DATOS IDENTIFICATIVOS DE LA UNIDAD FORMATIVA

UNIDAD FORMATIVA	PRINCIPIOS DE LA PROGRAMACIÓN ORIENTADA A OBJETOS.	DURACIÓN Específica	90
Código	UF2404		
Familia profesional	INFORMÁTICA Y COMUNICACIONES		
Área Profesional	Desarrollo.		
Certificado de profesionalidad	PROGRAMACIÓN CON LENGUAJES ORIENTADOS A OBJETOS Y BASES DE DATOS RELACIONALES	Nivel	3
Módulo formativo	Programación orientada a objetos.	Duración	250
Resto de unidades formativas que completan el módulo	Modelo de programación web y bases de datos.	Duración	80
	El ciclo de vida del desarrollo de aplicaciones.		80

Apartado A: **REFERENTE DE COMPETENCIA**

Esta unidad formativa se corresponde con la RP1 en lo referido a la implementación de los componentes de software.

Apartado B: **ESPECIFICACIÓN DE LAS CAPACIDADES Y CONTENIDOS**

Capacidades y criterios de evaluación:

C1: Dominar los conceptos fundamentales del paradigma orientado a objetos.

CE1.1 Explicar las características del ciclo de desarrollo del software bajo el paradigma de orientación a objetos, distinguiendo la programación orientada a objetos como una fase dentro del mismo.

CE1.2 Describir y enumerar las características de una clase: atributos, métodos y mecanismo de encapsulación, identificando la interfaz de la clase y lo que representa.

CE1.3 Describir y enumerar las características que definen un objeto, distinguiendo las diferencias entre los conceptos de objeto y clase.

CE1.4 Describir la estructura y el significado de los mensajes y su relación con el comportamiento de los objetos.

CE1.5 Explicar las características fundamentales que tienen que estar presentes en una relación entre dos clases para que pueda ser calificada como relación de herencia.

CE1.6 Describir el mecanismo de herencia múltiple y los problemas que presenta en el proceso de desarrollo de software.

CE1.7 Explicar el concepto de polimorfismo y enumerar y describir las características que introduce en el proceso de desarrollo del software.

CE1.8 En un supuesto práctico, a partir de una documentación típica de diseño detallado, identificar las clases establecidas, los atributos y las relaciones.

C2: Desarrollar clases aplicando los fundamentos del paradigma Orientado a Objetos.

CE2.1 Enumerar y describir los principales criterios de calidad del software y los principales factores evaluados por las métricas orientadas a objetos.

CE2.2 Enumerar y describir los mecanismos de gestión de memoria utilizados en la creación y destrucción de los objetos.

CE2.3 Describir los mecanismos existentes para realizar la implementación de las relaciones entre clases.(Clases contenedores, objetos colección, etc).

CE2.4 Explicar la utilización de los objetos «super» y «this» («current», «self» u otros), en relación con el acceso a los atributos definidos en una clase, desde una subclase o desde el código de la propia clase.

CE2.5 Clasificar los diferentes lenguajes de programación, identificando y reconociendo en los mismos las principales características del paradigma orientado a objetos: Clases, objetos, herencia y polimorfismo.

CE2.6 Distinguir y utilizar las características proporcionadas por un entorno de desarrollo asociado a un lenguaje Orientado a Objetos.

CE2.7 Distinguir las estructuras de datos más habituales (listas, pilas, árboles, grafos, etc) y los posibles mecanismos de construcción en los lenguajes orientados a objetos.

CE2.8 Distinguir las librerías de clases estándares del lenguaje de programación conociendo la utilidad de cada una de ellas y la forma básica de uso.

CE2.9 En un supuesto práctico, construir las clases que representan las estructuras de datos en un lenguaje orientado a objetos.

CE2.10 En un supuesto práctico, en el que se pide realizar la programación de una clase con un lenguaje orientado a objetos y desde una documentación a nivel de diseño detallado:

- Diseñar un algoritmo para cada operación definida en la clase, aplicando técnicas de programación estructurada y modular.
- Elegir la estructura de datos más adecuada para cada atributo.
- Codificar cada atributo utilizando los tipos base proporcionados por el lenguaje, si es el caso, y las librerías de clases existentes.
- Codificar los métodos de acceso a los atributos siguiendo los criterios de calidad que se establezcan
- Codificar los métodos constructores utilizando la sobrecarga si es necesario siguiendo los criterios de calidad que se establezcan
- Codificar los métodos, como función o procedimiento, teniendo en cuenta la interfaz de la clase y los algoritmos diseñados y siguiendo los criterios de calidad que se establezcan.
- Incluir las relaciones de especialización / generalización, agregación / composición y / o de asociación con el resto de las clases descritas en el diseño en la construcción de la clase
- Incluir el código para el tratamiento de casos de error y excepciones de usuario
- Usar las librerías de clases existentes para incorporar accesos a bases de datos, interfaces gráficas y otras librerías

Contenidos:

1. Introducción al paradigma orientado a objetos

- Ciclo de desarrollo del software bajo el paradigma de orientación a objetos: Análisis, diseño y programación orientada a objetos.
- Análisis del proceso de construcción de software: Modularidad.
- Distinción del concepto de módulo en el paradigma orientado a objetos.
- Identificación de objetos como abstracciones de las entidades del mundo real que se quiere modelar.
  - Descripción de objetos: Conjunto de datos que definen un objeto y conjunto comportamientos que pueden solicitarse a los objetos.
  - Identificación del comportamiento de un objeto: Concepto de mensaje.

