## 1.- DATOS DE LA ASIGNATURA

| | |
|---|---|
| Nombre de la asignatura: | **Fundamentos de Programación** |
| Carrera: | **Ingeniería en Sistemas Computacionales** |
| Clave de la asignatura: | **SCD-1008** |
| SATCA[1] | **2-3-5** |

## 2.- PRESENTACIÓN

**Caracterización de la asignatura.**

Esta asignatura aporta, al perfil del ingeniero, la capacidad para desarrollar un pensamiento lógico, identificar el proceso de creación de un programa y desarrollo de algoritmos para resolver problemas.

La asignatura proporciona al estudiante de ingeniería una herramienta para resolver problemas de aplicaciones de la vida ordinaria y de aplicaciones de la ingeniería.

Está diseñada para el logro de competencias específicas dirigidas al aprendizaje de los dominios: manejo de consola y diseño de algoritmos. Comprenderá los conceptos básicos de la programación y escribirá expresiones aritméticas y lógicas en un lenguaje de programación. Así como el uso y funcionamiento de las estructuras secuenciales, selectivas, arreglos unidimensionales y multidimensionales en el desarrollo de aplicaciones. Será capaz de aplicarlos al construir y desarrollar aplicaciones de software que requieran dichas estructuras.

Este curso genera las competencias necesarias para que el alumno desarrolle aplicaciones que den solución a los problemas que le plantee la vida diaria.

Fundamentos de programación es el soporte directo de las asignaturas: programación orientada a objetos, estructura de datos, tópicos avanzados de programación y de forma indirecta se relaciona con el desarrollo de sistemas de software, sistemas operativos y programación de sistemas.

**Intención didáctica.**

La asignatura proporciona al alumno los conceptos esenciales del diseño algorítmico, el temario se organiza en cinco unidades.

En la primera unidad se estudian los conceptos básicos para introducir al estudiante en la programación con la finalidad de obtener las bases conceptuales para abordar las siguientes unidades temáticas.

El análisis y desarrollo de algoritmos, como segunda unidad, es con la finalidad de dar

[1] Sistema de asignación y transferencia de créditos académicos

solución a problemas reales utilizando el razonamiento lógico.

La tercera unidad, tiene la finalidad de obtener y aplicar herramientas necesarias para diseñar e implementar soluciones en un lenguaje de programación, utilizando los conceptos adquiridos.

La cuarta unidad tiene como objetivo que el alumno identifique, comprenda, seleccione e implemente la estructura de control más adecuada a un problema específico, así como el diseño de bloques de códigos reutilizables, dado que es común encontrar en la práctica problemas cuyas operaciones están condicionadas o deban ejecutarse un número repetido de veces.

La quinta unidad tiene la finalidad de implementar arreglos para una gran variedad de propósitos que proporcionan un medio conveniente de agrupar variables relacionadas y organizar datos de una manera que puedan ser fácilmente procesados.

## 3.- COMPETENCIAS A DESARROLLAR

| **Competencias específicas:** | **Competencias genéricas:** |
|---|---|
| Analizar, diseñar y desarrollar soluciones de problemas reales utilizando algoritmos computacionales para implementarlos en un lenguaje de programación. | **Competencias instrumentales**<br>• Capacidad de análisis y síntesis.<br>• Capacidad de pensamiento lógico, algorítmico, heurístico, analítico y sintético.<br>• Resolución de problemas.<br>• Toma de decisiones.<br>• Destrezas tecnológicas relacionadas con el uso de maquinaria, destrezas de computación.<br>• Búsqueda y manejo de información.<br>**Competencias interpersonales**<br>• Capacidad crítica y autocrítica<br>• Trabajo en equipo<br>• Habilidades interpersonales<br>**Competencias sistémicas**<br>• Capacidad de aplicar los conocimientos en la práctica<br>• Habilidades de investigación<br>• Capacidad de aprender<br>• Capacidad de generar nuevas ideas (creatividad).<br>• Habilidad para trabajar en forma autónoma.<br>Búsqueda del logro |

