TÉCNICO SUPERIOR UNIVERSITARIO EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN

HOJA DE ASIGNATURA CON DESGLOSE DE UNIDADES TEMÁTICAS

1. Nombre de la asignatura	**Programación.**
2. Competencias	Implementar aplicaciones de software mediante técnicas de programación y considerando los requerimientos de la organización para eficientar sus procesos.
3. Cuatrimestre	Segundo
4. Horas Prácticas	64
5. Horas Teóricas	26
6. Horas Totales	90
7. Horas Totales por Semana Cuatrimestre	6
8. Objetivo de la Asignatura	El alumno generará una aplicación informática utilizando el paradigma orientado a objetos en un lenguaje de programación para la solución de un problema específico; aplicando buenas prácticas de programación.

Unidades Temáticas	Horas		
	Prácticas	**Teóricas**	**Totales**
I. Fundamentos de programación orientada a objetos	6	4	10
II. Ambiente de desarrollo de la POO	4	2	6
III. Programación orientada a objetos	36	8	44
IV. Arreglos	12	6	18
V. Manejo de excepciones	10	2	12
Totales	**68**	**22**	**90**

PROGRAMACIÓN

UNIDADES TEMÁTICAS

1. Unidad Temática	**I. Fundamentos de programación orientada a objetos.**
2. Horas Prácticas	6
3. Horas Teóricas	4
4. Horas Totales	10
5. Objetivo	El alumno identificará los fundamentos de la programación orientada a objetos para abstraer los elementos de un escenario.

Temas	Saber	Saber hacer	Ser
Fundamentos de la POO	Identificar el paradigma de la POO. Describir los conceptos de objetos y clases (métodos y atributos).	Representar los métodos y atributos, de un conjunto de objetos a partir de un modelo real.	Analítico Ordenado Sistemático Hábil para comunicarse de forma oral y escrita Hábil para trabajo en equipo
Características de la POO	Identificar las características de la POO: abstracción, herencia, encapsulamiento y polimorfismo.	Representar las clases, la herencia, el encapsulamiento y el polimorfismo en un escenario.	Analítico Ordenado Sistemático Hábil para comunicarse de forma oral y escrita Hábil para trabajo en equipo

PROGRAMACIÓN

Proceso de evaluación		
Resultado de aprendizaje	**Secuencia de aprendizaje**	**Instrumentos y tipos de reactivos**
Elaborará, a partir de problema planteado, un modelo conceptual donde identifique las clases (atributos y métodos), encapsulamiento, herencia y polimorfismo.	1. Analizar el concepto de clase y objeto, abstracción, encapsulamiento, herencia y polimorfismo. 2. Identificar en un problema las clases, métodos, atributos y objetos involucrados. 3. Representar las clases (atributos y métodos) involucradas en la solución de un problema.	Ejercicios prácticos Lista de cotejo

PROGRAMACIÓN

Proceso enseñanza aprendizaje	
Métodos y técnicas de enseñanza	**Medios y materiales didácticos**
Equipos colaborativos Análisis de casos Mapas mentales y conceptuales	Cuadernillo de prácticas Cañón Pintarrón Computadora

Espacio Formativo		
Aula	**Laboratorio / Taller**	**Empresa**
X		

PROGRAMACIÓN

UNIDADES TEMÁTICAS

1. Unidad Temática	**II. Ambiente de desarrollo de la POO.**
2. Horas Prácticas	4
3. Horas Teóricas	2
4. Horas Totales	6
5. Objetivo	El alumno operará las funciones principales de un entorno de desarrollo previamente seleccionado, para ejecutar una aplicación.

Temas	Saber	Saber hacer	Ser
Generalidades de la herramienta de desarrollo	Identificar los diferentes tipos de entornos de desarrollo para los lenguajes de programación existentes.	Seleccionar el entorno de desarrollo acorde al lenguaje de programación.	Analítico Ordenado Sistemático Proactivo
Características de la interfaz de desarrollo	Identificar las funciones principales (crear, abrir, guardar, compilar, ejecutar, depurar) de operación en los entornos de desarrollo de programación.	Operar las funciones principales del entorno de desarrollo de programación.	Analítico Ordenado Sistemático Proactivo

PROGRAMACIÓN

Proceso de evaluación		
Resultado de aprendizaje	**Secuencia de aprendizaje**	**Instrumentos y tipos de reactivos**
Generará un proyecto, a partir de un código dado, y demostrará el manejo de las funciones principales del entorno de desarrollo: crear, abrir, guardar, compilar, ejecutar, depurar.	1. Analizar el concepto de entorno de desarrollo. 2. Identificar los tipos de entornos de desarrollo. 3. Operar las funciones principales del entorno de desarrollo.	Ejercicios prácticos Lista de cotejo

PROGRAMACIÓN

Proceso enseñanza aprendizaje	
Métodos y técnicas de enseñanza	**Medios y materiales didácticos**
Práctica demostrativa Trabajos de Investigación Equipos colaborativos	Cuadernillo de prácticas Cañón Pintarrón Computadora Entorno de desarrollo de programación

Espacio Formativo		
Aula	**Laboratorio / Taller**	**Empresa**
	X	

PROGRAMACIÓN

UNIDADES TEMÁTICAS

1. Unidad Temática	**III. Programación orientada a objetos.**
2. Horas Prácticas	36
3. Horas Teóricas	8
4. Horas Totales	44
5. Objetivo	El alumno codificará clases empleando el paradigma de la programación orientada a objetos para el desarrollo de aplicaciones.

