Resolución Consejo Directivo FCEyN N° 456 / 2025

Santa Rosa, 24 de octubre de 2025

VISTO:

El Expediente. Nº 633/2025, iniciado por Secretaría Académica, Programas actualizados Dpto. de Matemática - año 2025, y

CONSIDERANDO:

Que el docente Prog. Sup. Claudio Luis PONZIO, a cargo de la asignatura "Introducción a la Computación" que se dicta para la carrera Profesorado en Matemática (Plan 2015), eleva el programa de la citada asignatura para su aprobación a partir del ciclo lectivo 2025 en adelante.

Que el mismo cuenta con el aval del Mg. Rubén PIZARRO y de la Mesa de Carrera del Profesorado en Matemática.

Que en la sesión ordinaria del 23 de octubre de 2025 el Consejo Directivo aprobó, por unanimidad, el despacho presentado por la Comisión de Enseñanza.

POR ELLO:

EL CONSEJO DIRECTIVO DE LA
FACULTAD DE CIENCIAS EXACTAS Y NATURALES
R E S U E L V E:

ARTÍCULO 1º: Aprobar el Programa de la asignatura "Introducción a la Computación" correspondiente a la carrera Profesorado en Matemática (Plan 2015), a partir del ciclo lectivo 2025 en adelante, que como Anexos I, II, III, IV, V, VI y VII forma parte de la presente Resolución.

ARTÍCULO 2º: Regístrese, comuníquese. Pase a conocimiento de Secretaría Académica, Departamento de Asuntos Estudiantiles, Departamento de Matemática y Computación, del docente Prog. Sup. Claudio Luis PONZIO, y del CENUP. Cumplido, archívese.

Maite BETELU – Secretaria Académica – FCEyN - UNLPam

Nora Claudia FERREYRA – Decana – FCEyN - UNLPam

ANEXO I

DEPARTAMENTO: Matemática y Computación

ACTIVIDAD CURRICULAR: Introducción a la Computación

CARRERA-PLAN/ES:
Profesorado en Matemática (Plan 2015)

CURSO: Segundo año

RÉGIMEN: Cuatrimestral (Primer Cuatrimestre)

CARGA HORARIA SEMANAL:

Teóricos: 4 horas semanales
Prácticos: 4 horas semanales

CARGA HORARIA TOTAL:
120 horas

CICLO LECTIVO: 2025 en adelante

EQUIPO DOCENTE:

Prog. Sup. Claudio Luis PONZIO: Profesor Adjunto - Dedicación Simple - Regular
Mg. Silvia BAST: Jefa de Trabajos Prácticos - Dedicación Simple - Interino
Prof. Lorena CAVERO: Ayudante de Primera - Dedicación Simple - Regular
Prof. Alejandro Rojas: Ayudante de Primera - Dedicación Simple - Interino
Lic. Sofia FUNKNER: Ayudante de Primera - Dedicación Simple – Interino

FUNDAMENTACIÓN:
La tecnología ocupa un lugar central en la vida contemporánea. La mayoría de las actividades cotidianas se encuentran mediadas por dispositivos y sistemas tecnológicos: búsqueda de información, formación virtual, gestión de trámites en línea, transacciones comerciales, trabajo colaborativo en entornos digitales o interacción en redes sociales. A esto se suman los entornos domésticos y urbanos, en los que la domótica, los electrodomésticos conectados, los sistemas de climatización inteligentes, los vehículos con asistencia computarizada, los GPS o los teléfonos inteligentes forman parte del escenario habitual.

En este contexto, resulta imprescindible comprender el funcionamiento del medio tecnológico en el que se vive. Las Ciencias de la Computación constituyen un campo fundamental, ya que aportan las bases conceptuales de las Tecnologías de la Información y la Comunicación (TIC). Entre sus aportes más significativos se encuentran los métodos para describir y resolver problemas mediante algoritmos y programación.

El aprendizaje de la programación contribuye al desarrollo del pensamiento lógico, la capacidad de abstracción, la previsión y la creatividad en la resolución de problemas. También impulsa el ejercicio de la verificación, la revisión crítica y la optimización de soluciones a través de procesos iterativos de mejora.