## 4.- HISTORIA DEL PROGRAMA

| **Lugar y fecha de elaboración o revisión** | **Participantes** | **Observaciones (cambios y justificación)** |
|---|---|---|
| Instituto Tecnológico de Saltillo del 5 al 9 de Octubre de 2009. | Representantes de los Institutos Tecnológicos de: | Reunión nacional de Diseño e innovación curricular de la carrera de Ingeniería en |
| Institutos Tecnológicos Superiores de: Lerdo, Coatzacoalcos, Tepexi de Rodríguez.<br><br>Instituto Tecnológico de Mérida, Istmo y Villahermosa.<br><br>Fecha: 12 de Octubre de 2009 al 19 de Febrero del 2010. | Representantes de la Academia Sistemas Computacionales. | Análisis, enriquecimiento y elaboración del programa de estudio propuesto en la Reunión Nacional de Diseño Curricular de la carrera de Ingeniería en Sistemas Computacionales. |
| Instituto Tecnológico de fecha | Representantes de los Institutos Tecnológicos participantes en el diseño de la carrera de Ingeniería | Reunión nacional de consolidación de la carrea de ingeniería en |

## 5.- OBJETIVO(S) GENERAL(ES) DEL CURSO

Analizar, diseñar y desarrollar soluciones de problemas reales utilizando algoritmos computacionales para implementarlos en un lenguaje de programación.

## 6.- COMPETENCIAS PREVIAS

- Ninguna

## 7.- TEMARIO

| Unidad | Temas | Subtemas |
|---|---|---|
| 1 | Conceptos Básicos | 1.1 Clasificación del software de:<br>sistemas y aplicación.<br>1.2 Algoritmo.<br>1.3 Lenguaje de Programación.<br>1.4 Programa.<br>1.5 Programación.<br>1.6 Paradigmas de programación.<br>1.7 Editores de texto.<br>1.8 Compiladores e intérpretes.<br>1.9 Ejecutables.<br>1.10 Consola de línea de comandos. |
| 2 | Algoritmos | 2.1 Análisis de problemas.<br>2.2 Representación de algoritmos: gráfica y pseudocódigo.<br>2.3 Diseño de algoritmos aplicados a problemas.<br>2.4 Diseño algorítmico de funciones |
| 3 | Introducción a la Programación | 3.1 Características del lenguaje de programación<br>3.2 Estructura básica de un programa.<br>3.3 Traducción de un programa:<br>compilación, enlace de un programa, errores en tiempo de compilación.<br>3.4 Ejecución de un programa.<br>3.5 Elementos del lenguaje:<br>datos, literales y constantes, identificadores, variables, parámetros, operadores, entrada y salida de datos.<br>3.6 Errores en tiempo de ejecución. |
| 4 | Control de flujo. | 4.1 Estructuras secuenciales.<br>4.2 Estructuras selectivas:<br>simple, doble y múltiple.<br>4.3 Estructuras iterativas:<br>repetir mientras, hasta, desde<br>4.4 Diseño e implementación de funciones |
| 5 | Arreglos | 5.1 Unidimensionales:<br>conceptos básicos, operaciones y aplicaciones.<br>5.2 Multidimensionales:<br>conceptos básicos, operaciones y aplicaciones. |

## 8.- SUGERENCIAS DIDÁCTICAS

El profesor debe:

Ser conocedor de la disciplina que está bajo su responsabilidad. Desarrollar la capacidad para coordinar y trabajar en equipo; orientar el trabajo del estudiante, potenciar en él la autonomía, el trabajo cooperativo y la toma de decisiones. Mostrar flexibilidad en el seguimiento del proceso formativo y propiciar la interacción entre estudiantes.

