Campus de la UNA
SAN LORENZO-PARAGUAY

UNIVERSIDAD NACIONAL DE ASUNCIÓN
FACULTAD POLITÉCNICA
CONSEJO DIRECTIVO

RESOLUCIÓN 24/25/30-00
ACTA 1207/09/12/2024

"POR LA CUAL SE APRUEBA EL PROGRAMA DE ESTUDIO DE LA ASIGNATURA *LENGUAJES DE PROGRAMACIÓN 2*, DE LA CARRERA INGENIERÍA EN INFORMÁTICA – PLAN 2023 DE LA FP-UNA"

VISTO: El Memorando DA/2379/2024 del Director Académico de la FP-UNA, Prof. MSc. Felipe Santiago Uzabal Escurra, con el cual remite el Memorando CCPTCC/034/2024 de la Comisión Coordinadora del Proyecto de Transformación Curricular de Carreras de Grado de la FP-UNA, en el que presenta la propuesta de Programas de Estudio de las Asignaturas de la Carrera Ingeniería en Informática.

CONSIDERANDO: La Ley 4995/2013 de Educación Superior, el Estatuto de la Universidad Nacional de Asunción y las deliberaciones sobre el tema.

Que la Comisión Coordinadora del Proyecto de Transformación Curricular de Carreras de Grado, solicita la aprobación del Programa de Estudio de la asignatura **"Lenguajes de Programación 2"**, de la carrera Ingeniería en Informática – Plan 2023, cuyo plan de estudio ya fue aprobado por el Consejo Superior Universitario.

EL CONSEJO DIRECTIVO DE LA FACULTAD POLITÉCNICA
RESUELVE:

24/25/30-01 APROBAR el Programa de Estudio de la Asignatura **"Lenguajes de Programación 2"**, de la carrera Ingeniería en Informática – Plan 2023 de la FP-UNA, detallado en el ANEXO 16 de la presente Acta.

24/25/30-02 COMUNICAR, copiar y archivar.

Prof. Abg. Joel Arsenio Benítez Santacruz
Secretario

Prof. Ing. Silvia Teresa Leiva León, MSc.
Presidenta

TEL/FAX: +595 21 5887000 – C.C. 1130 (Asunción) – 2111 (San Lorenzo)
www.pol.una.py

DEPARTAMENTO DE ENSEÑANZA DE INFORMÁTICA
PROGRAMA DE ESTUDIO

I. IDENTIFICACIÓN

Asignatura	Lenguajes de Programación 2				
Carrera	**Plan**	**Sede/Filial**	**Carácter**	**Semestre**	**Prerrequisitos**
Ingeniería en Informática	2023	Sede San Lorenzo	Obligatoria	Segundo	Lenguajes de Programación 1, Algoritmos y Estructura de Datos 1
Horas semanales	4				
Total de horas teóricas semestral	36				
Total de horas prácticas semestral	36				
Total de horas semestral	72				
Valor en créditos académicos	La valoración en créditos académicos será comunicada en su oportunidad, ajustada al reglamento para la aplicación del Sistema Nacional de Créditos Académicos – Paraguay en la UNA; ajuste que se encuentra en proceso de elaboración conforme a las disposiciones de la Resolución CONES N° 221/2024, en su artículo N° 10.				
Actualización	Al egreso de la primera cohorte.				

II. FUNDAMENTACIÓN

La asignatura de Lenguajes de Programación 2 incorpora una nueva forma de pensar acerca del proceso de descomposición de problemas y de desarrollo de soluciones de programación, mediante el conocimiento tanto teórico como práctico del paradigma de la programación orientada a objetos, permitiendo al estudiante desarrollar la lógica para resolver problemas de tipo algorítmico bajo los importantes conceptos que el paradigma requiere.

Es una asignatura que involucra la integración de varios elementos específicos de la programación como el paradigma orientado a objetos, el lenguaje de programación, el entorno de desarrollo, la metodología de desarrollo y el lenguaje de modelado, así como patrones de diseño, principios de desarrollo, y la lógica de programación.

Esta asignatura consta de 4 unidades programáticas que incluyen los conceptos del paradigma de la programación orientada a objetos y contribuirá con el resultado de aprendizaje RA2 Desarrollar, mantener y evaluar sistemas y servicios basados en software cumpliendo estándares de calidad, aplicando teorías, principios, métodos y mejores prácticas de ingeniería de software para aportar a la concreción de la competencia del perfil de egreso MP10 Planificar, proyectar, diseñar y ejecutar proyectos sostenibles e integrales para la resolución de problemas, la mejora y la innovación en el ámbito de las ciencias informáticas.

