Departamento de Matemática

PENSAMIENTO COMPUTACIONAL Y PROGRAMACIÓN
PROGRAMA

I. Descripción del curso

El electivo Pensamiento Computacional y Programación (PCP) pretende que los estudiantes desarrollen las capacidades necesarias para resolver diferentes problemas mediante la programación de computadores. Para lograrlo los estudiantes aprenderán a crear algoritmos y modelos que podrán ejecutar utilizando el lenguaje de programación Python v3.

II. Objetivos

1. Explicar conceptos básicos relativos a un programa computacional tales como algoritmos, variables, expresiones, control de flujo, funciones, listas, strings, clases y objetos.

2. Aplicar técnicas fundamentales para la resolución de diversos problemas con ayuda del computador, como identificar datos relevantes, las relaciones entre ellos, modelar y descomponer problemas grandes en varios problemas más pequeños.

3. Aplicar el razonamiento algorítmico para generar la solución a un problema como una secuencia de pasos bien definidos, incluyendo pasos condicionales, repetición de pasos, llamadas a funciones, y recursión.

4. Llevar a cabo el proceso de desarrollo de programas, escribiendo y depurando programas, usando el lenguaje de programación Python v3.

5. Utilizar PyCharm como entorno de desarrollo de software para escribir, compilar y depurar programas.

6. Escribir programas que permitan realizar análisis estadísticos y geométricos.

7. Desarrollar aplicaciones para dispositivos móviles o dispositivos provistos de sensores y mecanismos de control.

III. Contenidos

1. Lógica

    - Lógica matemática y computacional
    - Operadores lógicos

2. Introducción a los algoritmos

    - Concepto de algoritmo
    - Modelamiento de situaciones mediante algoritmos

3. El lenguaje de programación Python

    - Qué es Python
    - Comandos y programas básicos

4. Variables, expresiones y operadores

    - int, float, bool, y str
    - +, -, *, /, //, %, **
    - Operadores booleanos: and, or, not
    - Comparación con ==, !=, <,<=, >,>=
    - Input/Output básico por consola

5. Control de flujo

    - if, elif, else
    - Loops (while y for)

6. Funciones

    - Creación de funciones propias
    - Parámetros por defecto en funciones
    - Utilización de funciones de otros módulos como math o random

7. Strings

- Declaración y manejo de strings
- Tipo chr y tabla ASCII
- Conseguir el caracter de la posición i
- Slices
- for sobre strings
- Concatenación de strings
- Métodos como find, upper, lower, strip, split, join, x in string

8. Listas

- Conseguir el elemento de la posición i
- Slices
- for sobre listas
- Concatenación de listas
- Métodos como append, pop, insert, remove, x in lista
- Manejo de la idea de "lista de listas"
- Búsqueda en listas

9. Archivos

- Guardar datos y ruta de archivo
- Modos r y w

10. Programación orientada a objetos. Clases.

- Conceptos de "clase" e "instancia de una clase"
- Definición de una clase, con métodos y atributos
- Creación de instancias de una clase
- Dunder methods como __str__

11. Ordenación y búsqueda

- Bubble sort, Insertion sort

12. Recursión
    - Definición de funciones recursivas
    - Quicksort, Permutación, Mergesort

13. Simulación
    - Para qué simular

14. Taller de estadística
    - Análisis de datos COVID
    - Análisis de sentimiento en twitter sobre COVID

15. Taller de desarrollo de apps
    - Flappy Bird

IV. Metodología

- Clases expositivas
- Participación en clases
- Discusión de casos prácticos
- Trabajo grupal para realización de tareas
- Charlas/Visitas*

V. Evaluación

- Laboratorios 30%
- Tareas: 50%
- Participación: 20%

(*) Estas actividades son tentativas, pues dependen de la situación sanitaria nacional, disponibilidad de los expositores y otros imprevistos.

Tareas (50%)

Se realizarán dos tareas semestralmente. Estas son de carácter **individual** y pueden desarrollarse usando apuntes y material disponible en internet. Para cada tarea se dispondrá entre 2 y 3 semanas para su entrega. Considerando este plazo, no se aceptarán excusas para no entregar en el periodo establecido.
La copia/plagio se castigará con nota 1.1 a todos los involucrados.
Se recibirán tareas con hasta tres días de atraso, sin embargo, cada 24 horas sobre la fecha de entrega oficial, se descontará un punto a la nota obtenida. Pasadas las 72 horas de atraso, la nota de tarea será 1.1

Laboratorios (30%)

Un laboratorio es una instancia de trabajo en clases desarrollado en papel o computador. El trabajo es individual y puede realizarse usando apuntes. Al finalizar un laboratorio cada estudiante deberá entregar un documento (papel) y/o archivo (pc) que contenga su desarrollo y respuestas para ser evaluado.

Participación (20%)

La nota de participación se obtiene acumulando puntos de participación (pp), donde 70 puntos equivalen a un 7.0 (Cada pp corresponde a una décima de la nota de participación a contar de los 10 pp – por ejemplo, 10pp = 1.0, 20 pp = 2.0, ⋯).

Se obtienen pp al contribuir a la clase, demostrar un comportamiento ejemplar y realizar tareas y desafíos. Durante cada semestre todo estudiante tendrá la oportunidad de acumular hasta 75 pp y podrá disponer de ellos como estime conveniente, estos son transferibles y acumulables. Los puntos se distribuyen de la siguiente manera:

- Actividades y desafíos propuestos en clases – 45 pp
- Participación en clase (preguntas, aportes, disposición) – 15 pp
- Puntualidad, aporte a un buen clima de aula, actitud de trabajo – 15 pp

VI. Bibliografía

- Python software foundation, Python v3 Documentation, http://docs.python.org/3/

- Ceder. The quick python book. Manning Publications Co., 2010.

- Downey, B. Think Python: How to think like a computer scientist. Green Tea Press, 2013.

- Zelle, J.M. Python programming: An introduction to computer science. Franklin, Beedle & Associates, Inc., 2nd edition, 2010.

Web:

http://www.greenteapress.com/thinkpython/
http://interactivepython.org/courselib/static/thinkcspy/index.html