- Proponer problemas que:
  - Propicien el desarrollo de la lógica de programación.
  - Permitan al estudiante la integración de los contenidos, para su análisis y solución.
  - Fortalezcan la comprensión de conceptos que serán utilizados en materias posteriores.
- Proponer actividades de búsqueda, selección y análisis de información en distintas fuentes.
- Propiciar el uso de las nuevas tecnologías en el desarrollo de los contenidos de la asignatura.
- Fomentar actividades grupales que propicien la comunicación, el intercambio argumentado de ideas, la reflexión, la integración y la colaboración de y entre los estudiantes.
- Propiciar, en el estudiante, el desarrollo de actividades intelectuales de inducción-deducción y análisis-síntesis, las cuales lo encaminan hacia la investigación, la aplicación de conocimientos y la solución de problemas.
- Desarrollar la capacidad de abstracción, análisis y síntesis.
- Fomentar el uso de las convenciones en la codificación de un algoritmo.
- Relacionar los contenidos de la asignatura con el respeto al marco legal, el cuidado del medio ambiente y con las prácticas de una ingeniería con enfoque sustentable.

## 9.- SUGERENCIAS DE EVALUACIÓN

La evaluación de la asignatura debe de ser continua y se debe considerar el desempeño en cada una de las actividades de aprendizaje, haciendo especial énfasis en obtener evidencias de aprendizaje como:

- Información obtenida durante las investigaciones solicitadas, plasmadas en documentos escritos o digitales
- Solución algorítmica a problemas reales o de ingeniería utilizando el diseño escrito o en herramientas digitales
- Codificación en un lenguaje de programación de las soluciones diseñadas
- Participación y desempeño en el aula y laboratorio
- Dar seguimiento al desempeño en el desarrollo del temario (dominio de los conceptos, capacidad de la aplicación de los conocimientos en problemas reales y de ingeniería)
- Se recomienda utilizar varias técnicas de evaluación con un criterio específico para cada una de ellas (teórico-práctico).
- Desarrollo de un proyecto por unidad que integre los tópicos vistos en la misma
- Desarrollo de un proyecto final que integre todas las unidades de aprendizaje
- Uso de una plataforma educativa en internet la cual puede utilizarse como apoyo para crear el portafolio de evidencias del alumno (integrando: tareas, prácticas, evaluaciones, etc.)

## 10.- UNIDADES DE APRENDIZAJE

**Unidad 1: Conceptos Básicos**

| Competencia específica a desarrollar | Actividades de Aprendizaje |
|---|---|
| Dominar los conceptos básicos de la programación. | • Investigar la clasificación del software.<br>• Reconocer los conceptos básicos: algoritmo, programa, programación, paradigmas de programación utilizando mapas conceptuales, mentales, cuadros sinópticos, etc.<br>• Conocer el entorno de un lenguaje de programación<br>• Manejar la consola para compilar y ejecutar programas. |

**Unidad 2: Algoritmos**

| Competencia específica a desarrollar | Actividades de Aprendizaje |
|---|---|
| Analizar problemas y representar su solución mediante algoritmos. | • Explicar los conceptos básicos para la formulación de algoritmos, así como sus ventajas y desventajas.<br>• Generar un catalogo de problemas para su análisis y solución.<br>• Resolver y analizar problemas cotidianos.<br>• Investigar los diferentes métodos para representar un algoritmo: diagrama de flujo, N-S (Nassi-Shneiderman), Pseudocódigo, Descripción Narrada. |

**Unidad 3: Introducción a la programación**

| Competencia específica a desarrollar | Actividades de Aprendizaje |
|---|---|
| Conocer las características principales de un lenguaje de programación.<br>Codificar algoritmos en un lenguaje de programación.<br>Compilar y ejecutar programas. | • Realizar un mapa conceptual sobre los tipos de software y los conceptos básicos de programación.<br>• Buscar y analizar información necesaria para Instalar y configurar el compilador del lenguaje de programación a utilizar.<br>• Realizar cambios en expresiones lógicas y algebraicas de un programa modelo y analizar los resultados obtenidos.<br>• Mostrar al estudiante programas completos de menor a mayor grado de dificultad y con base en cada una de las instrucciones que los componen enseñar la sintaxis del lenguaje. |

