Sede	Sede Andina
Localidad	Bariloche
Escuela	Escuela de Producción y Tecnología
Carrera	Ingeniería en Computación

Programa analítico de	Programación I	**Código Guaraní**
	Plan de estudio: ICOMP 2021-V9	B6003
	Ubicación en el plan de estudios	1 año - 2º CUATRIMESTRE

Correlativas vigentes	**Para cursar**	**Cursada aprobada**	Introducción a Ingeniería en Computación (B6001)
		Materia aprobada	Sin correlativas vigentes
	Para rendir	**Materia aprobada**	Introducción a Ingeniería en Computación (B6001)
Ciclo lectivo: 2024	**Régimen de cursada**		CUATRIMESTRAL
	Período de dictado		2º CUATRIMESTRE
Carga horaria semanal: 6			Carga horaria total: 96
Horas teóricas totales: 48			Horas prácticas totales: 48
Carga horaria virtual: 0			Horas de estudio extra clase recomendadas: 20

Profesor/a	MARTIN RENE VILUGRON
Equipo de docencia	Miguel MARIGUIN-MAURO ALEJANDRO FERMIN-DANIEL EDUARDO TEIRA

Fundamentación

La asignatura constituye un pilar fundamental en nuestra carrera universitaria. Nuestro objetivo principal es capacitar al estudiante en la adquisición de habilidades sólidas en programación, con un enfoque en la correctitud y eficiencia en el desarrollo de software de alta calidad desde los primeros pasos en la programación.

Esta materia se ha diseñado de manera incremental para garantizar una comprensión efectiva de la programación estructurada. Reconocemos que la programación es esencial tanto para el desarrollo de software como para la implementación de sistemas de hardware. Un diseño apropiado, una implementación precisa y la posterior optimización de sistemas son los cimientos de un desarrollo exitoso.

A través de esta asignatura, preparamos al estudiante para el diseño e implementación de soluciones a diversos problemas mediante la definición de algoritmos. Además, se abordan los conceptos fundamentales de programación, así como las estructuras de datos esenciales. Se introducen tipos simples, tipos definidos por el usuario y tipos de datos lineales y no lineales, que permiten abordar una amplia variedad de problemas a través de su combinación y variación.

Nuestro enfoque principal en esta asignatura es el lenguaje de programación C, que brinda una base sólida para el desarrollo de software eficiente y de alta calidad. Con Programación 1, sentamos las bases para el éxito de nuestros estudiantes en su futura trayectoria en el mundo de la programación y la informática.

Propósitos de la asignatura

Se ha optado por enmarcar la cátedra en un enfoque actual y moderno que aborda tanto técnicas teóricas como prácticas para el desarrollo de habilidades sólidas en programación. Nuestro objetivo es que los estudiantes adquieran competencias en programación estructurada y sean capaces de aplicar métodos de vanguardia en el campo de la informática.

Durante el curso, los estudiantes explorarán y dominarán el manejo de estructuras de datos en memoria estática y dinámica, lo que incluye el uso de arreglos de una o varias dimensiones y punteros. Estos conceptos son esenciales en la programación moderna y se promoverá su aplicación en situaciones reales.

Además, se fomentará el desarrollo de capacidades analíticas, técnicas e intuitivas en los estudiantes para que puedan diseñar y analizar la complejidad de algoritmos de manera efectiva. Se promoverá la generación de algoritmos recursivos como una herramienta fundamental para abordar una variedad de problemas. Los estudiantes también aprenderán a utilizar herramientas de depuración de algoritmos, lo que es esencial para el desarrollo de software robusto.

Contenidos mínimos según plan de estudio

Revisión de estructuras de control. Secuencia, condiciones, iteradores, funciones y parámetros. Prototipos y argumentos. Entrada y salida estándar. Datos definidos por el usuario: registros, vectores, matrices, arreglos n-dimensionales. Punteros y referencias. Memoria estática y dinámica. Recursión. Algoritmos de búsqueda. Algoritmos de orden iterativo y recursivo. Análisis y diseño de algoritmos: complejidad, notación 00, depuración.

