UNIVERSIDAD AUTÓNOMA DE SINALOA
FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS
CARRERA: INGENIERÍA EN ELECTRÓNICA

PROGRAMA DE ESTUDIOS

1. DATOS DE IDENTIFICACIÓN			
UNIDAD DE APRENDIZAJE	PROGRAMACIÓN ORIENTADA A OBJETOS		
Clave:	**1354**		
Semestre:	**III**		
Eje Curricular:	**( ) Tronco Común (X) Profesionalizante**		
Área:	**( ) Física-Matemática ( ) Cs. Sociales y Humanidades ( )Idiomas** **(X) Básico Profesional ( ) Profesional**		
Horas y créditos:	**Teóricas: 40**	**Prácticas: 40**	**Estudio Independiente: 16**
	Total de horas: 96	**Créditos: 6**	
Tipo de curso:	**Teórico**	**Teórico-práctico (X)**	**Práctico**
Competencias del perfil de egreso a la que aporta	Diseña sistemas electrónicos analógicos y digitales para resolver problemas del entorno haciendo uso de diversas tecnologías atendiendo las normas y reglamentos para su uso. Desarrolla software y firmware para dispositivos electrónicos atendiendo las normas de calidad y reglamentación establecidas.		
Componentes	Desarrollo de firmware para microcontroladores diversos. Desarrollo de firmware para plataformas de arquitectura reconfigurable. Desarrollo de software para sistemas embebidos. Manejo de sistemas operativos diversos para sistemas embebidos.		
Unidades de aprendizaje relacionadas	Lenguaje de Programación, Sistemas Embebidos, Microcontroladores, Microprocesadores Avanzados		
Responsables de elaborar y/o actualizar el programa:	**Dr. Carlos Duarte Galván** **Dr. Jesús Roberto Millán Almaraz**		
Fecha de:	**Elaboración: agosto 2017**	**Actualización:**	

2. PROPÓSITO
El estudiante aprenderá tópicos avanzados de programación orientada a objetos y su implementación por medio de un lenguaje de programación, que sirvan como base para cursos posteriores donde se trabajará con Ambientes Integrados de Desarrollo.

3. SABERES	
Teóricos:	- Conocer la teoría de programación de software, su relación con el mundo actual y la aplicación en resolución de problemas de ingeniería. - Conocer el lenguaje de programación orientado a objetos

	- Comprender la diferencia entre programación estructurada y programación orientada a objetos. - Conocer las características de la programación orientada a objetos, uso de palabras reservadas, procedimientos y librerías disponibles en la WEB.
Prácticos:	-Desarrollar un proceso de abstracción que permita simplificar un problema, discriminar partes y trabajar de manera puntual en la resolución del problema. -Describir un sistema como un conjunto de bloques interconectados con una lógica que permita entender con mayor claridad cómo funcionan las etapas del sistema. - Interconectar el software desarrollado con una etapa electrónica de hardware a través de algún periférico de la computadora. -Trabajar en diferentes sistemas operativos.
Actitudinales:	- Valorar el papel de la Ciencia en el entendimiento de la naturaleza. - Demostrar rigor científico en el planteamiento y solución de problemas. - Actitud de participación en la solución de ejercicios. - Cultivar el autoaprendizaje. - Desarrollar la lectura de textos científicos. - Actitud reflexiva en la asimilación de nuevos conceptos. - Valorar la potencialidad de la mecánica estadística como puente para la ciencia interdisciplinaria.

4. CONTENIDO TEMÁTICO

UNIDAD I. FUNDAMENTOS DEL LENGUAJE

1.1. Entorno de desarrollo.
1.2. Configuración del entorno de desarrollo.
1.3. Palabras reservadas.
1.4. Comentarios.
1.5. Tipos de datos.
1.6. Variables.
1.7. Constantes.
1.8. Operadores.
1.9. Sentencias.
1.10. Conversión de tipos de datos (cast).
1.11. Estructuras de control.

UNIDAD II. ARREGLOS

2.1. Unidimensional.
2.2. Multidimensional.

UNIDAD III. CLASES Y OBJETOS

3.1. Definición de una clase.
3.2. Declaración de clases.
3.3. Miembros de una clase.

3.4. Ámbito referente a una clase.
3.5. Especificadores de acceso.
3.6. Creación de objetos.
3.7. Puntero this.
3.8. Constructores y destructores.
3.9. Clases Predefinidas.
3.10. Definición, creación y reutilización de paquetes/librerías.
3.11. Manejo de excepciones.

UNIDAD IV. MÉTODOS

4.1. Definición de un método.
4.2. Estructura de un método.
4.3. Valor de retorno.
4.4. Declaración de un método.
4.5. Ámbito y tiempo de vida de variables.
4.6. Argumentos y paso de parámetros.
4.7. Sobrecarga de métodos.
4.8. Encapsulamiento.

5. ACCIONES ESTRATÉGICAS PARA EL APRENDIZAJE

Sensibilización y atención: realizar una exposición introductoria de los temas en cada unidad, haciendo mención del contexto histórico en que los conceptos fueron desarrollados, así como de los problemas teóricos o tecnológicos que ayudaron a resolver los temas que se verán en dicha unidad temática.

Recomendar lectura previa de temas selectos, para crear discusiones y debates en torno al tema

En la plataforma virtual: transferencia de información al alumno de algunos temas concretos, entregar al profesor de tareas como programas de práctica, resúmenes y reportes de investigación, apertura de foros de discusión y seguimiento a ellos.

Estrategias y técnicas de aprendizaje: aprendizaje basado en problemas, aprendizaje colaborativo en la resolución de problemas, programación de algoritmos y en exposiciones. Método de proyectos.

