FASES DE DESARROLLO

ANÁLISIS

1.- ¿Qué es fundamental para poder resolver un problema?

Analizarlo, para entender y comprender de forma detallada el problema que se va a resolver. Además es fundamental generar una documentación entendible, completa y fácil de verificar y modificar.

2.- ¿Qué se especifica en la fase del análisis?

Se analizan y especifican los requisitos o capacidades que el sistema debe tener porque el cliente así lo ha pedido.

3.- ¿Qué dificultades nos encontramos a la hora de realizar el análisis?

La obtención de información, ya que el cliente puede no expresar con claridad lo que quiere, sugerir cambios o dar información incorrecta debido a la falta de conocimientos informáticos.

4.- ¿Qué técnicas se pueden utilizar para facilitar la comunicación con el cliente?

Entrevistas, brainstorming, prototipos o otros métodos.

5.- ¿Qué técnica podemos utilizar para clarificar dudas iniciales del cliente y además puede utilizarse como primera versión del programa?

La técnica de prototipos.

6.- ¿De qué técnicas disponemos para representar los requisitos?

De diagramas de flujo de datos (DFD), diagramas de flujo de control (DFC), diagramas entidad relación (DER), diccionario de datos (DD) y otra documentación.

7.- ¿Toda la documentación generada en el análisis en qué documento se deberá recoger?

Sí, deberá verse reflejada en el documento de especificación de requisitos del software, según IEEE 830.

8.- Rellena la tabla.

1) Flujo de Datos
2) Entidad Relación
3) Flujo de Datos
4) Entidad Relación

DISEÑO

1.- ¿Qué establece el diseño?

La forma en la que se solucionará el problema.

2.- ¿Cuáles son los dos principales tipos de diseño?

- Diseño estructurado
- Diseño orientado a objetos

3.- ¿Qué tres construcciones están en los fundamentos del diseño estructurado?

Las estructuras secuenciales, las alternativas y las repetitivas.

4.- ¿Qué es un algoritmo?

Un conjunto de instrucciones que tienen como propósito solucionar una tarea concreta.

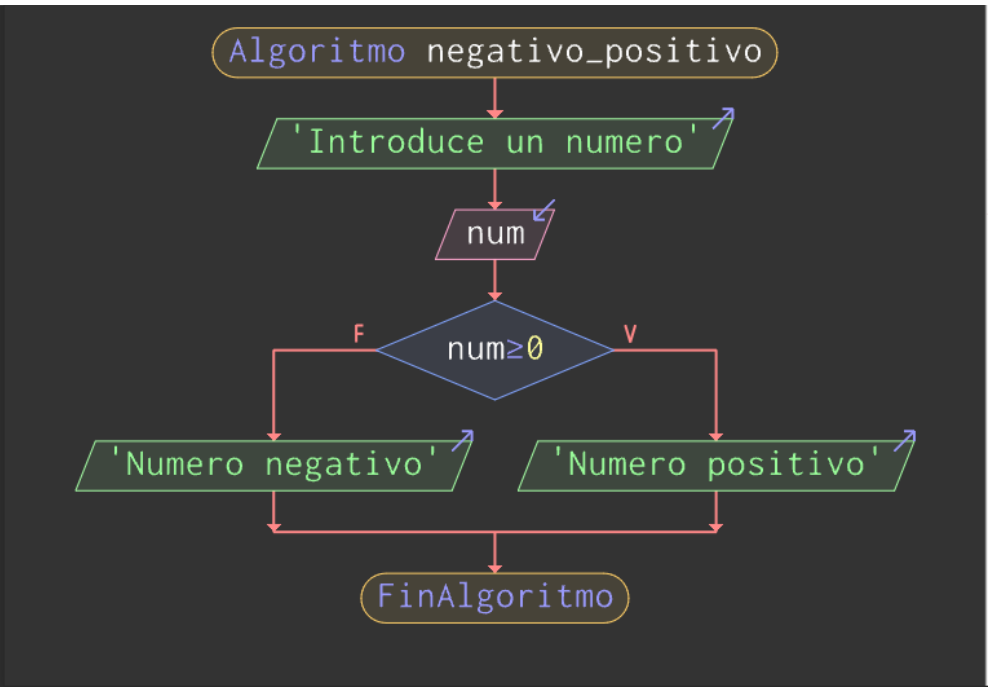
5.- ¿Qué dos técnicas se utilizan en el diseño de algoritmos?

