GOBIERNO DE LA CIUDAD DE BUENOS AIRES

INSTITUTO DE FORMACIÓN TÉCNICA SUPERIOR (IFTS) Nº 4

PROGRAMA DE LA ASIGNATURA: TÉCNICAS DE PROGRAMACIÓN

AÑO: Primer Año

APELLIDO Y NOMBRE PROFESOR: PÉREZ, Luis

AÑO: 2022

CARACTERIZACIÓN GENERAL DE LA ASIGNATURA/CARGO:

FUNDAMENTACIÓN

La materia parte de la necesidad educativa y social de actualizarse en los saberes tecnológicos. En este sentido la asignatura busca brindarles los conocimientos de resolución de problemas mediante algoritmos y su implementación por computadora mediante la programación. Para lograr una formación integral en sistemas computacionales se deben conocer los términos científicos propios del área, sin dejar de lado los conocimientos básicos de lógica que son las bases de entendimiento y desarrollo de software.

El alumno valora los aprendizajes significativos que suman a sus saberes previos y los avances tecnológicos que forman partes de día a día y del futuro. Se promueve el uso de las redes con sentido comunicativo, tecnológico y social, ofreciendo a los estudiantes nuevas oportunidades en la realización de prácticas, con carácter significativo y relevante.

PROPÓSITO

El propósito general de la asignatura es adquirir los saberes, conocimientos y habilidades de la resolución de problemas algorítmicos, y su implementación mediante un lenguaje de programación adecuado. Se busca fomentar la escritura de un código legible, bien documentado y de fácil mantenimiento.

Se organizan los contenidos en tres unidades: “Teoría de programación y lenguajes”, “Técnicas de programación” y “Resolución de problemas”.

La unidad “Teoría de programación y lenguajes” se centra en la formación teórica de programación, lenguajes, compiladores e intérpretes. Concepto de algoritmo y resolución algorítmica de problemas.

La unidad “Técnicas de programación” se centra en la formación práctica de resolución de problemas algorítmicos y su codificación en el lenguaje adecuado. Se busca que los estudiantes adquieran hábitos de validación y documentación de sus algoritmos y programas.

La unidad “Resolución de problemas” se centra en los conceptos teóricos y prácticos de la resolución de problemas mediante algoritmos. Se ven las estructuras de control del programa, condicionales y ciclos. Tipos y estructura de datos. Declaración y uso de funciones y subrutinas. Técnicas para la escritura de algoritmos eficientes.

Esta organización en unidades no se desarrolla en ese orden secuencial. La planificación y secuenciación se realiza teniendo en cuenta los objetivos de aprendizaje, y planteando el abordaje y resolución de problemas propios de los contenidos abordados.

OBJETIVOS GENERALES

Se espera que al finalizar el cursado de la asignatura, los estudiantes sean capaces de:

- Conocer los fundamentos de algoritmo y programa.
- Conocer los fundamentos de programación estructurada y estructuras de datos.
- Realizar programas como método de resolución de problemas.
- Diseñar algoritmos modulares para la resolución de problemas.
- Implementar los algoritmos diseñados en el lenguaje de programación utilizado.
- Diseñar y realizar pruebas para la validación de algoritmos y programas.
- Documentar actividades de análisis, definición de algoritmos y programas, implementación y prueba conforme criterios técnicos y de calidad

CONTENIDOS

Unidad “Teoría de programación y lenguajes”:

Concepto de algoritmo, resolución algorítmica de problemas. Algoritmos fundamentales, algoritmos numéricos simples. Ambientes de programación. Uso de librerías y APIs (interfaz de programación de aplicaciones).

Programas generadores de código. Concepto de lenguaje de alto nivel y la necesidad de traducción, comparación entre compiladores e intérpretes, aspectos de la traducción dependientes y no dependientes de la máquina. Lenguajes intermedios, asuntos de seguridad que surgen al ejecutar código en una máquina diferente. Máquinas virtuales, concepto, jerarquía de máquinas virtuales.

Unidad “Resolución de problemas”:

Estructuras de control condicionales y ciclos. Instrucciones de entrada/salida. Estrategias de diseño. Elementos de complejidad de algoritmos.

Lenguaje de programación: Estructura sintáctica de un programa en el lenguaje de aplicación. Reglas sintácticas del lenguaje.