Propuesta metodológica

La asignatura consistirá de clases teóricas y clases prácticas.

En las clases teóricas se desarrollarán los temas del programa de la asignatura, incluyendo múltiples ejemplos que faciliten la asimilación de los contenidos conceptuales. Mediante ejemplos prácticos se relacionarán cada uno de los temas vistos. Se fomentará la interacción del alumno con el objetivo de que logre una actitud activa para, a través de su propio razonamiento, logre crear soluciones creativas a problemas planteados durante las clases.

Mientras que en las clases prácticas, se plantearán distintas prácticas para favorecer la asimilación de los conceptos vistos en la materia. Al inicio de cada práctica el profesor hará una breve explicación de práctica donde se propondrán ejemplos similares a los ejercicios propuestos en la práctica.

La distribución de los ejercicios, así como su revisión, está pensado para utilizar la herramienta git junto al servicio GitHub. Así como la utilización de un espacio de discusión sobre la plataforma para facilitar las comunicaciones de manera asíncrona.

Para facilitar el cursado de los alumnos, se grabarán todas las clases para que estas puedan ser seguidas en cualquier momento.

Ajustes para estudiantes con discapacidad

La UNRN desarrolla políticas en torno a accesibilidad académica para estudiantes con discapacidad y acompañamiento de equipos docentes, en consonancia con las normativas nacionales y orientaciones del CIN. Al inicio de la cursada se espera contar con el relevamiento del espacio de APD (Asistencia Pedagógica en discapacidad) sobre los/las estudiantes que cursen la materia con el fin de identificar barreras y definir las configuraciones de apoyo necesarias.

Unidades

Nombre de la unidad: Lenguaje C base

Semanas: 1,2

Contenidos: Introducción a Lenguaje C.
Preprocesado y compilación.
Tipos de datos escalares y sus literales.
Instrucción de asignación
Control de pantalla y teclado para el ingreso y egreso de datos.
Sentencias condicionales: if, switch.
Estructuras de control de flujo: for, while, do/while.
Expresiones y operadores lógicos.
Importancia de las funciones en la descomposición de problemas.
Pasaje de parámetros y tipos de parámetros.
Ámbito o alcance de variables.
Cómo comentar y documentar funciones.

Actividades practicas: Repositorio Practica #1
Repositorio Practica #2

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.

Bibliografía complementaria: Sin bibliogafía complementaria

Nombre de la unidad: Arreglos y cadenas

Semanas: 3

Contenidos: Arreglos unidimensionales.
Noción de tamaño y capacidad en cadenas.
Introducción al modelo de memoria en C
Uso en funciones, efectos de su uso como argumento en funciones.
Como documentar su uso.

Actividades practicas: Repositorio Practico #3

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.
C Manual de referencia. 4ta Edición. Herbert Schildt. McGraw-Hill. 2004.

Bibliografía complementaria: Sin bibliogafía complementaria

Nombre de la unidad: Punteros

Semanas: 4

Contenidos: Concepto de puntero.
Usos de punteros.
El modelo de memoria en C.
Aritmética de punteros
Los operadores "contenido de" y "dirección de"
Similitudes y diferencias con arreglos.
Efectos de los punteros en el pasaje de argumentos.
Documentación del uso de punteros.
Punteros a función

Actividades practicas: Repositorio Practico #4

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.
C Manual de referencia. 4ta Edición. Herbert Schildt. McGraw-Hill. 2004.

Bibliografía complementaria: Sin bibliogafía complementaria

Nombre de la unidad: Repaso general #1

Semanas: 5

Contenidos: Se repasaran todos los temas anteriores, haciendo hincapié en las dudas y correcciones frecuentes de las practicas.
(esta previsto que el primer parcial sea la semana siguiente)

Actividades practicas: Revisión de las practicas 1 al 4 resueltas.

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.
C Manual de referencia. 4ta Edición. Herbert Schildt. McGraw-Hill. 2004.

