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

**PROGRAMA DE ASIGNATURA: PROGRAMACIÓN ESTRUCTURADA CLAVE: PRE .**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Propósito de aprendizaje de la Asignatura | | El estudiante desarrollará aplicaciones de software a través de técnicas de programación estructurada para implementar soluciones computacionales. | | | | |
| 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** | **2** | | **4.6875** | **Escolarizada** | **5** | **75** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Unidades de Aprendizaje** | **Horas del Saber** | **Horas del Saber Hacer** | **Horas Totales** |
|  |  |  |
| 1.- Introducción a la programación estructurada. | 4 | 6 | 10 |
| 2.- Estructuras de control | 6 | 9 | 15 |
| 3.- Subprogramas y recursividad | 10 | 15 | 25 |
| 4.- Arreglos y archivos | 10 | 15 | 25 |
| **Totales** | **30** | **45** | **75** |

|  |  |  |
| --- | --- | --- |
| **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. |
| 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. |
| 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. |
| Ejecuta pruebas de software para detectar y corregir errores.  Documenta los resultados de las pruebas. |
| Evaluar aplicaciones básicas de software aplicando pruebas para la detección y corrección de errores para asegurar su correcto funcionamiento. |
| Asegura el cumplimiento de los criterios de éxito con base en los requerimientos. |

**UNIDADES DE APRENDIZAJE**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Unidad de Aprendizaje | 1.- Introducción a la programación estructurada. | | | | | |
| Propósito esperado | El estudiante determinará los paradigmas, herramientas y técnicas para identificar los elementos de buenas prácticas de la programación. | | | | | |
| **Tiempo Asignado** | **Horas del Saber** | 4 | **Horas del Saber Hacer** | 6 | **Horas Totales** | 10 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Temas** | **Saber**  **Dimensión Conceptual** | **Saber Hacer**  **Dimensión Actuacional** | **Ser y Convivir**  **Dimensión Socioafectiva** |
| Clasificación de paradigmas de programación | Identificar las distintas clasificaciones, características, diferencias y aplicaciones de los paradigmas de programación. - Paradigmas imperativos:  - Estructurado.  - Procedimental.  - Modular. - Paradigmas declarativos:  - Lógico.  - Funcional. | Proponer el empleo de paradigmas de programación con base en las problemáticas presentadas. | Fomentar la habilidad de investigación de manera individual y en equipo en la selección de los paradigmas de programación y configuración de los entornos de desarrollo para alcanzar soluciones funcionales.  Promover la gestión de información en equipos de desarrollo identificando los procesos de buenas prácticas de codificación y documentación para mejorar la calidad del desarrollo de soluciones computacionales.  Asumir la ética y responsabilidad de forma individual en las acciones de su entorno. |
| Entornos de desarrollo integrados. | Identificar los diferentes procesos de desarrollo en la Programación estructurada: - Tipos de Lenguaje. - Fases de Implementación. - SDKs | Proponer procesos de desarrollo en la programación estructurada. |
| Describir las funcionalidades, componentes internos y externos de los entornos de desarrollo integrados: - Editores de código. - Herramientas de depuración. - Herramientas de control de versiones. - Plugin/Extensiones. | Configurar un entorno de desarrollo integrado para optimizar el proceso de desarrollo de software. |
| Buenas prácticas de codificación. | Identificar las buenas prácticas de codificación y documentación que contribuyen a la legibilidad y mantenibilidad del código de soluciones computacionales. | Implementar buenas prácticas de codificación y documentación en proyectos de desarrollo de software. |
| Documentación de código. | Explicar el uso de las herramientas generativas de programación (Prompt Engineering en programación) como herramientas de apoyo a la programación estructurada. | Emplear herramientas generativas como apoyo al desarrollo de proyectos de software. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Proceso Enseñanza-Aprendizaje** | | | |
| Métodos y técnicas de enseñanza | Medios y materiales didácticos | Espacio Formativo | |
| **Aula** |  |
| Investigación con tutoría  Instrucción programada  Aula invertida  Aprendizaje basado en problemas/soluciones  Resolver situaciones problemáticas | Pintarrones/Pizarras electrónicas  Proyectores  acceso Internet  Plataformas Educativas  Equipos de cómputo.  Bibliografía | **Laboratorio / Taller** | X |
| **Empresa** |  |

