método de Newton)
4. Solución de ecuaciones diferenciales ordinarias. (Método de Euler)
5. Caminante aleatorio.

4. Introducción a otros lenguajes de programación (1 semana)

1. Cómputo científico con GNU Octave como alternativa a MATLAB y Mathematica.
2. Análisis estadístico con R como alternativa a SAS.
3. Animaciones programáticas en Python con Manim (introducción a programación orientada a objetos).

Bibliografía recomandada para el curso

1. Lauwens y Downey, *Think Julia: How to Think Like a Computer Scientist* (2019).
2. Burden, Faires y Burden, *Numerical Analysis* (2016).
3. Cormen, Leiserson, Rivest y Stein, *Introduction to Algorithms* (2009).
4. Cairó, *Metodología de la programaciónñ: Algoritmos, diagramas de flujo y programas* (2003).

Bibliografía complementaria

1. Documentación de Julia.
2. Documentación de Jupyter.
3. Material del curso "Introduction to Computational Thinking" del MIT.
4. Blum y Bresnahan - *Linux Command Line and Shell Scripting Bible* (2015).