Bibliografía complementaria: Sin bibliogafía complementaria

Nombre de la unidad: Matrices y estructuras

Semanas: 6

Contenidos: Matrices: declaración, inicialización y operaciones.
Uso en funciones
Estructuras definidas por el usuario (struct).
Tipos de datos definidos por el usuario (typedef).

Actividades practicas: Repositorio practico #5

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.
C Manual de referencia. 4ta Edición. Herbert Schildt. McGraw-Hill. 2004.

Bibliografía complementaria: Sin bibliogafía complementaria

Nombre de la unidad: Archivos

Semanas: 7

Contenidos: Archivos de texto. Definición y uso.
Operaciones básicas, lectura y manipulación.
Gestión de errores.
Utilización de los argumentos de programa.

Actividades practicas: Repositorio practico #6

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.
C Manual de referencia. 4ta Edición. Herbert Schildt. McGraw-Hill. 2004.

Bibliografía complementaria: Sin bibliogafía complementaria

Nombre de la unidad: Memoria Dinámica

Semanas: 8

Contenidos: Las regiones de la memoria de un programa (automática/estática/dinámica), desbordamiento de segmentos.
Usos, ventajas y desventajas.
Punteros genéricos (void*), noción y usos.
Gestión de la memoria dinámica. malloc/realloc/free.
Conversiones (casts)
Arreglos de largo dinámico (ALV), usos y problemas.
Estrategias de uso.

Actividades practicas: Repositorio practico #7

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.
C Manual de referencia. 4ta Edición. Herbert Schildt. McGraw-Hill. 2004.

Bibliografía complementaria: Sin bibliogafía complementaria

Nombre de la unidad: Repaso General #2

Semanas: 9

Contenidos: Repaso de matrices, estructuras, archivos y memoria dinámica.
Previo al segundo parcial.

Actividades practicas: Revisión de las practicas 7 al 7 resueltas.

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.
C Manual de referencia. 4ta Edición. Herbert Schildt. McGraw-Hill. 2004.

Bibliografía complementaria: Sin bibliogafía complementaria

Nombre de la unidad: Estructuras basadas en nodos

Semanas: 10

Contenidos: Introducción a las estructuras de datos.
Estructuras para la gestión de los datos
Listas enlazadas, similitudes y diferencias con arreglos.
Pilas y colas
Utilización de memoria dinámica con estructuras.
Creación dinámica de estructuras

Actividades practicas: Repositorio practico #8

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.
C Manual de referencia. 4ta Edición. Herbert Schildt. McGraw-Hill. 2004.

Bibliografía complementaria: Sin bibliogafía complementaria

Nombre de la unidad: Algoritmos recursivos

Semanas: 11

Contenidos: Recursión
Condición base y condición de finalización. Sumatoria, Factorial, Fibonacci.
Efectos de su uso en un programa.
Divide y conquista

Actividades practicas: Repositorio practico #9

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.
C Manual de referencia. 4ta Edición. Herbert Schildt. McGraw-Hill. 2004.

Bibliografía complementaria: Sin bibliogafía complementaria

Nombre de la unidad: Análisis y diseño de algoritmos: complejidad, notación O().

Semanas: 12,13

Contenidos: Complejidad de algoritmos
Eficiencia de algoritmos.
Análisis, tiempo de ejecución y utilización de memoria.
Comportamiento en el mejor caso, caso promedio y peor caso.
Ejemplificación y análisis de algoritmos (búsqueda, ordenación, recursivos).
Notación O()
Análisis comparativo entre arreglos y listas.

Actividades practicas: Repositorio practico #10

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.
C Manual de referencia. 4ta Edición. Herbert Schildt. McGraw-Hill. 2004.

Bibliografía complementaria: Sin bibliogafía complementaria

Nombre de la unidad: Algoritmos de búsqueda y ordenamiento

Semanas: 14,15

Contenidos: Algoritmos de Búsqueda:
Búsqueda secuencial.
Búsqueda binaria.