|  |  |  |
| --- | --- | --- |
| **Proceso de Evaluación** | | |
| Resultado de Aprendizaje | Evidencia de Aprendizaje | Instrumentos de evaluación |
| Los estudiantes identifican los diferentes usos y aplicaciones de los paradigmas de programación, así como también, configuran un entorno de desarrollo integrado e implementan elementos de buenas prácticas de codificación y documentación de software. | A partir de una investigación con tutoría analizar y construir un organizador gráfico de los diferentes paradigmas de programación que incluyan sus principales características, ventajas, funcionalidades, aplicaciones, etc. | Rúbricas o matrices de Valoración.  Lista de cotejo. |
| A partir de una investigación con tutoría analizar y desarrollar un reporte que incluya las técnicas de buenas prácticas y documentación de código de programación computacional. |
| A partir de un aprendizaje basado en problemas/soluciones desarrollar un reporte que incluya el procedimiento de instalación y configuración de un entorno de desarrollo integrado. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Unidad de Aprendizaje | 2.- Estructuras de control. | | | | | |
| Propósito esperado | El estudiante empleará estructuras de control para el desarrollo de soluciones computacionales en programación estructurada. | | | | | |
| **Tiempo Asignado** | **Horas del Saber** | 6 | **Horas del Saber Hacer** | 9 | **Horas Totales** | 15 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Temas** | **Saber**  **Dimensión Conceptual** | **Saber Hacer**  **Dimensión Actuacional** | **Ser y Convivir**  **Dimensión Socioafectiva** |
| Tipos de datos y funciones de entrada-salida(I/O) en la programación estructurada. | Identificar las características de los datos primitivos, extendidos y su representación en lenguajes de programación estructurada y sus funciones de entrada-salida(I/O):  - Espacio de memoria de cada tipo de dato. - Rango de valores de cada tipo de dato.  - Conversión entre tipos de datos. - Identificadores. - Variables. - Constantes. - Palabras reservadas. |  | Cultivar la capacidad de razonamiento crítico, lógico y matemático en resolución de problemas aplicando estructuras de control selectivas e iterativas para la resolución de problemas.  Fomentar la colaboración y la comunicación efectiva en equipos de desarrollo de soluciones computacionales, reconociendo la importancia del trabajo en conjunto para alcanzar soluciones funcionales.  Incentivar la creatividad al explorar diferentes enfoques para la solución de problemas computacionales, valorando la diversidad de ideas y perspectivas.  Asumir la ética y responsabilidad de forma individual en las acciones de su entorno. |
| Estructuras de control Selectivas e Iterativas | Describir las características de las estructuras de control selectivas e iterativas en la codificación de programas: - Condicional simple. - Condicional doble. - Condicional múltiple. - Condicional anidada. - Repetición (For). - Mientras (While). - Condiciones Anidados. |  |
| Sintaxis de un lenguaje de programación estructurada | Explicar la sintaxis de las estructuras de control selectivas e iterativas en la codificación de programas. | Codificar estructuras de control selectivas e iterativas utilizando la sintaxis de programación estructurada. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Proceso Enseñanza-Aprendizaje** | | | |
| Métodos y técnicas de enseñanza | Medios y materiales didácticos | Espacio Formativo | |
| **Aula** |  |
| Investigación con tutoría  Instrucción programada  Aula invertida  Aprendizaje basado en problemas/soluciones  Resolver situaciones problemáticas | Pintarrones/Pizarras electrónicas  Proyectores  acceso Internet  Plataformas Educativas  Equipos de cómputo.  Bibliografía | **Laboratorio / Taller** | X |
| **Empresa** |  |