La asignatura Introducción a la Computación brinda un primer acercamiento a nociones básicas de resolución de problemas y estrategias de diseño de algoritmos, incluyendo el análisis de enunciados, el uso de heurísticas, la representación de la información y el empleo de elementos esenciales de un lenguaje de alto nivel. Además, incorpora el estudio de la historia de la informática como marco contextual que permite comprender la evolución de los lenguajes y de las estructuras de control. El trabajo con tipos de datos simples y estructurados homogéneos contribuye a sentar las bases para la construcción de programas más complejos en instancias posteriores de la formación.

El propósito central es que los/las futuros/as profesores/as adquieran criterios y métodos de trabajo que les permitan analizar, desde una perspectiva cualitativa y cuantitativa, la resolución e interpretación de algoritmos computacionales, comprendiendo la lógica de funcionamiento de las tecnologías que atraviesan los distintos ámbitos de la vida social, académica y profesional.

OBJETIVOS Y/O ALCANCES DE LA ASIGNATURA:

Que el/la estudiante logre

- Comprender y aplicar una metodología de resolución de problemas en situaciones prácticas.
- Comprender y utilizar apropiadamente estructuras de control y tipos de datos en la resolución de un problema dado.
- Diseñar programas en un lenguaje de alto nivel haciendo uso de un entorno integrado de desarrollo (IDE).

Objetivos específicos

- Describir la estructura básica de una computadora.
- Reconocer la evolución de los lenguajes de computación.
- Conocer técnicas para la resolución de problemas de solución algorítmica.
- Conocer y utilizar un IDE para la implementación de los algoritmos en un lenguaje de programación de alto nivel.
- Interpretar el enunciado de un problema.
- Utilizar técnicas para la resolución de problemas de solución algorítmica.
- Trazar un plan de accio(nes) que lleve(n) a una posible solución.
- Seleccionar estructuras de control y tipos de datos, que ofrece un lenguaje de alto nivel, para la resolución de problemas.

- Diseñar un programa en un lenguaje de alto nivel para dar solución a problemas sencillos.
- Analizar la solución en busca de mejoras.

ANEXO II

ASIGNATURA: Introducción a la Computación

CICLO LECTIVO: 2025 en adelante

PROGRAMA ANALÍTICO

Unidad I: Fases de la resolución de problemas. Resolución de problemas usando un lenguaje de diseño de algoritmos. Diseño estructural y modular. Empleos de algoritmos simples como primitivas en la resolución de problemas más complejos.

Unidad II: Algoritmos. Elementos de un algoritmo. Lenguaje de diseño de algoritmos: sintaxis y semántica. Estructura general de un algoritmo. Datos de entrada/salida. Acciones. Manipulación de datos. Introducción a las estructuras de control. Primitivas para manipulación de datos. Primitivas que implementan estructuras de control. Problemas con solución algorítmica.

Unidad III: Generalidades del lenguaje de programación utilizado. Estructura general de un programa. Descripción sintáctica. Encabezamiento, bloque, Declaraciones. Sección ejecutable. Elementos de un programa: declaración de constantes y variables; concepto de asignación; identificadores; palabras reservadas; identificadores predefinidos; símbolos especiales; constantes numéricas y constantes caracter; separadores de elementos; sentencias; separadores de sentencias; documentación interna y comentarios; indentación.

Unidad IV: Tipos de datos. Clasificación. Tipos de datos simples. Tipos predefinidos: integer, real, char y boolean. Tipos de datos definidos por el usuario: enumerado y subrango. Tipos de datos estructurados: array (unidimensionales y bidimensionales). Tipos de datos String: funciones y procedimientos. Operadores. Expresiones. Precedencia de operadores. Compatibilidad de tipos. Funciones de conversiones numéricas y de cadenas. Funciones matemáticas. Funciones especiales. Sentencias para Entrada/Salida de datos.

Unidad V: Estructuras de control. Construcciones condicionales: if...then, case...of. Usos y conveniencias. Definición de condición simple y condición compuesta. Construcciones iterativas: while...do, for...do , repeat...until. Correspondencia entre las distintas estructuras. Diferencias en la ejecución. Usos y conveniencias.

Unidad VI: Introducción al uso de procedimientos y funciones. Uso e invocación de procedimientos y funciones predefinidos. Creación de procedimientos de eventos mediante herramientas de programación. Reglas básicas de ámbito. Pasaje de parámetros. Parámetros formales y actuales. Parámetros de entrada y salida. Compatibilidad de tipos de datos en el pasaje de parámetros. Ventajas del uso de procedimientos y funciones.