6. EVALUACIÓN DEL APRENDIZAJE

6.1. Evidencias de aprendizaje	6.2. Criterios de desempeño	6.3. Calificación y acreditación
Exámenes Tareas Solución de problemas en clase Exposiciones en clase	Exámenes por unidad: Explicación clara y concreta de los conceptos relacionados con la materia. Solución correcta de problemas de ingeniería propuestos. En lo que respecta a los demás criterios de evaluación, se asignará 30% al formato, 40% al contenido y 30% a las conclusiones que el alumno presente.	70% exámenes. 30% Tareas, prácticas y demás trabajos.

7. *FUENTES DE INFORMACIÓN*

Fuentes de Información Básica:

1. Taylor David. Object Orient informations systems, planning and implementations. Canada: Wiley. 1992.
2. Larman Craig. UML y patrones introducción al análisis y diseño orientado a objetos. México: Pretince Hall. 1999.
3. Winblad, Ann L. Edwards, Samuel R. Software orientado a objetos. USA: Addison. Wesley/ Díaz Santos. 1993.
4. Deitel & Deitel. Java how to program. Prentice Hall.
5. Fco. Javier Ceballos. Java 2 Curso de Programación. Alfaomega.
6. Agustín Froufe. Java 2 Manual de usuario y tutorial. Alfaomega.
7. Laura Lemay, Rogers Cadenhead. Aprendiendo JAVA 2 en 21 días. Prentice Hall.
8. Herbert Schildt. Fundamentos de Programación en Java 2. McGrawHil.
9. Deitel y Deitel. Como programar en Java. Prentice Hall.

Fuentes de Información Complementaria:

1. Stephen R. Davis. Aprenda Java Ya. McGrawHill.
2. Kris Jamsa Ph D. ¡ Java Ahora!. McGrawHill.

8. PERFIL DEL PROFESOR:

- Posee un profundo conocimiento de programación en diferentes lenguajes, de manera que le permita conectar los saberes del curso con otras asignaturas, así como con el perfil de egreso del electrónico.

-Conocer y aplicar las diferentes potencialidades de la programación en la resolución de problemas de ingeniería.
- Conoce y aplicar adecuadamente la lógica de programación orienta a objetos a la resolución de problemas de ingeniería.
- Demuestra habilidades didácticas de enseñanza y evaluación del aprendizaje.

Programa Tec de culiacán

1. **Fundamentos del lenguaje**
   - 1.1. Entorno de desarrollo.
   - 1.2. Configuración del entorno de desarrollo.
   - 1.3. Palabras reservadas.
   - 1.4. Comentarios.
   - 1.5. Tipos de datos.
   - 1.6. Variables.
   - 1.7. Constantes.
   - 1.8. Operadores.
   - 1.9. Sentencias.
   - 1.10. Conversión de tipos de datos (cast).
   - 1.11. Estructuras de control.

2. **Arreglos**
   - 2.1. Unidimensional.
   - 2.2. Multidimensional.

3. **Clases y objetos**
   - 3.1. Definición de una clase.
   - 3.2. Declaración de clases.
   - 3.3. Miembros de una clase.
   - 3.4. Ámbito referente a una clase.
   - 3.5. Especificadores de acceso.
   - 3.6. Creación de objetos.

3.7. Puntero this.
3.8. Constructores y destructores.
3.9. Clases Predefinidas.
3.10. Definición, creación y reutilización de paquetes/librerías.
3.11. Manejo de excepciones.

4. Métodos

4.1. Definición de un método.
4.2. Estructura de un método.
4.3. Valor de retorno.
4.4. Declaración de un método.
4.5. Ámbito y tiempo de vida de variables.
4.6. Argumentos y paso de parámetros.
4.7. Sobrecarga de métodos.
4.8. Encapsulamiento.

5. Herencia y polimorfismo

5.1. Concepto de herencia y polimorfismo.
5.2. Definición de una clase base.
5.3. Definición de una clase derivada.
5.4. Clases abstractas.
5.5. Definición de herencia múltiple.
5.6. Implementación de herencia múltiple.
5.7. Reutilización de la definición de paquetes / librerías.
5.8. Clases genéricas (Plantillas).

6. Archivos

6.1. Definición de archivos de texto y archivos binarios
6.2. Operaciones básicas en archivos de texto y binarios
6.3. Manejo de excepciones en archivos.

Programa ESIME IPN

1. Fundamentos de la programación orientada a objetos

1.1. Conceptos fundamentales de la POO
1.2. Los lenguajes orientados a objetos
1.3. Relaciones entre clases y objetos
1.4. El papel de clases y objetos en el análisis y el diseño

2. Clases y funciones miembro

2.1. Construcción de clases y objetos
2.2. Estructuras, uniones y la palabra reservada class.
2.3. Funciones miembro (métodos, acciones u operaciones)
2.4. Resolución de ámbito

2.5. Constructor
2.6. Destructor

3. Herencia y polimorfismo

3.1. Polimorfismo
3.2. Polimosfismo paramétrico
3.3. Sobrecarga de operadores
3.4. Herencia
3.5. Herencia simple
3.6. Herencia múltiple
3.7. Herencia de elementos públicos, privados y protegidos.

4. Plantillas

4.1. Palabra reservada Template
4.2. Declaración y definición de una función template
4.3. Manejo de plantillas en el ambiente de programación (Turbo C++, Borland C++, etc.)

5. Aplicaciones

5.1. Análisis y diseño Orientado a Objetos en la resolución de un problema.
5.2. Codificación.
5.3. Pruebas modulares e integrales.
5.4. Mantenimiento.