Temas	Saber	Saber hacer	Ser
Estructura básica de una clase	Identificar la estructura básica de una clase, las instrucciones y su sintaxis. Identificar el punto de inicio de una clase (método main).	Codificar programas básicos con salida de datos, utilizando buenas prácticas de programación y corrigiendo errores de compilación.	Analítico Ordenado Sistemático Coherente Proactivo Asertivo
Aspectos léxicos	Identificar los tipos de datos y palabras reservadas. Identificar la sintaxis de la declaración de atributos (variables y constantes), operadores, comentarios, literales, expresiones.	Establecer atributos (variables y constantes) usando diversos tipos de datos, expresiones y operaciones y palabras reservadas.	Analítico Ordenado Sistemático Coherente Proactivo Asertivo

Temas	Saber	Saber hacer	Ser
Clases y Objetos	Identificar la sintaxis de la creación de clases y métodos (constructores, estáticos y no estáticos). Identificar la sintaxis de la declaración de objetos y la comunicación entre objetos y relación con modos de acceso.	Codificar clases que implementen métodos y la comunicación entre objetos.	Analítico Ordenado Sistemático Coherente Proactivo Asertivo
Estructuras de control	Identificar la sintaxis y el funcionamiento de las diferentes estructuras de control.	Codificar métodos empleando las estructuras de control.	Analítico Ordenado Sistemático Coherente Proactivo Asertivo
Encapsulamiento	Explicar el encapsulamiento a través del funcionamiento y sintaxis de los modificadores de acceso (private, protected, public) y los métodos mutator y acceso.	Codificar clases empleando propiedades de los diferentes modificadores de acceso y métodos de acceso.	Analítico Ordenado Sistemático Coherente Proactivo Asertivo
Herencia	Identificar la sintaxis de la herencia, método virtual, método sobrescrito, clases abstractas, interfaces, clases terminales (finales/selladas).	Codificar clases que implementen la herencia.	Analítico Ordenado Sistemático Coherente Proactivo Asertivo
Polimorfismo	Identificar el funcionamiento del polimorfismo de componentes (sobrecarga y sobre escritura) y de referencias (casting).	Codificar clases que implementen el polimorfismo.	Analítico Ordenado Sistemático Coherente Proactivo Asertivo

PROGRAMACIÓN

Proceso de evaluación		
Resultado de aprendizaje	**Secuencia de aprendizaje**	**Instrumentos y tipos de reactivos**
Codificará una aplicación que incluya: clases (atributos y métodos), empleando estructuras de control, encapsulamiento, herencia y polimorfismo.	1. Comprender la estructura de un programa y las reglas sintácticas del mismo. 2.- Identificar las clases (atributos y métodos), tipos de datos, variables, constantes y objetos involucrados en el problema y las relaciones entre ellos. 3.- Identificar las estructuras de control requeridas para la solución. 4.-Codificar la aplicación empleando encapsulamiento, abstracción, herencia y polimorfismo.	Ejecución de tareas Lista de cotejo

PROGRAMACIÓN

Proceso enseñanza aprendizaje	
Métodos y técnicas de enseñanza	**Medios y materiales didácticos**
Práctica demostrativa Ejercicios prácticos Equipos colaborativos	Cuadernillo de prácticas Cañón Pintarrón Computadora Entorno de desarrollo de programación

Espacio Formativo		
Aula	**Laboratorio / Taller**	**Empresa**
	X	

PROGRAMACIÓN

UNIDADES TEMÁTICAS

1. Unidad Temática	**IV. Arreglos.**
2. Horas Prácticas	12
3. Horas Teóricas	6
4. Horas Totales	18
5. Objetivo	El alumno implementará arreglos, para manejar conjuntos de elementos.

Temas	Saber	Saber hacer	Ser
Definición de Arreglos	Explicar el funcionamiento de un arreglo.		Analítico Ordenado Sistemático Coherente Proactivo Asertivo
Manejo de arreglos	Identificar la declaración de los tipos de arreglos (de datos primitivos y de objetos). Explicar el procedimiento para la manipulación de arreglos.	Codificar aplicaciones mediante arreglos.	Analítico Ordenado Sistemático Coherente Proactivo Asertivo

PROGRAMACIÓN

Proceso de evaluación		
Resultado de aprendizaje	**Secuencia de aprendizaje**	**Instrumentos y tipos de reactivos**
Entregará en un medio electrónico: • Código fuente. • Ejecutable de una aplicación que implemente el uso de arreglos de elementos.	1. Identificar los elementos de arreglos, tipos y su funcionamiento. 2. Comprender el funcionamiento de arreglo y la sintaxis del lenguaje para la declaración, creación e inicialización del arreglo. 3. Utilizar las estructuras repetitivas para la lectura, proceso y escritura de los elementos del arreglo.	Ejecución de tareas Lista de cotejo

