

**TÉCNICO SUPERIOR EN DESARROLLO DE SOFTWARE**

**Unidad Curricular:** Programación I

**Ubicación en el Diseño Curricular:**

**Formato curricular:**

**Régimen del cursado:**

**Carga horaria semanal:**

**Profesora:**

**Ciclo lectivo 2020**

**Plan:** Res. Ministerial Nº 2120/16

**Fundamento:**

Esta unidad curricular permite al estudiante conocer y aplicar alguno de los paradigmas de programación y llevarlo a la práctica mediante la codificación en un lenguaje de programación, desarrollando su capacidad de abstracción. Podrá revisar y corregir programas dados y resolver diversos tipos de problemas comenzando por especificar su propia comprensión de la asignación, diseñar una solución, codificar e integrar partes de código utilizando el ambiente de programación indicado, documentándola de acuerdo a buenas prácticas y realizar Ia verificación unitaria de lo realizado.

**Propósitos:**

* Lograr que el estudiante identifique los requerimientos en un enunciado.
* Desarrollar la habilidad de crear algoritmos en diagramas y luego poder escribirlo en código de un lenguaje.
* Desarrollar la capacidad de la investigación y la adaptación de código para resolver problemas cotidianos en programación. La importancia de estar actualizados. Conocer los foros de consulta de programadores,

**Objetivos:**

* Brindar una introducción a las herramientas metodológicas necesarias para el desarrollo de software.
* Lograr que el estudiante adquiera aptitud en la resolución de problemas a través del desarrollo de algoritmos.
* Que el estudiante logre autonomía y pueda explorar en forma independiente las posibilidades que ofrecen los distintos lenguajes.

**Contenidos:**

**Unidad 1. Algoritmos**

Algoritmos. Robustez de un algoritmo. Correctitud de un algoritmo. Completitud de un algoritmo. Eficiencia y Eficacia de un algoritmo. Resolución de problemas. Diagramas de flujo. Pseudocódigo. Diseño de algoritmos y comprobación su validez. Compresión de las características más importantes de los algoritmos. Utilización de diagramas de flujo como forma gráfica de representar una situación real. Lenguaje C. UML. Interés y comprensión en las representaciones gráficas como forma de resolver problemas reales.

**Unidad 2. Programas**

Constantes. Variables. Definición de Variables. Reservando memoria. Tipos de datos. Inicializar una variable. Asignación de un valor. Palabras reservadas o claves. Expresiones y Sentencias. Operadores. Aplicaciones en Pseudocódigo. Conocimiento de los distintos tipos de datos. Resolución de diferentes problemas de la vida real utilizando pseudocódigo. Lenguaje C. UML. Interés y valoración de la realización de programas en Pseudocódigo.

**Unidad 3. Estructuras de Control**

Estructuras de control. Partes de una estructura de control. Sentencias o instrucciones. Estructuras de selección if. Estructuras de selección if / else. Estructuras condicionales Anidadas. Estructura While. Estructura For. Estructura Do While. Ejercicios de Aplicación en Diagramas de Flujo y Pseudocódigo. Elaboración de algoritmos utilizando las estructuras elegidas. Lenguaje C. UML. Valoración de la importancia de la correcta selección de una estructura de programación.

**Unidad 4. Vectores, Matrices.**

Concepto. Dimensión. Notación de un vector. Inicialización de vectores y matrices. Ejercicios de Aplicación en Diagramas de Flujo y Pseudocódigo. Resolución de ejercicios de aplicación con vectores y matrices. Aplicación de los conceptos a problemas reales. Lenguaje C. Interés y comprensión sobre el correcto uso de vectores y matrices.

**Unidad 5. Programación Orientada a Objetos.**

Concepto de clase, objeto, propiedades, atributos, métodos, método constructor, setter, getters. Abstracción. Encapsulación. Herencia. Recursividad. Resolución de ejercicios de aplicación e implementación de clases . Lenguajes python y C++. Interés y comprensión sobre el correcto uso de Clases.

**Estrategias Metodológicas:**

* Uso de Meet, el 90% de clases. Todas grabadas por los estudiantes,
* Uso de Classroom, para prácticos y material bibliográfico.
* Uso de CS50 IDE desarrollada por la universidad de Harvard, para la enseñanza de programación.

**Evaluación:**

Como parte de la forma de adquirir estos aprendizajes y demostración práctica de los resultados alcanzados, los estudiantes tienen que realizar en un mínimo del 33%, las siguientes actividades:

Resolver ejercicios de programación, tanto con lápiz y papel come, en pc. Resolver diversos tipos de problemas comenzando por especificar su propia comprensión de Ia asignación, diseñar una solución, programar o integrar partes de código utilizando el ambiente de programación indicado.

Examen Final:

Será escrito y se aprobará con un porcentaje mínimo de 40%. Se evaluarán contenidos teóricos y prácticos.

**Bibliografía:**

Dr. Charles R. Severance. (2020) Python para todos. https://es.py4e.com/book

David Malan. (2020) Harvard College. https://cs50.harvard.edu/college/2020/fall/