Estructuras fundamentales, variables, tipos, expresiones y asignaciones. Representación de datos numéricos, rango, precisión y errores de redondeo. Arreglos. Representación de datos de caracteres, listas y su procesamiento. La elección de una estructura de datos adecuada.

Declaración y llamada de funciones, pasaje de parámetros. Sintaxis de procedimientos y funciones. Funciones matemáticas iterativas y recursivas, funciones recursivas simples.

Programación modular: Concepto. Descomposición estructurada. Aplicación: estructura de un programa utilizando procedimientos y funciones.

Búsqueda sucesiva, binaria y de ordenamiento. Algoritmos de camino mínimo.

Reglas para escribir algoritmos eficientes Elaboración de "algoritmos-tipo" o estándar a partir de métodos lógico-matemáticos.

Programación modular: Concepto. Descomposición estructurada.

Unidad “Técnicas de programación”:

Estrategias de implementación y depuración. Pruebas de escritorio para validar algoritmos y programas. Verificación unitaria de unidades de código, concepto de cubrimiento, organización, ejecución y documentación de la prueba.

Prácticas Formativas:

Las prácticas formativas se organizan e implementan siguiendo un criterio de complejidad creciente. La enseñanza de lenguajes de programación se abordará relacionándolos con las estructuras de datos y los algoritmos aplicados e implementados. Se introduce tempranamente el concepto de procedimiento para llegar finalmente al armado de algoritmos eficientes y de calidad. Se construyen pruebas que validen la corrección del algoritmo.

PLANIFICACIÓN

Unidad	Semana	Objetivos de aprendizaje	Contenidos	Actividades
Teoría de programación y lenguajes	1	Comprender el concepto de algoritmo. Reconocer los requisitos de un algoritmo (sec. de pasos, descripción formal, exploración acotada). Reconocer diferencia entre problemas resolubles y no resolubles por algoritmos	Concepto de algoritmo, resolución algorítmica de problemas. Algoritmos fundamentales, algoritmos numéricos simples. Ambientes de programación. Uso de librerías y APIs (interfaz de programación de aplicaciones).	Identificar algoritmos. Explicar y graficar los algoritmos de las operaciones matemáticas básicas
Resolución de problemas	2	Reconocer y utilizar, condicionales, ciclos condicionales y ciclos de cantidad finita de repeticiones. Seleccionar la/las estructuras requeridas para la solución de un problema simple.	Estructuras de control condicionales y ciclos. Instrucciones de entrada/salida. Estrategias de diseño, de implementación, de depuración. Pruebas de escritorio para validar algoritmos. Elementos de complejidad de algoritmos. Verificación unitaria de unidades de código, concepto de cubrimiento, organización, ejecución y documentación de la prueba. Lenguaje de programación: Estructura sintáctica de un programa en el lenguaje de aplicación. Reglas sintácticas del lenguaje. Reglas del lenguaje.	Realizar algoritmos para la resolución de problemas que requieran condicionales y ciclos. Reconocer y utilizar ciclos con cantidad fija de iteraciones y ciclos condicionales
Técnicas de programación (practica)	3	Ejercitación		Realizar el algoritmo t codificarlo. Validar el algoritmo y programa con juego de datos.
Resolución de problemas	4	Reconocer y seleccionar los tipos de datos atómicos adecuados para el problema. Incorporar las operaciones matemáticas y lógicas, entre datos.	Estructuras fundamentales, variables, tipos, expresiones y asignaciones. Representación de datos numéricos, rango, precisión y errores de redondeo.	Realizar algoritmos, y codificarlos, que requieran diferentes tipos de datos. Seleccionar los tipos de datos adecuados para cada problema
Resolución de problemas	5	Apropiarse de estrategias para la resolución de problemas algorítmicos.	Declaración y llamada de funciones y pasaje de parámetros. Programación modular: Concepto. Descomposición estructurada. Aplicación: estructura de un programa utilizando procedimientos y funciones. Sintaxis de procedimientos y funciones.	Descomponer algoritmos en partes funcionales. Codificar funciones y validarlas mediante juegos de datos conocidos.
	6	Apropiarse de estrategias para la resolución de problemas mediante funciones y procedimientos.	Funciones matemáticas recursivas, funciones recursivas simples.	Resolver problemas mediante algoritmos iterativos y/o recursivos
Técnicas de programación (practica)	7	Ejercitación		Realizar el algoritmo, y codificarlo, utilizando las técnicas vistas
Resolución de problemas	8	Reconocer y seleccionar las estructuras de datos requeridas para la resolución de un problema	Arreglos. Representación de datos de caracteres, listas y su procesamiento. La elección de una estructura de datos adecuada.	Reconocer estructuras de datos. Seleccionar la estructura de datos adecuada para los problemas planteados.
Resolución de problemas	9	Implementar las estructuras de datos en la	Búsqueda sucesiva y binaria y de ordenamiento.	Realizar algoritmos de búsqueda y

