CAPÍTULO 1

COMPRUEBA TU APRENDIZAJE

1. ¿Cuál de las siguientes afirmaciones es correcta?
2. En los modelos evolutivos no se necesita conocer todos los requisitos al comienzo.
3. Es muy común en el modelo en cascada el uso de prototipos.
4. El análisis de riesgos se lleva a cabo en cada incremento del modelo iterativo incremental.
5. El modelo es cascada es apropiado cuando se necesita una versión inicial del software a desarrollar.
6. Relaciona:

|  |  |
| --- | --- |
| Modelo en cascada (1)  Modelo iterativo incremental (2)  Modelo en espiral (3) | Es fácil de comprender. (1)  Los clientes necesitan versiones intermedias. (2)  No se necesita conocer todos los requisitos al comienzo. (3, 2)  Reduce riesgos del proyecto. (3)  Los requisitos son estables. (1)  Genera mucho trabajo adicional. (3)  No se sabe cuando va a terminar. (2, 3)  Los requisitos son estables. (1)  El proyecto es similar a uno ya realizado (1)  Se acomoda bien a los cambios de requisitos. (2, 3) |

1. ¿Qué se hace en la etapa de análisis del desarrollo de una aplicación? ¿Qué tipos de requisitos se especifican en esta fase? Nombra algunas herramientas para obtener y representar los requisitos. ¿Dónde debe quedar reflejado todo lo realizado en esta fase?

Epígrafe 1.4.1

1. ¿Cuál de estas afirmaciones sobre la fase de diseño en el desarrollo de una aplicación es correcta?
2. En esta fase se especifica qué hay que hacer.
3. En esta fase se especifica cómo hacerlo.
4. En esta fase se realiza el proceso de programación.
5. En esta fase se realizan las pruebas.
6. ¿Cuáles son los 4 elementos fundamentales que intervienen en el diseño estructurado? Nombra algunas notaciones gráficas utilizadas para representar la fase de diseño.

Diseño de datos, arquitectónico, de la interfaz y diseño a nivel de componentes.

Diagramas de flujo, diagramas de cajas, tablas de decisión y pseudocódigo.

1. ¿Por qué crees que debe haber normas de codificación y estilo a la hora de llevar a cabo la fase de codificación en en el desarrollo de una aplicación?

Epígrafe 1.4.3

1. ¿En qué consisten las tareas de verificación y validación del software?

Epígrafe 1.4.4

1. ¿Qué técnicas se utilizan para llevar a cabo el diseño de casos de prueba? ¿En qué se consisten?

Epígrafe 1.4.4.1

1. La documentación de un proyecto software se divide en dos clases: la documentación del proceso y la del producto. ¿Qué contiene cada una?

Epígrafe 1.4.5

1. ¿Qué información debe contener la documentación del usuario y la del sistema?

Epígrafe 1.4.5

1. ¿Qué tareas se llevan a cabo durante la explotación del sistema?

Epígrafe 1.4.6

1. ¿Cuántos tipos de mantenimiento del software existen? ¿En qué consisten?

Epígrafe 1.4.7

1. ¿Qué tareas se llevan a cabo en la etapa de mantenimiento del software?

Epígrafe 1.4.7

1. ¿Qué relación tiene un programa con el hardware del ordenador donde se ejecuta?

Epígrafe 1.5.1

1. ¿De qué tres elementos consta un lenguaje de programación?

Epígrafe 1.6

1. ¿Qué diferencia hay entre un lenguaje de alto nivel y otro de bajo nivel?

Epígrafe 1.6.1

1. Diferencias entre un compilador y un intérprete.

Epígrafe 1.6.1

1. ¿Qué ventajas aporta la programación modular?

Epígrafe 1.6.1

1. ¿Cuales son los diferentes estados por los que pasa el código de un programa desde que se escribe hasta que se ejecuta por el ordenador?

Epígrafe 1.7.1

1. ¿Cuáles son las etapas o fases de un compilador?

Epígrafe 1.7.2

1. Cuáles de las siguientes afirmaciones sobre la máquinas virtual de Java es cierta:
2. Un fichero *.class* contiene código en un lenguaje máquina.
3. La máquina virtual de java toma y traduce el bytecode en código binario.
4. Los ficheros *.class* sólo pueden ser ejecutados en Microsoft Windows y Linux.
5. A la hora de instalar el entorno de ejecución de la máquina virtual java necesitamos saber en qué sistema operativo se va a instalar.