III. COMPETENCIAS DEL PERFIL DE EGRESO ASOCIADAS

1. Comunicarse en las lenguas oficiales del país y en una lengua extranjera.
2. Actuar proactivamente frente a los problemas sociales y ambientales.
3. Evaluar el comportamiento de diversos fenómenos disciplinares e interdisciplinares relacionados con la ingeniería en informática con una visión de sistema, mediante modelos teóricos validados y actualizados, capaces de abarcarlos integralmente en un contexto de incertidumbre.
4. Planificar, proyectar, diseñar y ejecutar proyectos sostenibles e integrales para la resolución de problemas, la mejora y la innovación en el ámbito de la ingeniería informática.

IV. ORGANIZACIÓN DE LA ASIGNATURA

Unidades	Contenidos	Resultados de Aprendizaje
1. Introducción.	1.1. Terminología. 1.1.1. Algoritmo. 1.1.2. Objetos y Clases. 1.1.3. Métodos. 1.2. Resolución de problemas con software. 1.2.1. Proceso de programación. 1.2.2. Metodología de programación. 1.2.3. Herramientas de programación. 1.2.4. Lenguajes de programación orientados a objetos. 1.2.5. Librerías y documentación del lenguaje. 1.3. Sistemas Operativos. 1.3.1. Windows. 1.3.2. Linux. 1.3.3. Android.	1. Comprende la terminología relacionada con la programación orientada a objetos. 2. Entiende la complejidad del software mediante descomposición, abstracción y jerarquías para organizar y estructurar el código de manera efectiva.
2. Conceptos básicos.	2.1. Propiedades fundamentales. 2.1.1. Abstracción 2.1.2. Encapsulamiento y ocultación de datos. 2.1.3. Herencia. 2.1.4. Reutilización o reusabilidad. 2.1.5. Polimorfismo. 2.2. Clases y objetos. 2.1.1. Tipo abstracto de datos. 2.1.2. Identificación de objetos. 2.1.3. Estado de un objeto. 2.1.4. Comportamiento. 2.1.5. Identidad. 2.1.6. Declaración e instancia de un objeto. 2.1.7. Mensajes. 2.1.8. Reglas de visibilidad.	1. Explica conceptos clave relacionados con los sistemas operativos, incluyendo su función, gestión de recursos, multitarea, y cómo los sistemas operativos interactúan con el hardware. 2. Aplica conceptos avanzados de programación, como herencia múltiple, interfaces, patrones de diseño y otros conceptos relacionados con la programación orientada a objetos.
3. Programación en Lenguaje Orientado a	3.1. Programación orientada a objetos.	1. Implementa programas aplicando conceptos de tipos y operadores,

Unidades	Contenidos	Resultados de Aprendizaje
Objetos.	3.1.1. Declaración de una clase. 3.1.2. Implementación de las clases. 3.1.3. Palabras reservadas. 3.1.4. Tipos de datos. Miembros de una clase. 3.1.5. Métodos. 3.1.6. Constructores. 3.1.7. Tipos de datos. 3.1.8. Utilización de métodos de librerías. 3.1.9. Comentarios. 3.2. Instrucciones de control 3.2.1. Estructuras de control. 3.2.2. Estructuras de repetición. 3.2.3. Instrucción de selección múltiple. 3.2.4. Operadores lógicos. 3.3. Herencia y Polimorfismo. 3.3.1. Tipos de herencia. 3.3.2. Clases abstractas. 3.3.3. Métodos abstractos. 3.3.4. Sobrecarga de métodos. 3.3.5. Uso aplicaciones del polimorfismo. 3.3.6. Sobreescritura de métodos 3.4. Arreglos. 3.4.1. Declaración y creación de arreglos. 3.4.2. Paso de arreglos a los métodos. 3.4.3. Arreglos multidimensionales. 3.4.4. Colecciones. 3.4.5. Genericidad. Código genérico. 3.5. Manejo de excepciones. 3.5.1. Manejo de Excepciones y errores. 3.5.2. Bloque try - catch. 3.5.3. Excepciones definidas en el lenguaje. 3.5.4. Lanzamiento de Lanzar y Atrapar excepciones. 3.5.5. Declaración de nuevos tipos de excepciones. 3.6. Multitarea. 3.6.1. Utilización de la multitarea. 3.6.2. Creación de hilos. 3.6.3. Estados de un hilo.	instrucciones, manipulación de strings, trabajo con arreglos, objetos y clases, así como el manejo de excepciones. 2. Diseña programas utilizando los conceptos de herencia, interfaces y encapsulamiento. 3. Diseña aplicaciones que ejecuten múltiples hilos de manera eficiente para comprender los conceptos relacionados con la concurrencia.