		resolución de un problema.	Algoritmos de camino mínimo.	ordenamiento de un vector.
Técnicas de programación (practica)	10	Ejercitación		Resolver y codificar problemas que requieran el uso de las técnicas vistas.
Teoría de programación y lenguajes	11	Reconocer las caracteristicas de lenguajes de alto y bajo nivel, lenguajes compilados e intérpretes. Conocer los conceptos básicos de lenguajes orientado a objetos, lenguajes orientados a eventos.	Programas generadores de código. Concepto de lenguaje de alto nivel y la necesidad de traducción, comparación entre compiladores e intérpretes, aspectos de la traducción dependientes y no dependientes de la máquina. Lenguajes intermedios, asuntos de seguridad que surgen al ejecutar código en una máquina diferente. Máquinas virtuales, concepto, jerarquía de máquinas virtuales.	Explorar las diferencias, ventajas y desventajas de los lenguajes compilados e intérpretes.
Resolución de problemas	12	Utilizar el modelo funcional como herramienta para la resolución de problemas.	Reglas para escribir algoritmos eficientes Elaboración de "algoritmos-tipo" o estándar a partir de métodos lógico-matemáticos.	Trabajo final. Dividir el problema en funciones para su resolución. Realizar los algoritmos correspondientes. Realizar la codificación de los algoritmos. Validar los algoritmos y programa mediante una prueba de escritorio. Documentar los algoritmos y código.
Técnicas de programación (practica)	13	Ejercitación		Trabajo final. Dividir el problema en funciones para su resolución. Realizar los algoritmos correspondientes. Realizar la codificación de los algoritmos. Validar los algoritmos y programa mediante una prueba de escritorio. Documentar los algoritmos y código.
Resolución de problemas	14	Utilizar el modelo dinámico como herramienta para la resolución de problemas.	Programación modular: Concepto. Descomposición estructurada.	Trabajo final. Dividir el problema en funciones para su resolución. Realizar los algoritmos correspondientes. Realizar la codificación de los algoritmos. Validar los algoritmos y programa mediante una prueba de escritorio. Documentar los algoritmos y código.
Técnicas de programación (practica)	15	Ejercitación		Trabajo final. Dividir el problema en funciones para su resolución. Realizar los algoritmos correspondientes. Realizar la codificación de los algoritmos. Validar los algoritmos y programa mediante una prueba de escritorio. Documentar los algoritmos y código.
	16	Evaluación		

METODOLOGÍA

Exposición del docente, dialogo y roll play.

Trabajos prácticos por computadora utilizando guías preparadas para el aprendizaje sistemático de los conocimientos teóricos y prácticos.

Simulacros y trabajos en el campo de Sistemas informáticos.

Cumplir con el 75% de asistencia a las clases presenciales o sincrónicas.

MODALIDAD DE EVALUACIÓN

Trabajos Prácticos parciales de elaboración individual y grupal, se tendrá en cuenta:

Aplicación de conocimientos vistos, aporte personal en el trabajo en equipo, propuesta creativa y original frente a la tarea, y entrega en tiempo y forma.

Examen Parcial integradores por cuatrimestre.

Examen Final integrador.

BIBLIOGRAFÍA

- https://tutorial.python.org.ar/en/latest/index.html
- https://www.tutorialpython.com/
- https://pythones.net/
- https://wiki.python.org/moin/SpanishLanguage
- Ian Sommerville. **Ingenieria de software**, 9na edición. Addison Wesley. 2011
- Luis Rodríguez Ojeda. **Python Programación**, versión 3.0. Escuela Superior Politécnica del litoral. 2017
- Apuntes del docente.