|  |  |  |
| --- | --- | --- |
| **Proceso de Evaluación** | | |
| Resultado de Aprendizaje | Evidencia de Aprendizaje | Instrumentos de evaluación |
| Los estudiantes integran las estructuras de control selectivas e iterativas y aplican correctamente la sintaxis de programación estructurada en el desarrollo de soluciones computacionales lógicas y eficientes. | A partir de aprendizaje basado en problemas/soluciones conformar un portafolios de evidencias que contenga:  - Planteamiento de ejercicios  - Códigos Fuentes de la Solución.  - Corridas o pruebas de escritorio de la solución. | Rúbricas o matrices de Valoración.  Lista de cotejo. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Unidad de Aprendizaje | 3.- Funciones y procedimientos recursividad. | | | | | |
| Propósito esperado | El estudiante empleará estructuras de control para el desarrollo de soluciones computacionales en programación estructurada. | | | | | |
| **Tiempo Asignado** | **Horas del Saber** | 10 | **Horas del Saber Hacer** | 15 | **Horas Totales** | 25 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Temas** | **Saber**  **Dimensión Conceptual** | **Saber Hacer**  **Dimensión Actuacional** | **Ser y Convivir**  **Dimensión Socioafectiva** |
| Describir las características de los tipos de subprogramas: - Funciones. - Procedimientos. | Describir las características de los tipos de subprogramas: - Funciones. - Procedimientos. |  | Cultivar la capacidad de razonamiento crítico, lógico y matemático en resolución de problemas complejos aplicando procesos recursivos para incrementar la funcionalidad de las soluciones computacionales.  Incentivar la creatividad al explorar diferentes enfoques para la solución de problemas computacionales, valorando la diversidad de ideas y perspectivas.  Asumir la ética y responsabilidad de forma individual en las acciones de su entorno. |
| Explicar la sintaxis de los tipos de subprogramas en lenguaje de programación estructurada. | Explicar la sintaxis de los tipos de subprogramas en lenguaje de programación estructurada. | Codificar soluciones que empleen funciones y procedimientos en la programación estructurada. |
| Describir el concepto de recursividad. | Describir el concepto de recursividad. |  |
| Explicar características de los procesos recursivos. | Explicar características de los procesos recursivos. | Codificar soluciones recursivas en problemas complejos en la programación estructurada. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Proceso Enseñanza-Aprendizaje** | | | |
| Métodos y técnicas de enseñanza | Medios y materiales didácticos | Espacio Formativo | |
| **Aula** |  |
| Investigación con tutoría  Instrucción programada  Aula invertida  Aprendizaje basado en problemas/soluciones  Resolver situaciones problemáticas | Pintarrones/Pizarras electrónicas  Proyectores  acceso Internet  Plataformas Educativas  Equipos de cómputo.  Bibliografía | **Laboratorio / Taller** | X |
| **Empresa** |  |

|  |  |  |
| --- | --- | --- |
| **Proceso de Evaluación** | | |
| Resultado de Aprendizaje | Evidencia de Aprendizaje | Instrumentos de evaluación |
| Los estudiantes diseñan y aplican funciones, procedimientos y técnicas recursivas en la resolución de problemas computacionales complejos. | A partir de aprendizaje basado en problemas/soluciones conformar un portafolios de evidencias que contenga:  - Planteamiento del ejercicio Implementando.  - Funciones y Procedimientos.  - Recursividad  - Códigos Fuentes de la Solución.  - Corridas o pruebas de escritorio de la solución. | Rúbricas o matrices de Valoración.  Lista de cotejo. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Unidad de Aprendizaje | 4.- Funciones y procedimientos recursividad. | | | | | |
| Propósito esperado | El estudiante empleará estructuras de control para el desarrollo de soluciones computacionales en programación estructurada. | | | | | |
| **Tiempo Asignado** | **Horas del Saber** | 10 | **Horas del Saber Hacer** | 15 | **Horas Totales** | 25 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Temas** | **Saber**  **Dimensión Conceptual** | **Saber Hacer**  **Dimensión Actuacional** | **Ser y Convivir**  **Dimensión Socioafectiva** |
| Arreglos | Definir el concepto, tipos, características y operaciones de arreglos. |  | Cultivar la capacidad de razonamiento crítico, lógico y matemático en resolución de problemas complejos aplicando arreglos y archivos para el control de datos e incrementar la funcionalidad de las soluciones computacionales.  Incentivar la creatividad al explorar diferentes enfoques para la solución de problemas computacionales, valorando la diversidad de ideas y perspectivas.  Asumir la ética y responsabilidad de forma individual en las acciones de su entorno. |
| Explicar la sintaxis de los arreglos en la programación estructurada. | Codificar operaciones con arreglos en la programación estructurada. |
| Organización y operaciones con archivos | Describir el concepto, tipos, características y operaciones con archivos |  |
| Explicar las operaciones con archivos. | Codificar operaciones con archivos en la programación estructurada. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Proceso Enseñanza-Aprendizaje** | | | |
| Métodos y técnicas de enseñanza | Medios y materiales didácticos | Espacio Formativo | |
| **Aula** |  |
| Investigación con tutoría  Instrucción programada  Aula invertida  Aprendizaje basado en problemas/soluciones  Resolver situaciones problemáticas | Pintarrones/Pizarras electrónicas  Proyectores  acceso Internet  Plataformas Educativas  Equipos de cómputo.  Bibliografía | **Laboratorio / Taller** | X |
| **Empresa** |  |