Unidades	Contenidos	Resultados de Aprendizaje
	3.6.4. Prioridad entre hilos. 3.6.5. Sincronización y Concurrencia. 3.6.6. Implementación de hilos. 3.7. Clases parametrizadas. 3.7.1. Genéricos. 3.7.2. Colecciones. 3.8. Modularización de soluciones. 3.9. Implementación de tipo abstracto de datos con estructuras de datos lineales o no lineales.	
4. Análisis y diseño orientado a objetos.	4.1. Especificaciones de UML 4.1.1. Diseño y representación gráfica de objetos en UML. 4.1.1.1. Características de los objetos, comportamiento, mensajes, representación gráfica. 4.1.2. Diseño y representación gráfica de clases en UML. 4.1.2.1. Representación gráfica - diagrama, declaración de una clase, reglas de visibilidad. 4.2. Proceso de análisis orientado a objetos. 4.2.1. Indagación de los requerimientos. 4.2.2. Desarrollo de casos de uso. 4.2.3. Elaboración del modelo de requerimientos. 4.2.4. Modelado de los requerimientos con UML. 4.3. Proceso de diseño orientado a objetos. 4.3.1. Conceptos de diseño orientado a objetos. 4.3.2. Modelado del diseño con UML. 4.3.3. Conceptos de mapeo relacional.	1. Aplica el proceso de análisis en el contexto de la programación orientada a objetos. 2. Aplica el proceso de diseño en programación orientada a objetos, coherentes a los problemas planteados en la fase de análisis. 3. Relaciona de manera efectiva los diferentes aspectos del desarrollo de software en un enfoque de programación orientada a objetos.

V. ESTRATEGIAS DIDÁCTICAS

En el desarrollo del programa se aplicarán estrategias didácticas conducentes a la apropiación teórica y la ejecución práctica de procesos y procedimientos, a saber:

- **Aprendizaje basado en problemas:** exposición por parte del docente de los conceptos básicos por unidad, con materiales de lectura y ejemplos orientados a la enseñanza de las competencias específicas de la asignatura. El estudiante buscará resolver un problema a través del conocimiento que adquirió en el aula.

- **Aprendizaje basado en proyectos**: el docente propondrá la realización de un proyecto que involucre todos los resultados de aprendizaje de la materia. De esta forma el estudiante participa activamente en su aprendizaje, desarrollando diferentes habilidades para solucionar un problema a través de este proyecto.

La elección particular de la estrategia didáctica aplicada será explícita en el plan de clases, de acuerdo con el perfil de los estudiantes, los recursos disponibles y el contexto educativo.

VI. ESTRATEGIAS EVALUATIVAS

Procesos de producción grupales e individuales, pruebas individuales orales y/o escritas durante el desarrollo de las unidades, implementación de algoritmos utilizando sistemas de control de versiones, redacción de informes de resultados experimentales de algoritmos. Todos estos serán valorados y que en su conjunto aportarán para la calificación y promoción, las que serán aplicadas según normativas institucionales.

Con fines de calificación y promoción se aplicará la normativa sobre evaluación vigente en la institución que prevé valoraciones de proceso y final.

VII. MEDIOS AUXILIARES

Pizarras, marcadores, computadoras, proyector, parlantes para multimedia, plataforma virtual. Sala de laboratorio equipada para prácticas con sistemas operativos Linux o Windows y acceso a Internet.

VIII. BIBLIOGRAFÍA

- Deitel, P. y Deitel, H. Cómo programar en Java. Décima edición. Pearson Educación.2016.
- Joyanes Aguilar, Luis., Zahonero Martínez, Ignacio. Programación en C, C++, Java y UML. McGraw-Hill, 2014.
- Y. Daniel Liang. Introduction to Java Programming and Data Structures, Comprehensive Version. 12th Ed. Pearson. 2024.
- E. Balagurusamy. Programming with Java. 7th Ed. McGraw-Hill. 2023.
- David J. Eck. Introduction to Programming Using Java. Version 9, JavaFX Edition. 2022. Online: https://math.hws.edu/javanotes/
- SZNAJDLEDER, Pablo. El gran libro de Java a fondo: curso de programación. Alpha Editorial, 2020.
- Deitel, P. y H. Deitel. Java How to Program. Late Objects. 11th Ed. Pearson. 2020.
- Raoul-Gabriel Urma, Mario Fusco y Alan Mycroft. Modern Java in Action. Lambdas, streams, functional and reactive programming. Manning Publications. 2019.
- F. Javier, Moldes T. Manual imprescindible Java 9. Anaya. 2017.
- Jimenez Marin, Alfonso; Pérez Montes, Francisco. Aprende a programar con Java (2. Ediciones Paraninfo, SA, 2016.
- Winblad Ann, Samuel Edwards and David R. King. Software Orientado a objetos. Addison-Wesley/Diaz de Santos. 2004.
- Meyer, Bertrand. Construcción de software Orientado a Objetos. 2da Ed/ Bertrand Meyer: Prentice, 1998.