**Unidad 4: Control de flujo**

| Competencia específica a desarrollar | Actividades de Aprendizaje |
|---|---|

| | |
|---|---|
| Construir programas utilizando estructuras condicionales y repetitivas para aumentar su funcionalidad. | • Realizar una investigación sobre el funcionamiento y aplicación de las estructuras de selección y de repetición.<br>• Diseñar programas donde se utilicen las estructuras de repetición y selección.<br>• Construir programas que implementen métodos o funciones. |

**Unidad 5: Arreglos**

| **Competencia específica a desarrollar** | **Actividades de Aprendizaje** |
|---|---|
| Construir programas que utilicen arreglos unidimensionales y multidimensionales para solucionar problemas. | • Diseñar algoritmos que utilicen arreglos unidimensionales y multidimensionales.<br>• Desarrollar programas para implementar las operaciones básicas en arreglos. |

## 11.- FUENTES DE INFORMACIÓN


1. Luis Joyanes Aguilar, Fundamentos de Programación, Ed. Prentice Hall.
2. Jesús J. García Molina Introducción a la programación un Enfoque Algorítmico, Ed. Paraninfo.
3. Leobardo López Román, Metodología de la Programación Orientada a Objetos, Ed. Alfaomega.
4. Cairo Osvaldo, Metodología de la Programación, Ed. Alfaomega.
5. Deitel y Deitel. Como Programar en C++ quinta Edición. Prentice Hall.
6. Deitel y Deitel. Como Programar en C# quinta Edición. Prentice Hall.
7. Deitel y Deitel. Java como programar. Séptima edición. Prentice Hall.
8. Joyanes Aguilar, Luis Fernández, Azuela Matilde, Rodríguez Baena Luis, Fundamentos de Programación Libro de Problemas Algoritmos Estructura de Datos y Objetos. 2a. edición Ed. Mc. Graw Hill
9. Luis Joyanes Aguilar. Programación en JAVA 2 1ª Edición. Mc Graw Hill.
10. Martín Flowler Kendall Scott. UML Gota a Gota. Addison Wesley.
11. Ramírez Felipe, Introducción a la Programación, Algoritmos y su Implementación En Vb.Net C# Java y C++, 2a. edición, Alfa Omega.
12. Jean-Paul Tremblay, Richar B. Bunt. Introducción a la Ciencia de Las Computadoras. Enfoque Algorítmico. McGraw Hill.
13. Bjarne Storstrup. Lenguaje de Programación C/C++.
14. Cairo Battistutti Osvaldo, Metodología de la Programación, Algoritmos Diagramas de Flujo y Programas, 3a. edición, Alfa Omega.
15. Flores Cueto, Juan José, Método de las 6'D UML – Pseudocódigo – Java Enfoque Algorítmico, Serie Textos Universitarios Facultad de Ingeniería y Arquitectura, ed. Universidad de San Martín de Porres, (http://books.google.com/).


## 12.- PRÁCTICAS PROPUESTAS

- Investigar los tipos de software que se utilizan en una organización e identificarlos dentro de la clasificación del software.
- Elaborar ejercicios que impliquen entrada y salida de datos, así como el uso de operadores, operandos.
- Solucionar problemas con algoritmos a partir de enunciados proporcionados por el profesor.
- Crear, compilar y ejecutar programas.
- Declarar variables y usar expresiones aritméticas, relacionales, lógicas y de igualdad.
- Resolver problemas que utilicen entrada y salida de datos.
- Solucionar problemas utilizando sentencias de control.
- Implementar soluciones con arreglos.