Campus de la UNA
SAN LORENZO-PARAGUAY

UNIVERSIDAD NACIONAL DE ASUNCIÓN
FACULTAD POLITÉCNICA
CONSEJO DIRECTIVO

RESOLUCIÓN 24/26/62-00
ACTA 1208/16/12/2024

"POR LA CUAL SE APRUEBA EL PROGRAMA DE ESTUDIO DE LA ASIGNATURA *TALLER DE LENGUAJES DE PROGRAMACIÓN*, DE LA CARRERA LICENCIATURA EN CIENCIAS INFORMÁTICAS – PLAN 2023 DE LA FP-UNA"

VISTO: El Memorando DA/2437/2024 del Director Académico de la FP-UNA, Prof. MSc. Felipe Santiago Uzabal Escurra, con el cual remite el Memorando CCPTCC/036/2024 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 Licenciatura en Ciencias Informáticas.

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 **"Taller de Lenguajes de Programación"**, de la carrera Licenciatura en Ciencias Informáticas – Plan 2023, cuyo plan de estudio ya fue aprobado por el Consejo Superior Universitario.

EL CONSEJO DIRECTIVO DE LA FACULTAD POLITÉCNICA
RESUELVE:

24/26/62-01 APROBAR el Programa de Estudio de la Asignatura **"Taller de Lenguajes de Programación"**, de la carrera Licenciatura en Ciencias Informáticas – Plan 2023 de la FP-UNA, detallado en el ANEXO 54 de la presente Acta.

24/26/62-02 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	Taller de Lenguajes de Programación				
Carrera	**Plan**	**Sede/Filial**	**Carácter**	**Semestre**	**Prerrequisitos**
Licenciatura en Ciencias Informáticas	2023	Sede San Lorenzo / Filial Villarrica / Filial Coronel Oviedo	Obligatoria	Segundo	Introducción a la Programación.
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 Taller de Lenguajes de Programación se incluye como obligatoria en la malla curricular debido a su relevancia en la formación de profesionales capaces de comprender y aplicar el paradigma de la programación orientada a objetos. Este paradigma es de suma importancia en el desarrollo de software moderno, permitiendo la resolución estructurada y eficiente de problemas algorítmicos a través de conceptos clave como clases, objetos, herencia y polimorfismo.

Con un enfoque teórico-práctico, esta asignatura proporciona al estudiante las habilidades necesarias para integrar múltiples elementos del desarrollo de software: el lenguaje de programación, el entorno de desarrollo, los principios de diseño, patrones de diseño, y metodologías de desarrollo orientado a objetos.

La asignatura está estructurada en cuatro unidades, que abordan progresivamente los conceptos y habilidades requeridas:

Unidad 1: Introducción al paradigma orientado a objetos.
Unidad 2: Conceptos fundamentales de la programación orientada a objetos.
Unidad 3: Análisis y diseño orientado a objetos.
Unidad 4: Programación orientada a objetos.

III. COMPETENCIAS DEL PERFIL DE EGRESO ASOCIADAS

1. 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.

2. Seleccionar, utilizar y construir instrumentos innovadores asociados al ejercicio de las ciencias informáticas.

3. Aplicar en la práctica profesional los valores humanos, la ética y los mecanismos de seguridad laboral.

IV. ORGANIZACIÓN DE LA ASIGNATURA

Unidades	Contenidos	Resultados de aprendizaje
1. Introducción al paradigma orientado a objetos.	1.1. Terminología. 1.1.1. Objetos. 1.1.2. Clases. 1.1.3. Métodos. 1.2. Resolución de problemas con software. 1.3. Proceso de programación. 1.4. Metodología de programación. 1.5. Herramientas de programación. 1.6. Lenguajes de programación orientados a objetos. 1.7. Librerías y documentación del lenguaje. 1.8. Sistemas Operativos.	1. Explica el paradigma orientado a objetos considerando sus características principales. 2. Reconoce la relevancia del paradigma orientado a objetos en la resolución de problemas con software. 3. Identifica herramientas de programación adecuadas para escribir código en un lenguaje orientado a objetos.
2. Conceptos fundamentales de la programación orientada a objetos.	2.1. Propiedades fundamentales. 2.1.1. Abstracción 2.1.2. Encapsulamiento y ocultación de datos. 2.1.3. Herencia Reutilización o reusabilidad. 2.1.4. Polimorfismo. 2.2. Clases y objetos. 2.2.1. Tipo abstracto de datos. 2.2.2. Identificación de objetos. 2.2.3. Estado de un objeto. 2.2.4. Comportamiento. 2.2.5. Identidad. 2.2.6. Declaración e instancia de un objeto. 2.2.7. Mensajes. 2.1.1 Reglas de visibilidad.	1. Distingue conceptos fundamentales de la programación orientada a objetos para la resolución de problemas. 2. Identifica objetos según la forma de abstracción permitida en la programación orientada a objetos. 3. Reconoce la información y el comportamiento de un objeto para representarlos en una clase.
3. Análisis y Diseño orientado a objetos.	3.1 Especificaciones de UML. 3.1.1. Diseño y representación. gráfica de objetos en UML. 3.1.2. Diseño y representación. gráfica de clases en UML. 3.1.3. Proceso de análisis orientado a objetos. 3.1.4. Indagación de los requerimientos.	1 Identifica las características de un sistema utilizando diagramas UML. 2 Explica los esquemas de un modelo orientado a objetos con las notaciones correspondientes. 3 Formula soluciones de software con arquitectura empresarial mediante análisis y diseño. 4 Colabora en equipo compartiendo