Diagramas de flujo y pseudocódigo.

6.- Realiza el diseño del algoritmo mediante diagramas de flujo y pseudocódigo para el siguiente problema:

Dado un número decir si es positivo o negativo.

```
1  Algoritmo negativo_positivo
2      Escribir "Introduce un numero"
3      Leer num
4      Si num ≥ 0 Entonces
5          Escribir "Numero positivo"
6      SiNo
7          Escribir "Numero negativo"
8      Fin Si
9  FinAlgoritmo
10
```



CODIFICACIÓN Y PRUEBAS

1.- ¿Qué realiza el programador en la fase de codificación?

Recibe las especificaciones del diseño y las transforma en instrucciones, en algún lenguaje de programación.

2.- ¿Cómo se llama el conjunto de instrucciones generado en la fase de codificación?

Código fuente.

3.- Una vez obtenemos el código fuente, ¿lo podemos ejecutar?

Sí, si está libre de errores y compilado (en el caso de que sea compilado).

4.- ¿Por qué convenciones de código?

Para que el software sea más fácil de mantener, entendible por otras personas, más legible y presentable.

5.- Comenta 5 convenciones que te llamen la atención.

Los comentarios de documentación, convenciones de nombres, convenciones en paréntesis, la expresión '?' y los comentarios especiales.

6.- ¿Cuál es el objetivo de la fase de pruebas?

Encontrar errores de manera eficiente y optimizar el software.

7.- ¿Cuándo consideraremos que ha tenido éxito una prueba?

Cuando se encuentra un error nuevo.

8.- ¿Qué es un caso de prueba?

Un documento que especifica los valores de entrada y salida esperados, así como las condiciones para la ejecución de la prueba.

9.- Marca como verdadero falso las siguientes afirmaciones

Afirmación	V o F
Cada prueba debe definir unos resultados esperados	V
Una prueba tiene éxito si no hay errores ya que demuestra que el software funciona correctamente	F
El mejor probador es el propio programador ya que conoce donde pueden estar los errores en su código	F
Las pruebas deben incluir únicamente datos de entrada válidos y esperados	F
Se debe evitar hacer pruebas sin documentar	F
Uno de los dos objetivos en los que nos debemos centrar es comprobar que el software no hace lo que debe hacer	V
Uno de los dos objetivos en los que nos debemos centrar es comprobar que el software hace lo que no debe hacer	V
Es importante comprobar que el software no hace lo que no debe hacer	V

Un caso de prueba es un documento que especifica valores de entrada, salida esperada y condiciones de la ejecución de la prueba	V

MANTENIMIENTO

1.- Explica en qué consiste el mantenimiento.

Consiste en realizar cambios al software para corregir errores, modificar el funcionamiento o adaptarlo a distintas situaciones.

2.- ¿Cuándo termina el mantenimiento?

Cuando termina el ciclo de vida del software.

3.- ¿Qué cuatro tipos de mantenimiento existen?

Adaptativo, correctivo, perfectivo y preventivo.

4.- Rellena la tabla

Hechos	Tipo
La empresa migra de Windows a Linux sus equipos	Adaptativo
El cliente piensa en añadir un nuevo gráfico que le ayudará en la toma de decisiones	Preventivo
Tras entrega el software el cliente se da cuenta que nuestro software muestra la contraseña sin codificar al introducirla	Correctivo
Añadimos comentarios en la entrada de cada programa con la fecha y desarrollador	Perfectivo