PROGRAMACIÓN

Proceso enseñanza aprendizaje	
Métodos y técnicas de enseñanza	**Medios y materiales didácticos**
Práctica demostrativa Ejercicios prácticos Equipos colaborativos	Cuadernillo de prácticas Cañón Pintarrón Computadora Entorno de desarrollo de programación

Espacio Formativo		
Aula	**Laboratorio / Taller**	**Empresa**
	X	

PROGRAMACIÓN

UNIDADES TEMÁTICAS

1. Unidad Temática	**V. Manejo de excepciones.**
2. Horas Prácticas	10
3. Horas Teóricas	2
4. Horas Totales	12
5. Objetivo	El alumno controlará errores en tiempo de ejecución a través de excepciones para generar programas robustos.

Temas	Saber	Saber hacer	Ser
Concepto y tipos de excepciones.	Identificar los diferentes tipos de excepciones y la jerarquía de clases correspondientes a estas.		Analítico Ordenado Sistemático Coherente Proactivo Asertivo
Manejo de excepciones.	Identificar la sintaxis de las cláusulas para el manejo de excepciones (try, catch, final, throw y throws).	Controlar las excepciones generadas a través de las clausulas apropiadas en el manejo de errores en tiempo de ejecución.	Analítico Ordenado Sistemático Coherente Proactivo Asertivo
Creación y lanzamiento de excepciones.	Reconocer la estructura para la generación de excepciones y su lanzamiento.	Operar (crear, lanzar y manejar) excepciones.	Analítico Ordenado Sistemático Coherente Proactivo Asertivo

PROGRAMACIÓN

Proceso de evaluación		
Resultado de aprendizaje	**Secuencia de aprendizaje**	**Instrumentos y tipos de reactivos**
Entregará en un medio electrónico: • Código fuente. • Ejecutable de una aplicación que integre el control de errores en tiempo de ejecución, a través de excepciones.	1. Identificar los posibles errores que pueden generarse en tiempo de ejecución. 2. Analizar el concepto de excepción y relacionarlo con el tipo de error. 3. Utilizar la sintaxis requerida para el manejo de excepciones.	Ejecución de tareas Lista de cotejo

PROGRAMACIÓN

Proceso enseñanza aprendizaje	
Métodos y técnicas de enseñanza	**Medios y materiales didácticos**
Práctica demostrativa Ejercicios prácticos Equipos colaborativos	Cuadernillo de prácticas Cañón Pintarrón Computadora Entorno de desarrollo de programación

Espacio Formativo		
Aula	**Laboratorio / Taller**	**Empresa**
	X	

PROGRAMACIÓN

CAPACIDADES DERIVADAS DE LAS COMPETENCIAS PROFESIONALES A LAS QUE CONTRIBUYE LA ASIGNATURA

Capacidad	Criterios de Desempeño
Desarrollar la aplicación utilizando algún lenguaje de programación; para solucionar un problema específico.	Realiza la traducción del diseño al lenguaje de programación. Interpreta los errores de compilación y los corrige. Genera un programa ejecutable y realiza la verificación de los resultados (errores lógicos) al menos con 2 corridas usando valores de entrada diferentes. Presenta el código fuente del programa de acuerdo a las buenas prácticas de programación (tabulaciones, comentarios, nombres de variables, entre otras).

PROGRAMACIÓN

FUENTES BIBLIOGRÁFICAS

Autor	Año	Título del Documento	Ciudad	País	Editorial
Arnow, David	(2004)	*Introduction to programming using Java : an object-oriented approach*	Boston, Massachusett	Estados Unidos	Pearson Education
Baltasar García, J.	(2008)	*Metodología de la Programación*	Madrid	España	Prentice Hall
Cairo Battistuti Osvaldo	(2004)	*Metodología De La Programación*	D.F.	México	Alfaomega Grupo Editor S.A. De C.V.
Cairo Battistuti Osvaldo	(2008)	*Metodología de La Programación: Algoritmos, Diagramas De Flujo Y Programas (3ª Ed.)*	Buenos Aires	Argentina	Computec
Joyanes Aguilar Luis	(1988)	*Metodología de la Programación. Diagramas de Flujo, Algoritmos Y Programación Estructurada*	Madrid	España	Mc Graw Hill Interamericana Editores, S.A. De C.V.
Joyanes Aguilar Luis	(1992)	*Problemas de Metodología de La Programación*	Madrid	España	Mc Graw Hill Interamericana Editores, S.A. De C.V.
Katcheroff, Pablo	(2006)	*El Gran libro de la Programación*	Buenos Aires	Argentina	MP Ediciones
López Leobardo	(2006)	*Metodología De La Programación Orientada A Objetos*	D.F.	México	Alfaomega Grupo Editor S.A. De C.V.

Autor	Año	Título del Documento	Ciudad	País	Editorial
Rodríguez Almeida Ángel M.	(1991)	*Metodología De La Programación a través de Pseudocódigo.*	Madrid	España	Mc Graw Hill Interamericana Editores, S.A. De C.V.