Unidad VII: Evolución histórica de la computación. Breve descripción de la organización física (hardware) de una computadora: partes de la CPU. Periféricos de entrada/salida.

Almacenamiento de la información. Unidades de medidas. Software, clasificación y distintas consideraciones sobre nuevas tecnologías.

ANEXO III

ASIGNATURA: Introducción a la Computación

CICLO LECTIVO: 2025 en adelante

BIBLIOGRAFÍA:

Arboledas Brihuega, D. (2019) **Aprenda a programar con Lazarus**. RA-MA S.A. Editorial y Publicaciones. ISBN: 9788499645117.

Cornell, G. y Strain, T. (1995). **Programación en Delphi.** MADRID: MCGRAW-HILL INTERAMERICANA.

Joyanes Aguilar, L. (1993). **Programación en Turbo Pascal, versión 5.5, 6.0 y 7.0**. S.A. MCGRAW-HILL / INTERAMERICANA DE ESPAÑA, segunda edición.

Joyanes Aguilar, L. (2003). **Fundamentos de programación**. MADRID: MCGRAW-HILL INTERAMERICANA, tercera edición.

Joyanes Aguilar, L. (2006). **Programación en Pascal,** MADRID: MCGRAW-HILL INTERAMERICANA, cuarta edición.

Thomson S. (1996). **How to program it.** Computing Laboratory. University of Kent, Canterbury, UK. Disponible en: https://www.cs.kent.ac.uk/people/staff/sjt/Haskell_craft/HowToProgIt.html

Polya, G. (1965). **Cómo plantear y resolver problemas**. México: Editorial Trillas.

Zapotecatl López, J. L. (2018) **Introducción al pensamiento computacional: conceptos básicos para todos.** ACADEMIA MEXICANA DE COMPUTACIÓN, A, C., Ciudad de México.

Material producido por la propia cátedra.

ANEXO IV

ASIGNATURA: Introducción a la Computación

CICLO LECTIVO: 2025 en adelante

PROGRAMA DE TRABAJOS PRÁCTICOS:

La práctica de la asignatura se organiza en torno al uso de Guías Prácticas (GP). En ellas se plantean ejercicios y problemas, junto con tips y referencias del lenguaje de programación que resultan de utilidad para que los/las estudiantes puedan avanzar con mayor autonomía.

Cada GP presenta situaciones problemáticas que apuntan a la consecución de los siguientes objetivos específicos:

- Conocer técnicas para la resolución de problemas de solución algorítmica.
- Interpretar el enunciado de un problema.
- Trazar un plan de acciones que lleve a una posible solución.
- Utilizar técnicas de resolución algorítmica en la implementación de programas.
- Comprobar y analizar los resultados obtenidos, en busca de mejoras en la solución.

Las GP se estructuran siguiendo una secuencia progresiva. En primer lugar, se presentan situaciones problemáticas iniciales que los/las estudiantes deben resolver utilizando únicamente los conocimientos con los que cuentan hasta ese momento. Posteriormente, se introduce el nuevo concepto o estructura que permite mejorar el código generado inicialmente.

A continuación, cada GP incluye ejercicios específicos destinados a ejercitar el nuevo concepto o estructura. Finalmente, se presentan **situaciones problemáticas**, en las que los/las estudiantes deben:

- Reutilizar, contextualizar y adaptar los ejercicios resueltos.
- Combinar distintos recursos trabajados.
- Elaborar nuevas propuestas de solución.

En los primeros ejercicios, las GP suelen acompañar los enunciados con imágenes y videos de posibles soluciones, favoreciendo la comprensión y la autonomía en el proceso de aprendizaje.

Guía Práctica 1: Introducción a Entornos de Desarrollo Integrado (IDE)
Para el desarrollo de la práctica de la asignatura se hará uso de un IDE para la implementación de programas en un lenguaje de alto nivel. El objetivo de esta guía es que los/las estudiantes conozcan los elementos del IDE y puedan comenzar a utilizarlo para producir programas sencillos haciendo uso de algunos de los componentes visuales que provee. Asimismo, se trabaja con la solución de problemas y ejercicios a través de la implementación de algoritmos utilizando la estructura de control secuencial.
Corresponde a contenidos incluidos en las Unidades I, II y III.