Unidades	Contenidos	Resultados de aprendizaje
	3.1.5. Desarrollo de casos de uso. 3.1.6. Elaboración del modelo de requerimientos. 3.1.7. Modelado de los requerimientos con UML. 3.2 Proceso de diseño orientado a objetos. 3.3 Conceptos de diseño orientado a objetos. 3.3.1 Modelado del diseño con UML. 3.3.2 Conceptos de mapeo relacional.	responsabilidades y tareas de manera equitativa en el desarrollo de software. 5 Diseña programas aplicando los conceptos de herencia, interfaces y encapsulamiento.
4. Programación orientada a objetos.	4.1. Programación orientada a objetos. 4.1.1. Declaración de una clase. 4.1.2. Implementación de las clases. 4.1.3. Palabras reservadas. 4.1.4. Tipos de datos. Miembros de una clase 4.1.5. Métodos. 4.1.6. Constructores. 4.1.7. Utilización de métodos de librerías. 4.1.8. Comentarios. 4.2. Instrucciones de control 4.2.1. Estructuras de control. 4.2.2. Estructuras de repetición. 4.2.3. Instrucción de selección múltiple. 4.2.4. Operadores lógicos. 4.3. Herencia y Polimorfismo. 4.3.1. Tipos de herencia. 4.3.2. Clases y métodos abstractos. 4.3.3. Métodos abstractos. 4.3.4. Usos y aplicaciones del polimorfismo. 4.3.5. Sobreescritura de métodos 4.4. Arreglos. 4.4.1. Declaración y creación de arreglos. 4.4.2. Paso de arreglos a los métodos. 4.4.3. Arreglos multidimensionales. 4.4.4. Colecciones. 4.4.5. Genericidad 4.5. Manejo de excepciones. 4.5.1. Excepciones y errores. 4.5.2. Bloque try - catch. 4.5.3. Excepciones definidas en el lenguaje. 4.5.4. Lanzar y atrapar excepciones. 4.5.5. Declaración de nuevos tipos de excepciones.	1. Identifica las palabras reservadas del lenguaje para su implementación. 2. Diseña jerarquías de clases orientadas a la solución de problemas con software. 3. Aplica técnicas de programación adecuadas al paradigma orientado a objetos. 4. Identifica excepciones comunes en la implementación de código en un lenguaje orientado a objetos. 5. Presenta soluciones multitarea utilizando bibliotecas del lenguaje. 6. Colabora en equipo compartiendo responsabilidades y tareas de manera equitativa en el desarrollo de software. 7. Implementa programas que integren instrucciones de control, herencia, polimorfismo, arreglos y manejo de excepciones. 8. Desarrolla aplicaciones que ejecuten múltiples hilos de manera eficiente, aplicando conceptos de concurrencia.

Unidades	Contenidos	Resultados de aprendizaje
	4.6. Multitarea. 4.6.1. Utilización de la multitarea. 4.6.2. Creación de hilos. 4.6.3. Estados de un hilo. 4.6.4. Prioridad entre hilos. 4.6.5. Sincronización y Concurrencia. 4.6.6. Implementación de hilos.	

IV. 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.

V. ESTRATEGIAS EVALUATIVAS

Procesos de producción grupales e individuales, pruebas individuales orales y/o escritas durante el desarrollo de las unidades con diálogos e interpretaciones que los estudiantes realicen sobre los contenidos, debates, retroalimentación en casos necesarios y actividades que amplíen el conocimiento, que 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.

VI. MEDIOS AUXILIARES

Aula virtual, pizarrón, proyector, marcadores, ordenadores, wifi, celulares, salas de chats, correo electrónico.

VII. BIBLIOGRAFÍA

- Deitel, H. M. (2016). Java: como programar.
- Deitel, P., Deitel, H. & Elizondo, A. (2008). Como programar en Java. Mexico, D.F: Pearson Ed
- Sznajdleder, P. (2017). Programación orientada a objetos y estructura de datos a fondo. Alpha Editorial.
- Joyanes, L. (2010). Programación en C, C++, JAVA y UML. México DF.
- López, L. (2013). Metodología de la programación orientada a objetos. Alpha Editorial.
- Pressman, R. S. (2010). Ingeniería del Software-Un enfoque práctico. 7ª Edición McGraw-Hil.
- Barnes, D. J., Kölling, M., &Brenta, B. I. (2007). Programación orientada a objetos con Java. Pearson Educación.
- Goytia, L. L., & González, Á. G. (2014). Programación orientada a objetos C++ y Java. Grupo

Editorial Patria.

- Groussard, T. (2014). JAVA 8: Los fundamentos del lenguaje Java (con ejercicios practices corregidos). Ediciones Eni.
- Sierra, F. J. C. (2018). Programación orientada a objetos con C++. Ra-Ma Editorial.
- Putier, S. (2015). C# 6 y Visual Studio 2015: Los fundamentos del lenguaje. Ediciones Eni.