

Programación

- Número de créditos: 10
- Semestre: 1
- Horas a la semana: (presenciales y de trabajo al estudiante)
- Teoría: 4
- Práctica: 2
- Autoestudio:
- Requisitos: NA
- Clave:
- Asignatura: Etapa Básica
- Materia asociada a la Línea de investigación:

Descripción de la asignatura: El objetivo de este curso es introducir al estudiante a la programación mediante el planteamiento de problemas y manejo de algoritmos y acercarlo a los lenguajes de programación a través de una introducción de python.

Contenido: Introducción a python, numpy, Matplotlib, scipy, sympy, pandas

Índice temático:

- **Introducción a la programación**

Programación de computadoras. Definición de algoritmo y programa. Lenguajes de programación. Fases de la programación. Análisis de un problema. Diseño de un algoritmo. Implementación de un programa.

- **Algoritmia**

Algoritmos y sus propiedades. Estructura. Representación. Conceptos fundamentales. Tipos de datos. Literales. Variables. Constantes. Identificadores. Entrada y salida de datos. Expresiones. Operador de asignación. Operadores aritméticos. Prioridad y asociatividad de operadores.

- **Estructuras de programación**

Estructura secuencial. Estructura de selección. Condiciones: operadores relacionales y lógicos (Simple, doble, anidada, múltiple). Estructura de repetición (evaluación previa, evaluación posterior, anidada). Tipos de ciclos (controlados por contador, controlados por suceso)

- **Introducción a Python (16 sesiones)**

Tipo de datos, control de flujo, funciones, excepciones, clases, módulos

- **Buenas prácticas de desarrollo (4 sesiones)**

escribir, probar y depuración; y optimizar de programas

- **Matplotlib y Graficado (8 sesiones)**

pyplot, propiedades de línea, texto, multiples figuras y ejes, polares y contorno

Bibliografía:

- Joyanes Aguilar, L. (2008). Fundamentos de programación (4ta. ed.). España: McGraw-Hill.
- Clua Führer, Jan Erik Solem, Iliver Verdier, *Scientific Computing with Python 3* (2016), Packt Publishing
- John V. Guttag, Introduction to Computation and Programming Using Python with Application to Understanding Data (2016), MIT Press.
- Hans Petter Langtangen, *A Primer on Scientific Programming with Python* (2014), Springer
- Ivan Idris, *NumPy Beginner's Guide, Second Edition* (2013). Packt Publishing.
- Allen Chi Shing Yu, Claire Yik Lok Chung, Aldrin Kay Yuen Yim, *Matplotlib 2.x by examples* (2017), Packt Publishing.

Planeación Educativa

Competencias a Desarrollar:

El estudiante adquirirá la capacidad de crear algoritmos de manera eficiente para diferentes problemas, así como el poder codificarlos en el lenguaje script Python.

El profesor promoverá el trabajo fuera del aula, mediante tareas las cuales tendrán que ser escritas para posteriormente revisarlas en la sesión de talleres. De esta forma los alumnos desarrollaran su capacidad de comunicar sus ideas de forma escrita y oral.

Mediante la sesión de talleres, el alumno aprenderá a discutir con sus compañeros sus ideas y diferentes soluciones para un problema, lo cual fomentará el trabajo en equipo, el respeto y la tolerancia.

Generales:

El estudiante desarrollará capacidades de; abstracción, análisis y síntesis, organización y planificación del tiempo, habilidad para trabajar de forma autónoma y al mismo tiempo el trabajo en equipo, habilidad para transmitir sus conocimientos de forma oral y escrita, creatividad, identificación y planteamiento de problemas, motivar y conducir hacia metas comunes, respeto por la diversidad y multiculturalidad, compromiso con la calidad.

Específicas:

Dominio de los conceptos básicos de la computación científica a través del uso de diferente software científico. Capacidad para expresarse correctamente. Capacidad de abstracción incluido el desarrollo lógico. Capacidad para comprender problemas. Capacidad para presentar sus razonamientos de forma apropiada para sus pares académicos, tanto oralmente como por escrito. Capacidad para detectar inconsistencias.

Matriz Educacional			
Resultados del Aprendizaje	Actividades Educativas	Horas	Evaluación
		Teóricas, Prácticas, autoestudio, y numérico-computacionales	
1. Introducción a la programación	Asistencia a clases, tareas a casa, clases prácticas	Teóricas 2 Prácticas 4 Autoestudio 4	Proyecto de elementos de programación
2. Algoritmia	Asistencia a clases, tareas a casa, clases prácticas	Teóricas 2 Prácticas 4 Autoestudio 4	Proyecto y examen sobre algoritmos: Diagrama de Flujo y Pseudocódigo
3. Estructuras de Programación	Asistencia a clases, tareas a casa, clases prácticas	Teóricas 2 Prácticas 4 Autoestudio 4	Proyecto pseudocódigos estructurados y diagramas de flujo
4. Introducción a Python	Asistencia a clases, tareas a casa, clases prácticas	Teóricas 2 Prácticas 4 Autoestudio 4	Examen problemas científicos

Matriz Educacional			
			numéricos
5. Buenas prácticas de desarrollo	Asistencia a clases, tareas a casa, clases prácticas	Teóricas 2 Prácticas 4 Autoestudio 4	Proyecto y Examen
6. Matplotlib y graficado	Asistencia a clases, tareas a casa, clases prácticas	Teóricas 2 Prácticas 4 Autoestudio 4	Proyecto