Algoritmos de Ordenamiento
Ordenamiento por burbuja.
Ordenamiento por inserción.
Ordenamiento por selección.
Merge Sort.
Quick Sort.
Bogosort

Análisis de los diferentes casos.
Estrategias de ordenamiento (particionado, inserción, intercambio, selección)
Noción de estabilidad del algoritmo.
Comparación de la eficiencia de los algoritmos de búsqueda y ordenamiento

Actividades practicas: Trabajo Integrador Final.

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.
C Manual de referencia. 4ta Edición. Herbert Schildt. McGraw-Hill. 2004.

Bibliografía complementaria: Sin bibliogafía complementaria

Nombre de la unidad: Recuperatorios y cierre de la materia

Semanas: 16

Contenidos: Se expondrán los trabajos integradores y tomará el examen recuperatorio a quienes lo necesiten.

Actividades practicas: Trabajo Integrador Final

Bibliografía obligatoria: El lenguaje de programación C. 2da Edición. Brian W. Kernighan, Dennis M. Ritchie. Prentice-Hall Hispanoamericana. 1991.
C Manual de referencia. 4ta Edición. Herbert Schildt. McGraw-Hill. 2004.

Bibliografía complementaria: Sin bibliogafía complementaria

Propuesta de evaluación

La evaluación de la cátedra se compone de dos ejes principales:

- Evaluación formativa: Se llevará a cabo a través de Trabajos Prácticos (TP) semanales. Los TP serán evaluados de manera cualitativa, recibiendo una calificación de 'OK', 'A corregir' o 'Rechazado'. Esta evaluación se verá respaldada por el uso de herramientas de análisis estático y de calidad de código automatizadas. Estas herramientas permitirán identificar de manera temprana posibles errores, malas prácticas de programación y áreas de mejora en los códigos entregados. La retroalimentación proporcionada por estas herramientas, junto con la evaluación cualitativa del docente, permitirá a los estudiantes obtener una visión más precisa y detallada de su progreso.

- Evaluación sumativa: Se realizarán dos exámenes escritos en papel, diseñados para evaluar los conocimientos teóricos adquiridos. Estos exámenes tendrán un carácter más formal y contribuirán a la calificación final de la materia.

Asignatura posible de ser promocionada sin examen final

Condiciones de regularidad y régimen de promoción

Se establece que la materia es promocionable con la escala Promoción 7 - Promoción.
El régimen resuelve las siguientes condiciones: Todos los trabajos prácticos entregados, con un máximo de uno (1) con correcciones pendientes.

Asignatura posible de ser rendida libre

Se establece el régimen para las/los estudiantes que deseen rendir libre de las siguiente manera: Los estudiantes que deseen rendir la materia en esta modalidad, deberán demostrar un alto grado de autonomía y capacidad para resolver problemas de programación.

Para ello, deberán:

- Resolver un ejercicio base antes de rendir: La cátedra se contactará para indicarles el ejercicio a resolver en C. El mismo está diseñado para evaluar los conocimientos y habilidades requeridos por la cátedra.
- Desarrollar el ejercicio de manera independiente: Implementar una solución al ejercicio propuesto, utilizando los recursos disponibles y demostrando un dominio sólido de los conceptos de programación en C, junto a las técnicas y herramientas dadas por la cátedra.
- Presentar el ejercicio y sustentar las decisiones tomadas: Durante el examen oral, el estudiante deberá explicar el código desarrollado, justificar las decisiones de diseño y responder a preguntas sobre los conceptos teóricos involucrados.

Requisitos de acreditación

Para aprobar la materia es necesario entregar todos los trabajos prácticos dentro de las fechas establecidas (hasta 2 semanas luego de la 'emisión' de la práctica). Se prevé algún grado de flexibilidad en la fecha de entrega a medida de la cursada.

Las notas de los parciales no se promedian. Para regularizar y promocionar es necesario que ambos exámenes estén en la misma condición.

Fechas tentativas de evaluación

Se prevé tomar parcial en la semana 5, luego una segunda evaluación en la semana 10 más una evaluación a definir en la semana 15 que puede ser un examen escrito o la elaboración de un proyecto de mayor complejidad.
Con un recuperatorio para estos exámenes en la última semana de clases.