2. Clases y objetos

- Distinguir el concepto de clase y sus atributos, métodos y mecanismo de encapsulación:
  - Relación entre interfaz y clase.
  - Distinción de los tipos de datos y clases.
- Análisis de los objetos: Estado, comportamiento e identidad:
  - Análisis de mensajes.
  - Tipos de métodos y su clasificación: Métodos de acceso, de selección o consulta, de construcción, de destrucción.
- Uso de objetos como instancias de clase. Instancia actual (this, self, current).
- Identificación del concepto de programa en el paradigma orientado a objetos. POO = Objetos + Mensajes.

3. Generalización/Especialización: herencia

- Descripción del concepto de herencia: Simple y múltiple:
  - Relación de herencia: Características.
  - Reglas y características que definen una relación de herencia: Regla «Es-un».
  - Transmisión de atributos y métodos.
  - Regla de especialización de la superclase en la subclase.
  - Acceso a los atributos de una clase y acoplamiento entre las clases.
  - Utilización de objetos this (current, self u otros) y super.
  - Leyes de Demeter.
- Distinción de la herencia múltiple:
  - Problemas: Conflictos de nombres, herencia repetida.
  - Soluciones.
- Creación de objetos en la herencia.
- Clasificación jerárquica de las clases:
  - Clase raíz.
  - Clases abstractas.
  - Métodos virtuales.
  - Redefinición de métodos.

4. Relaciones entre clases

- Distinción entre Agregación/Composición.
- Distinción entre Generalización / Especialización.
- Identificación de asociaciones..

5. Análisis del polimorfismo

- Concepto.

– Tipos:
- Polimorfismo en tiempo de compilación (sobrecarga).
- Polimorfismo en tiempo de ejecución (ligadura dinámica).

– Objetos polimórficos.
- Comprobación estática y dinámica de tipos.

6. Técnicas de programación estructurada

– Identificación de elementos básicos: constantes, variables, operadores y expresiones.
– Análisis de estructuras de control: Secuencial, condicional y de repetición.
– Distinción entre funciones y procedimientos:
- Interfaz.
- Paso de parámetros: Por valor y por referencia.
- Parámetros actuales y formales.
- Funciones: valor de retorno.
- Procedimientos.
- Ámbito de las variables.
- Almacenamiento de las variables.

– Demostración de llamadas a funciones y procedimientos.
– Empleo de llamadas a funciones y procedimientos incluidos en las clases:
- Llamadas calificadas y no calificadas (instancia actual).
- Paso de parámetros.
- Los atributos de la clase.

7. Estructura de la información

– Enumeración de datos simples: Numéricos (enteros y reales), lógicos, carácter, cadena de caracteres, puntero o referencia a memoria.
– Datos estructurados: Arrays:
- Listas enlazadas, pilas y colas.
- Estructuras.
- Ficheros.
- Otras estructuras complejas: Tablas hash e Introducción a los árboles y grafos.

– Mecanismos de gestión de memoria:
- Uso de la gestión automática de memoria.
- Construcción y destrucción de objetos.
- Objetos inalcanzables.
– Recolección de «basura».
– Métodos constructores y destructores.

8. Lenguajes de programación orientados a objetos

– Análisis del lenguaje de programación orientado a objetos y paradigma orientado a objetos:
- Lenguajes de programación orientados a objetos.
- Lenguajes de programación basados en objetos.
- Lenguajes de programación que utilizan objetos.

– Comparación entre los lenguajes de programación orientados a objetos más habituales. Características esenciales.
– Librerías de clases:
– Definición de su estructura.
- Creación y utilización.

9. Implementación del paradigma utilizando un lenguaje de programación orientado a objetos

– Elección del lenguaje.
– Enumeración de los tipos de aplicaciones.
– Herramientas de desarrollo.
– Tipos de datos y elementos básicos característicos del lenguaje. Instrucciones.
– Estudio y utilización de las clases básicas incluidas en la librería de clases.
– Definición de clases:
– Construcción de métodos. Sobrecarga.
– Construcción de atributos.
– Construcción de la interfaz de la clase.
– Construcción de clases incluyendo relaciones de Agregación /Composición y Asociación.
– Construcción de clases con herencia.
– Construcción de clases con herencia múltiple.
– Definición de clases abstractas.
– Construcción de clases con herencia incluyendo polimorfismo.

- Empleo de excepciones.
- Gestión de eventos:
  - Eventos, fuentes y auditores de eventos.
  - Tipos de eventos. Mecanismos de gestión de eventos.
  - Librerías de clases asociadas.
- Empleo de hilos:
  - Fundamentos.
  - Creación.
  - Prioridad.
  - Comunicación.
  - Sincronización.
  - Estados.
  - Creación y ejecución de hilos en el lenguaje.
  - Librerías de clases asociadas.
  - Programación multihilo.
- Definición y análisis de programación en red:
  - Aplicaciones cliente servidor.
  - Sockets.
- Acceso a bases de datos desde las aplicaciones. Librerías de clases asociadas.

Apartado C: **REQUISITOS Y CONDICIONES**

Deberá cumplir alguno de los requisitos siguientes:

- Estar en posesión del título de Bachiller.
- Estar en posesión de algún certificado de profesionalidad de nivel 3.
- Estar en posesión de un certificado de profesionalidad de nivel 2 de la misma familia y área profesional.
- Cumplir el requisito académico de acceso a los ciclos formativos de grado superior o haber superado las correspondientes pruebas de acceso a ciclos de grado superior.
- Tener superada la prueba de acceso a la universidad para mayores de 25 años y/o de 45 años.
- Tener, de acuerdo con la normativa que se establezca, los conocimientos formativos o profesionales suficientes que permitan cursar con aprovechamiento la formación.

En relación con las exigencias de los formadores o de las formadoras, instalaciones y equipamientos se atenderá las exigencias solicitadas para el propio certificado de profesionalidad.