ACTIVIDADES DE AMPLIACIÓN

1. Modifica el pseudocódigo y el diagrama de flujo de los Ejemplos 3 y 4, de forma que se utilice la estructura *Mientras <condición> Hacer <instrucciones> Fin mientras*.
2. Modifica el pseudocódigo y el diagrama de flujo del Ejemplos 5, de forma que se utilice la estructura *Repetir <instrucciones> Hasta <condición>*.
3. Utilizando como modelo el Ejemplo 6, realiza un pseudocódigo y un diagrama de flujo que lea un número y muestre la nota a la que corresponde. Si el valor es 5 debe mostrar *Suficiente,* si es 6 *Bien,* si es 7 *Notable bajo,* si es 8 *Notable alto,* si es 9 *Sobresaliente,* si es 10 *Matrícula de honor*; para cualquier otro valor debe mostrar *Suspenso.*
4. Modifica el pseudocódigo del Ejemplo 6 de manera que se utilice la estructura *Si <condición> Entonces <Instrucciones> Si no <Instrucciones> Fin si*, en lugar de Según sea.
5. Utilizando como modelo el Ejemplo 2, realiza un pseudocódigo y un diagrama de flujo que lea un número y muestre el menor de ellos. Si los números son iguales debe mostrar un mensaje indicándolo.
6. Escribe el pseudocódigo para el siguiente diagrama de flujo:

|  |  |
| --- | --- |
| Dibujo7.png | **Inicio**  Declarar Cuenta=0  Abrir Fichero Notas  Leer Registro (Nombre, Curso, Nota)  **Mientras** NO sea Final de Fichero **Hacer**  **Si** Nota >= 8 **Entonces**  Cuenta = Cuenta +1  **Fin si**  Leer Registro (Nombre, Curso, Nota)  **Fin mientras**  Visualizar “Alumnos con un 8:” Cuenta  Cerrar Fichero Notas  **Fin** |

1. Escribe el diagrama de flujo para el siguiente pseudocódigo:

|  |  |
| --- | --- |
| **Inicio**  Declarar Cuenta=0, Suma =0,  Declarar Positivos = 0, Negativos = 0  **Repetir**  Visualizar “Escribe un número”  Leer A  Cuenta = Cuenta +1  **Si** A >= 0 **Entonces**  Positivos=Positivos +1  **Si no**  Negativos = Negativos +1  **Fin si**  Suma = Suma + A  **Hasta** Cuenta = 10  Visualizar Suma, Positivos, Negativos  **Fin** | **SolucEjer7.png** |

1. Se desea realizar un sencillo sistema de gestión de notas en el que van a interactuar profesores y alumnos. El sistema permitirá a un alumno consultar la nota de las asignaturas en que está matriculado y a un profesor introducir las notas de los cursos a los que imparte clase. Tanto el profesor como el alumno deben identificarse para entrar al sistema mediante un nombre de usuario y una clave, por lo tanto debe existir un almacén con los usuarios que pueden entrar en el sistema. Si la identificación no es correcta el sistema rechaza al usuario. Los procesos para alumno y profesor son los siguientes:

* Para alumno, 2 procesos: comprobar si el usuario está registrado y consultar la nota de la asignatura solicitada, estas notas estarán en un almacén. Cuando el usuario se identifica en el sistema puede introducir el nombre de la asignatura para que este le devuelva la nota. Para nombrar el flujo de datos entre el proceso de comprobación de usuario y de consulta de notas puedes usar: *pantalla de usuario*.
* Para profesor, 3 procesos: comprobar si el usuario está registrado, solicitar el curso al que introducirá las notas e introducir las notas del curso solicitado. Para nombrar el flujo de datos entre el proceso de comprobación y de solicitud de curso puesdes usar: *pantalla de profesor*, y para el de solicitud de curso e introducción de notas: *pantalla de curso*. Cuando el profesor se identifica envía al sistema el curso del que desea introducir las notas y las notas, estas se irán registrando en un almacén.

Se pide realizar los DFDs de nivel 0, 1 y 2. Define dos almacenes, uno para los usuarios que se identifican en el sistema y otro para las notas.

Una posible solución se muestra a continuación:







