|  |  |  |
| --- | --- | --- |
|  | **PROGRAMA EDUCATIVO:** **LICENCIATURA EN INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN E INNOVACIÓN DIGITAL** **EN COMPETENCIAS PROFESIONALES** |  |

**PROGRAMA DE ASIGNATURA:** \_\_**FUNDAMENTOS DE PROGRAMACIÓN**\_\_\_\_\_\_\_\_\_\_\_\_ **CLAVE:­­­**\_\_\_\_**FUP**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
| Propósito de aprendizaje de la Asignatura | El estudiante desarrollará soluciones algorítmicas de manera efectiva, a través del análisis de requisitos, diseño modular y uso de metodologías de programación estructurada con el fin de resolver problemas específicos. |
| Competencia a la que contribuye la asignatura | Desarrollar soluciones tecnológicas a través de lenguajes de programación estructurada, programación orientada a objetos y de consulta, herramientas de desarrollo asistido de software, usabilidad y pruebas, fundamentos de redes de área local, sistemas operativos, medidas de seguridad informática para contribuir a la eficiencia y productividad en diferentes contextos con un enfoque de impulso al desarrollo social, ambiental y de economía socialmente responsable. |
|  Tipo de competencia | Cuatrimestre | Créditos | Modalidad | Horas por semana | Horas Totales |
|
| **Específica** | **1** | **3.75** | **Escolarizada** | **4** | **60** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Unidades de Aprendizaje** | **Horas del Saber** | **Horas del Saber Hacer** | **Horas Totales** |
|  |  |  |
| 1. Algoritmos
 | 6 | 10 | 16 |
| 1. Estructuras de control
 | 8 | 12 | 20 |
| 1. Metodologías y herramientas del desarrollo del programa
 | 10 | 14 | 24 |
| **Totales** | **24** | **36** | **60** |

|  |  |  |
| --- | --- | --- |
| **Funciones** | **Capacidades** | **Criterios de Desempeño** |
| Implementar soluciones básicas de software utilizando lenguajes de programación estructurada, orientada a objetos y de consulta, aplicando herramientas básicas de desarrollo de software como entornos de desarrollo para contribuir a satisfacer las necesidades de la organización. | Diseñar aplicaciones básicas de software utilizando algoritmos, diagramas de flujo y casos de uso para la representación de la lógica de negocio, aplicando principios básicos de diseño funcional y seleccionando lenguajes de programación y herramientas de desarrollo de software adecuados a las necesidades y requerimientos del proyecto. | Diseña diagramas funcionales que representen la lógica de negocio de una aplicación básica, considerando: algoritmos, diagramas de flujo y casos de uso.Propone interfaces de usuario con elementos básicos de usabilidad. Utiliza lenguajes de programación y herramientas de desarrollo de software de acuerdo a las necesidades del proyecto. |
| Codificar aplicaciones básicas de software utilizando lenguajes de programación estructurada, orientada a objetos y de consulta, empleando herramientas básicas de desarrollo de software en diversos entornos de desarrollo. | Codifica aplicaciones básicas de software utilizando lenguajes de programación estructurada y orientada a objetos a través de un código documentado con las siguientes características:- Integración del diseño de la aplicación: algoritmo, diagrama de flujo y casos de uso.- Utilizando estándares y técnicas de codificación y documentación. |
| Evaluar aplicaciones básicas de software aplicando pruebas para la detección y corrección de errores para asegurar su correcto funcionamiento. | Ejecuta pruebas de software para detectar y corregir errores.Documenta los resultados de las pruebasAsegura el cumplimiento de los criterios de éxito con base en los requerimientos. |

**UNIDADES DE APRENDIZAJE**

|  |  |
| --- | --- |
| Unidad de Aprendizaje | 1. Algoritmos
 |
| Propósito esperado  | El estudiante diseñará algoritmos secuenciales para resolver problemas específicos de manera precisa. |
| **Tiempo Asignado** | **Horas del Saber**  | 6 | **Horas del Saber Hacer** | 10 | **Horas Totales** | 16 |

| **Temas** | **Saber****Dimensión Conceptual** | **Saber Hacer****Dimensión Actuacional** | **Ser y Convivir****Dimensión Socioafectiva** |
| --- | --- | --- | --- |
| Metodología para la solución de problemas. | Explicar la metodología para la solución de problemas.  |  | Cultivar la perseverancia y la paciencia al enfrentar desafíos en la resolución de problemas algorítmicos, reconociendo que el proceso de aprendizaje puede requerir tiempo y esfuerzo. |
| Conceptos básicos de algoritmos | Describir los conceptos básicos de algoritmos. |   |
| Notación para la representación de algoritmos | Identificar la notación utilizada en la representación de algoritmos, como diagramas de flujo y pseudocódigos. | Diseñar algoritmos secuenciales utilizando diagramas de flujo y pseudocódigo. |
| Tipos de datos básicos | Diferenciar entre tipos de datos básicos (enteros, reales, cadenas de caracteres y valores lógicos) utilizados en algoritmos. | Estructurar algoritmos que hagan uso de tipos de datos básicos como enteros, reales, cadenas de caracteres y valores lógicos. |
| Variables y constantes | Definir el concepto de variables y constantes en el contexto de la programación. | Seleccionar correctamente variables y constantes en el diseño de algoritmos. |
| Operadores y expresiones | Describir los diferentes tipos de operadores (aritméticos, de comparación y lógicos) y su orden de precedencia.Describir las expresiones utilizadas en la construcción de algoritmos. | Determinar la secuencia de operadores y expresiones necesarios para resolver un problema algorítmico específico. |
| Estructura básica de un algoritmo secuencial | Explicar la estructura básica de un algoritmo secuencial y su importancia en la resolución de problemas. | Validar la estructura básica de un algoritmo secuencial, verificando su coherencia y precisión. |

|  |
| --- |
| **Proceso Enseñanza-Aprendizaje** |
| Métodos y técnicas de enseñanza | Medios y materiales didácticos | Espacio Formativo |
| **Aula** | X |
| Instrucción programadaEquipos cooperativosResolver situaciones problemáticas | ProyectorPizarrónBibliografíaBuscadores académicosLecciones en un LMSAcceso a internetSoftware de interpretación de pseudocódigo y/o diagramas de flujo. | **Laboratorio / Taller** |  |
|  |  | **Empresa** |  |

|  |
| --- |
| **Proceso de Evaluación** |
| Resultado de Aprendizaje | Evidencia de Aprendizaje | Instrumentos de evaluación |
| Los estudiantes identifican la aplicación y uso de algoritmos secuenciales para resolver problemas específicos, demostrando un dominio de los conceptos básicos de algoritmos, notación para su representación y estructuras de control secuenciales. | A partir de un portafolio de evidencias de prácticas, desarrollar algoritmos secuenciales para resolver problemas específicos, presentando un informe técnico que incluya diagramas de flujo y pseudocódigos, así como una explicación detallada de las variables y expresiones utilizadas. | Ejercicios prácticosRúbrica |

|  |  |
| --- | --- |
| Unidad de Aprendizaje | 1. Estructuras de control
 |
| Propósito esperado  | El estudiante diseñará algoritmos que utilicen estructuras de control selectivas y repetitivas para diseñar soluciones efectivas en diferentes contextos. |
| **Tiempo Asignado** | **Horas del Saber**  | 8 | **Horas del Saber Hacer** | 12 | **Horas Totales** | 20 |

| **Temas** | **Saber****Dimensión Conceptual** | **Saber Hacer****Dimensión Actuacional** | **Ser y Convivir****Dimensión Socioafectiva** |
| --- | --- | --- | --- |
| Decisiones compuestas, anidadas y múltiples | Describir las características y usos de las decisiones compuestas, anidadas y múltiples en la construcción de algoritmos. | Diseñar decisiones compuestas, anidadas y múltiples | Incentivar la creatividad al explorar diferentes enfoques para la solución de problemas algorítmicos, valorando la diversidad de ideas y perspectivas. |
| Representación de algoritmos con estructuras de control selectivas | Explicar diferentes tipos de estructuras de control selectivas.Describir la representación gráfica de las estructuras de control selectivas. | Diseñar algoritmos que hagan uso de decisiones compuestas, anidadas y múltiples |
| Conceptos básicos de contador, acumulador y bandera | Distinguir entre los conceptos de contador, acumulador y bandera | Seleccionar el tipo de variable de control adecuada para una estructura de control repetitiva. |
| Estructura Mientras - FinMientras | Explicar la estructura y el funcionamiento de la instrucción Mientras - FinMientras | Diseñar algoritmos que hagan uso de las estructuras Mientras - FinMientras para ejecutar tareas repetitivas. |
| Estructura Haga - MientrasQue | Explicar la estructura y el funcionamiento de la instrucción Haga - MientrasQue | Diseñar algoritmos que hagan uso de las estructuras Haga - MientrasQue para ejecutar tareas repetitivas. |
| Estructura Para - FinPara | Describir la estructura de la instrucción Para - FinPara y su utilidad en la ejecución de algoritmos repetitivos. | Diseñar algoritmos que hagan uso de las estructuras Para - FinPara para ejecutar tareas repetitivas |
| Representación de algoritmos con estructuras de control repetitivas | Describir la representación gráfica de las estructuras de control repetitivas. | Diseñar diagramas de flujo que utilicen estructuras de control repetitivas |

|  |
| --- |
| **Proceso Enseñanza-Aprendizaje** |
| Métodos y técnicas de enseñanza | Medios y materiales didácticos | Espacio Formativo |
| **Aula** |  |
| Aula invertidaResolver situaciones problemáticasEquipos cooperativos | ProyectorPizarrónEquipo de cómputoVideosPlataformas digitalesAcceso a internetSoftware de interpretación de pseudocódigo y/o diagramas de flujo. | **Laboratorio / Taller** | X |
|  |  | **Empresa** |  |

|  |
| --- |
| **Proceso de Evaluación** |
| Resultado de Aprendizaje | Evidencia de Aprendizaje | Instrumentos de evaluación |
| Los estudiantes comprenden y analizan algoritmos que utilizan estructuras de control selectivas y repetitivas, demostrando habilidades en el diseño de soluciones efectivas. | A partir de la resolución de casos prácticos, desarrollar algoritmos que utilicen estructuras de control selectivas y repetitivas para resolver problemas reales, presentando diagramas de flujo y pseudocódigos bien documentados y un análisis detallado de su funcionamiento. | Estudios de casosRúbrica |

|  |  |
| --- | --- |
| Unidad de Aprendizaje | 1. Metodologías y herramientas del desarrollo del programa
 |
| Propósito esperado  | El estudiante implementará algoritmos en pseudocódigo, con el fin de desarrollar proyectos que resuelvan problemas específicos, demostrando habilidades en la planificación, diseño y depuración de algoritmos antes de su implementación final. |
| **Tiempo Asignado** | **Horas del Saber**  | 10 | **Horas del Saber Hacer** | 14 | **Horas Totales** | 24 |

| **Temas** | **Saber****Dimensión Conceptual** | **Saber Hacer****Dimensión Actuacional** | **Ser y Convivir****Dimensión Socioafectiva** |
| --- | --- | --- | --- |
| Sub-algoritmos | Diferenciar entre los distintos tipos de sub-algoritmos, como procedimientos y funciones y su aplicación en la modularización de programas. | Seleccionar el tipo de sub-algoritmos necesarios para la solución de un problema. | Valorar la importancia del trabajo colaborativo al desarrollar algoritmos, reconociendo la contribución de cada miembro del equipo en la resolución de problemas. |
| Proceso de desarrollo de programas estructurados | Describir el proceso de desarrollo de programas, identificando sus etapas principales y su importancia en la creación de soluciones de software. | Diseñar sub-algoritmos mediante procedimientos y funciones |
| Principios básicos de diseño funcional | Explicar los principios básicos del diseño funcional, destacando la importancia de la estructura lógica y la modularidad. | Implementar algoritmos en pseudocódigos modulares y estructurados. |
| Técnicas de recolección de requerimientos. | Describir técnicas de recolección de requerimientos utilizadas para identificar y documentar las necesidades de los usuarios. | Identificar y documentar necesidades de los usuarios en el desarrollo de programas. |
| Técnicas de prueba y depuración. | Identificar las actividades realizadas durante la etapa de prueba de un pseudocódigo y su papel en la detección de errores.Describir el proceso de depuración de un pseudocódigo, incluyendo técnicas y herramientas utilizadas para identificar y corregir errores. | Diseñar casos de prueba que permitan evaluar el funcionamiento y la robustez de los pseudocódigos desarrolladosDepurar pseudocódigos identificando y corrigiendo errores de lógica, sintaxis o ejecución. |

|  |
| --- |
| **Proceso Enseñanza-Aprendizaje** |
| Métodos y técnicas de enseñanza | Medios y materiales didácticos | Espacio Formativo |
| **Aula** |  |
| Aprendizaje basado en proyectosAprendizaje cooperativoForo | ProyectorPizarrónEquipo de cómputoBibliografíaBuscadores académicosAcceso a internetSoftware de interpretación de pseudocódigo y/o diagramas de flujo. | **Laboratorio / Taller** | X |
|  |  | **Empresa** |  |