Guía Práctica 2: Entrada/Salida. Ingresar y mostrar información

Las sentencias de entrada/salida permiten la comunicación entre el usuario y el sistema, y los componentes del IDE que la facilitan forman parte de la interfaz. Esta GP propone diferentes situaciones problemáticas que requieren el uso de sentencias de entrada y salida y de los objetos o componentes del IDE que las implementan. Los/las estudiantes deben seleccionar el tipo de dato (simple) apropiado y definir las variables, operaciones, procedimientos y funciones sencillas (predefinidas) para resolver dichos problemas.
Corresponde a contenidos incluidos en las Unidades IV y VI del programa analítico y retoma los tratados en unidades anteriores.

Guía Práctica 3: Sentencias selectivas

Las sentencias selectivas permiten evaluar una condición y, en función de su resultado, optar por la ejecución de uno u otro grupo de instrucciones. Este tipo de estructura resulta fundamental para dotar a los programas de flexibilidad y adaptabilidad frente a distintos escenarios posibles. En esta GP se presentan situaciones problemáticas que requieren:

- Integrar los conceptos de las guías anteriores.
- El uso de estructuras selectivas (if y case) para decidir el camino de ejecución más adecuado.
- La selección de tipos de datos simples apropiados, así como la definición de variables, operaciones y funciones necesarias para cada solución.

Los/las estudiantes deben diseñar programas que permitan elegir, durante su ejecución, el conjunto de instrucciones que resuelva cada problema, transfiriendo y ampliando los saberes adquiridos previamente.
De esta manera, la guía introduce las estructuras de control selectivas y promueve el desarrollo de soluciones más complejas, robustas y acordes a los requerimientos de cada situación.
Corresponde a contenidos incluidos en las Unidades V y VI del programa analítico y retoma y profundiza los tratados en unidades anteriores.

Guía Práctica 4: Sentencias repetitivas simples + string

Las sentencias repetitivas (o estructuras de control repetitivas) permiten ejecutar un conjunto de instrucciones varias veces de manera más clara y eficiente. Esta GP propone a los/las estudiantes situaciones problemáticas que integran los aprendizajes de las guías anteriores, incorporando ahora el uso apropiado de la estructura de control repetitiva simple for.

La propuesta se organiza en tres etapas:

- Optimización de proyectos previos: se revisan programas desarrollados antes de conocer las sentencias repetitivas y se los reescribe para hacerlos más claros, legibles y eficientes.
- Ejercicios específicos: permiten ejercitar el uso de la repetitiva for aplicada a strings y otros contextos básicos.
- Problemas integradores: requieren aplicar los ejercicios resueltos, contextualizándolos y adaptándolos a nuevas situaciones, lo que demanda integrar conceptos de GP anteriores y el nuevo recurso.

De esta manera, los/las estudiantes avanzan en el reconocimiento de patrones de repetición, la identificación de grupos de sentencias que deben reiterarse y la selección de la estructura más adecuada para cada situación.

Corresponde a contenidos incluidos en las Unidades IV, V y VI del programa analítico y retoma y profundiza los tratados en unidades anteriores.

Guía Práctica 5: Arreglos unidimensionales + sentencias repetitivas simples y condicionales

Un arreglo es un tipo de dato estructurado que permite almacenar múltiples elementos del mismo tipo, por lo que también se lo denomina tipo estructurado homogéneo. Su uso facilita la representación y manipulación de grandes volúmenes de datos de manera ordenada y eficiente.

En esta GP se introducen las sentencias repetitivas condicionales (while y repeat…until), que permiten ejecutar un conjunto de instrucciones mientras se cumpla una condición. Este recurso resulta especialmente útil en situaciones donde no se conoce de antemano la cantidad de repeticiones necesarias.

Las actividades propuestas requieren:

- Integrar los contenidos de las guías anteriores.
- Seleccionar el tipo de dato estructurado más apropiado para representar la información (arreglos unidimensionales).
- Elegir la estructura repetitiva que mejor se adapte a cada problema.
- Combinar estructuras de control (secuenciales, condicionales y repetitivas) con el uso de arreglos.
- Aplicar estrategias de resolución de problemas para diseñar programas claros, legibles y eficientes.