|  |  |  |
| --- | --- | --- |
| **Proceso de Evaluación** | | |
| Resultado de Aprendizaje | Evidencia de Aprendizaje | Instrumentos de evaluación |
| Los estudiantes aplican arreglos y/o archivos en la resolución de problemas computacionales complejos. | A partir de aprendizaje basado en problemas/soluciones conformar un portafolios de evidencias que contenga:  - Planteamiento de los ejercicios implementando:  - Funciones y procedimientos.  - Recursividad.  - Códigos fuentes de la solución.  - Corridas o pruebas de escritorio de la solución. | Proyectos grupales y/o individuales  Rúbricas o matrices de valoración |

|  |  |  |
| --- | --- | --- |
| **Perfil idóneo del docente** | | |
| **Formación académica** | **Formación Pedagógica** | **Experiencia Profesional** |
| Ing. Tecnologías de La información  Ing. Sistemas Computacionales  Ing. Desarrollo de Software  Lic. Informática, Ing. Computación  Posgrados a fin a las áreas de Tic´s. | Manejo de herramientas didácticas para enseñanza-aprendizaje, de evaluación, técnicas de manejo de grupos. | Desarrollador de Software  Probador de Software  Analista de Sistemas. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Referencias bibliográficas** | | | | | |
| Autor | Año | Título del documento | Lugar de publicación | Editorial | ISBN |
| Allen B. Downey, Chris Mayfield | 2019 | Think Java How To Think Like A Computer Scientist | Usa | O'reilly Media | 978-149-207-247-8 |
| Luis Joyanes Aguilar | 2022 | Fundamentos De Programación Algoritmos, Estructura De Datos Y Objetos | España | Mcgraw Hill | 978-607-151-468-4 |
| Alfonso Jiménez Marín, Francisco Manuel Pérez Montes | 2021 | Programación | México | Ediciones Paraninfo S.A | 978-842-834-286-5 |
| Mihaela Juganaru Mathieu | 2014 | Introducción A La Programación | México | Patria | 978-607-438-920-3 |
| Edgar Danilo Domínguez Vera | 2014 | Programación: Estructurada Raptor Y Lenguaje | México | Alfaomega | 978-607-622-168-6 |
| Cairó Battistutti, O | 2022 | Aprende A Programar En Java: De Cero Al Infinito | España | Marcombo | 9788426735058, 8426735053 |
| Sznajdleder, P. A. | 2024 | Java A Fondo. Curso De Programación | España | Marcombo | 9788426738325, 842673832X |
| Claudio Arroyo Díaz | 2019 | Programación En Java I | Argentina | Redusers | 9874958049, 9789874958044 |
| Nadal, M. | 2021 | Curso De Programación Java | España | Anaya Multimedia | 9788441544253, 8441544255 |
| Vegas, J. M. | 2022 | JAVA 17: Fundamentos Prácticos De Programación | Colombia | Ediciones De La U | 9789587924114, 9587924118 |
| Guardati Buemo, S., Cairó Battistutti, O. | 2022 | Aprende a programar en Python: de cero al infinito | España | Marcombo | 9788426735553, 842673555X |
| Ángel Pablo Hinojosa Gutiérrez | 2020 | Python paso a paso | España | RA-MA S.A. Editorial y Publicaciones | 8499646115, 978-8499646114 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Referencias digitales** | | | |
| Autor | Fecha de recuperación | Título del documento | Vínculo |
| MIT OpenCourseWare,45039 | 01/05/2024 | Introduction To Programming in Java | https://ocw.mit.edu/courses/6-092-introduction-to-programming-in-java-january-iap-2010/ |
| MIT OpenCourseWare,45040 | 01/05/2024 | Introduction to Computer Science and Programming | https://ocw.mit.edu/courses/6-00-introduction-to-computer-science-and-programming-fall-2008/ |