|  |
| --- |
| **Proceso de Evaluación** |
| Resultado de Aprendizaje | Evidencia de Aprendizaje | Instrumentos de evaluación |
| Los estudiantes identifican la aplicación de algoritmos en pseudocódigo y el uso de técnicas de modularización y estructuración, aplicando los principios del diseño funcional y las técnicas de desarrollo de programas estructurados, demostrando habilidades para identificar y documentar las necesidades de los usuarios, así como para diseñar casos de prueba y depurar pseudocódigos para garantizar su funcionalidad y robustez. | A partir de la elaboración de un proyecto práctico, desarrollar un conjunto de algoritmos modulares en pseudocódigo con el objetivo de resolver un problema específico, documentando en un informe técnico que incluya la descripción detallada de los algoritmos desarrollados, su estructura modular, casos de prueba diseñados para evaluar su funcionalidad y la depuración realizada para corregir posibles errores lógicos o de sintaxis. | Proyectos grupales y/o individualesEvaluación de desempeño |

|  |
| --- |
| **Perfil idóneo del docente** |
| **Formación académica** | **Formación Pedagógica** | **Experiencia Profesional** |
| Licenciatura en Ingeniería en Sistemas Computacionales, Ingeniería Informática, Ingeniería de Software o carrera afín. | Dominio de técnicas de enseñanza-aprendizaje adaptadas a la enseñanza de la programación.Habilidad para explicar conceptos complejos de manera clara y concisa.Capacidad para fomentar la participación activa de los estudiantes en el proceso de aprendizaje.Habilidad para fomentar el pensamiento lógico de los estudiantes. | Deseable experiencia previa como desarrollador de software o ingeniero de software.Participación en proyectos de desarrollo de software en la industria.Certificaciones o cursos relacionados con la enseñanza de la programación o el desarrollo de software. |

|  |
| --- |
| **Referencias bibliográficas** |
| Autor | Año | Título del documento | Lugar de publicación | Editorial | ISBN |
| Thomas H. Cormen, Charles E. Leiserson, Ronald Rivest, Clifford Stein | 2022 | Introduction to Algorithms, fourth edition | Estados Unidos | MIT Press | 9780262046305 |
| Omar Iván Trejos Buriticá | 2022 | Lógica de programación | Colombia | Ediciones de la U | 9789587627206 |
| Sebastián Rubén Gómez Palomo, Eduardo Moraleda Gil | 2020 | Aproximación a la ingeniería del software | España | Centro de Estudios Ramón Areces | 9788499613291 |
| Gregorio de Miguel Casado, Jorge Júlvez Bueno, Jorge Gracia del Río | 2021 | Introducción a la programación C++ para ingenieros | España | Prensas de la Universidad de Zaragoza | 9788413404219 |
| López Franco, Diego, ‎Guerrero Mendieta, Luz Edith, ‎Villegas Jaramillo, Eduardo | 2024 | Algoritmos en Entornos Virtuales Educativos | Colombia | Universidad de Caldas | 9789587595970 |

|  |
| --- |
| **Referencias digitales** |
| Autor | Fecha de recuperación | Título del documento | Vínculo |
| Ruiz, C. E. y Ávila, A. | Abril, 2024 | Etapas de la metodología de solución de problemas. En Metodología de resolución de problemas. | [https://portalacademico.cch.unam.mx/cibernetica1/metodología-resolución-problemas/etapas-solución](https://portalacademico.cch.unam.mx/cibernetica1/metodolog%C3%ADa-resoluci%C3%B3n-problemas/etapas-soluci%C3%B3n)  |
| Lucid Software Español | Abril, 2024 | Cómo hacer un diagrama de flujo - Tutorial (Paso a Paso) | <https://youtu.be/IffiTrRZR4E?si=-xOXKb1E20bXXeX4>  |
| Marcos Prunello | Abril, 2024 | Introducción a la Programación: Guía de estudio - Lic. en Estadística - FCEyE - UNR | <https://mpru.github.io/introprog/index.html>  |
| Juan Carlos Amaranto González | Abril, 2024 | Algoritmos desde Cero | <https://www.udemy.com/course/algoritmos-desde-cero/>  |
| José Domingo Muñoz Rodríguez | Abril, 2024 | Curso de introducción a la programación con pseudocódigo | <https://github.com/josedom24/curso_programacion.git>  |
| Pablo Novara | Abril, 2024 | Manuales y Documentación de PSeInt | <https://pseint.sourceforge.net/index.php?page=documentacion.php>  |