De este modo, los/las estudiantes comienzan a trabajar con estructuras de datos que permiten organizar la información y a utilizarlas en conjunto con las estructuras de control, avanzando hacia soluciones cada vez más cercanas a situaciones reales de la programación.

Corresponde a contenidos incluidos en las Unidades IV, V y VI del programa analítico y retoma y profundiza los tratados en unidades anteriores.

Guía Práctica 6: Arreglos bidimensionales + sentencias repetitivas simples y condicionales

Un arreglo bidimensional es un tipo de dato estructurado que permite almacenar datos homogéneos en forma de tabla, organizados en filas y columnas. Su uso resulta fundamental cuando la información no puede representarse de manera lineal, como ocurre con horarios, planillas de calificaciones, mapas o matrices matemáticas.

En esta GP se propone trabajar con arreglos bidimensionales en combinación con estructuras de control secuenciales, condicionales y repetitivas. Los/las estudiantes deberán:

- Integrar los conceptos de las guías anteriores.
- Seleccionar el tipo de arreglo (unidimensional o bidimensional) que mejor represente la información planteada en cada problema.
- Elegir la estructura repetitiva más adecuada según la situación.
- Aplicar técnicas de resolución de problemas para diseñar programas que hagan un uso eficiente de arreglos y estructuras de control.
- Contextualizar las soluciones en situaciones reales (por ejemplo, procesamiento de datos en tablas o registros múltiples).

De esta manera, los/las estudiantes avanzan en la construcción de programas que manipulan volúmenes mayores y más estructurados de información, fortaleciendo las bases para comprender y trabajar con estructuras de datos más complejas en etapas posteriores de la carrera.

Corresponde a contenidos incluidos en las Unidades IV, V y VI del programa analítico y retoma y profundiza los tratados en unidades anteriores.

ANEXO V

ASIGNATURA: Introducción a la Computación

CICLO LECTIVO: 2025 en adelante

ACTIVIDADES ESPECIALES QUE SE PREVÉN
Desarrollo y resolución de un problema, trabajo práctico integrador, involucrando los contenidos abordados en cada una de las unidades presentadas en el Programa Analítico.

ANEXO VI

ASIGNATURA: Introducción a la Computación

CICLO LECTIVO: 2025 en adelante

PROGRAMA DE EXAMEN

El programa de examen coincide con el programa analítico.

ANEXO VII

ASIGNATURA: Introducción a la Computación

CICLO LECTIVO: 2025 en adelante

METODOLOGÍA DE EVALUACIÓN

La evaluación del estudiantado para regularizar la asignatura se llevará a cabo a través de dos instancias principales:

Evaluación de las Guías Prácticas (GP)

Al finalizar cada GP, se propone a los/las estudiantes un proyecto de programación con errores. El objetivo es que lo analicen, lo corrijan y lo hagan funcionar, como instancia de ejercitación y preparación para el cuestionario de autoevaluación que sigue a continuación.
Posteriormente, se selecciona un problema de la GP y se publican los criterios de corrección o la lista de cotejo. Los/las estudiantes deben presentar la solución en la fecha estipulada.
Estas actividades buscan marcar un ritmo sostenido de avance en la cursada, manteniendo la continuidad de trabajo y contribuyendo a la calificación de los parciales.

Evaluación mediante exámenes parciales

Cada examen parcial consiste en la resolución de un ejercicio tomado de las GPC incluidas en el programa, con la consigna de modificar y agregar funcionalidades al problema planteado.
Durante los primeros 15 minutos, los/las estudiantes participan de una instancia de evaluación colaborativa, discutiendo colectivamente el enunciado. Luego, cada estudiante continúa de manera individual con la resolución del examen.

Los/las estudiantes deben aprobar dos instancias evaluativas (o sus respectivos recuperatorios). Tendrán acceso a rendir un examen adicional en caso de haber aprobado un solo parcial o su respectivo recuperatorio, según reglamento vigente.

Hoja de firmas

Sistema: sudocu
Fecha: 28/10/2025 10:47:13
Cargado por: Nora Claudia Ferreyra

Sistema: sudocu
Fecha: 28/10/2025 11:46:39
Autorizado por: Maite Betelu

Sistema: sudocu
Fecha: 28/10/2025 12:12:49
Autorizado por: Nora Claudia Ferreyra