PAC 1. UF1.

Desarrollo de software.

Aníbal Santos Gómez

1. **¿Encuentras alguna diferencia entre el freeware y el software libre? Explícalas.**

Para entender estas dos definiciones deberemos entender el concepto de licencia. Una licencia es contrato entre el desarrollador de un software y el usuario final.

El desarrollador es el que especifica qué tipo de licencia distribuye. Existen tres tipos de licencias, pero para este caso nos interesan dos:

- **Software libre (Open Source):** el autor de la licencia concede libertades al usuario, entre las que destacan, la libertad para usar el programa con cualquier fin, libertad para saber como funciona el programa y adaptar el código a las necesidades propias, libertad para compartir copias con otros usuarios y libertad para poder mejorar el programa y publicar las modificaciones realizadas.

- **Software propietario:** es el software que no nos permitirá acceder al código fuente del programa y nos prohibirá la redistribución y la reprogramación del mismo o su copia.

  En este caso existen dos variantes, el shareware (que permite compartir la copia con un tiempo de expiración de prueba, o dependiendo del caso se permite distribuir algunas piezas) y el software freeware, que es software libre de pago, pero la diferencia con el software libre es que este tipo de software no es libre, ya que no se garantiza la modificación y la redistribución de dichas versiones modificadas del programa.

2. **Explica con tus palabras el funcionamiento de un compilador.**

Un compilador es un programa informático, que traduce programa escrito en un lenguaje de programación a un lenguaje común.

Básicamente es un proceso a través del cual se convierte un programa en el lenguaje de una máquina, a través de otro programa escrito en otro lenguaje.

Este proceso se lleva a cabo a través de dos programas, el primero de ellos, el compilador, detecta los fallos y generará el código hasta que se solventen.

Una vez solventados los fallos se generará el código intermedio y se optimizará el código intermedio para después generar un código objeto que será pasado al segundo programa, el enlazador.

El enlazador tomará los objetos generados y guiará los recursos que no sean necesarios, enlazando el código objeto con sus bibliotecas con lo que finalmente se producirá un fichero ejecutable o una biblioteca.

3. Explica las ventajas y desventajas de la máquina virtual de java.

Las maquinas virtuales son un tipo de software capaz de ejecutar programas como si fueran una maquina real.

Los programas que se compilan en lenguaje Java son capaces de funcionar en cualquier plataforma, ya que el código no lo ejecuta el procesador del ordenador sino la Maquina Virtual de Java, esta es una de las principales ventajas, de Java, que es universal, solo necesitaremos la maquina virtual para nuestro sistema operativo, por lo que estamos ante una maquina virtual de proceso.

Otra ventaja es que java está diseñado pensando en la seguridad, puesto que la maquina virtual está equipada con características de seguridad que permiten a los programadores escribir programas en Java de alta seguridad. Impide que el software malicioso ponga en peligro el sistema operativo ya que mantiene las aplicaciones para interactuar con los recursos del sistema de forma intermedia.

En cuanto a las desventajas, hablaremos del rendimiento, ya que los programas java se ejecutan en una maquina virtual y tienden a funcionar más lento que los programas equivalentes escritos en otro lenguaje ya compilado como C++.

4. ¿Cuáles son los elementos de los lenguajes de programación?

Los lenguajes de programación constan de un alfabeto o vocabulario, también denominado léxico, una sintaxis y una semántica.

- El léxico o vocabulario está formado por el conjunto de símbolos permitidos.

- La sintaxis son las reglas que indican cómo realizar las construcciones con los símbolos del lenguaje.

- La semántica son las reglas que determinan el significado de cualquier construcción del lenguaje.

5. Realiza la clasificación de los siguientes lenguajes: C, Java, C#, JavaScript, PHP, Prolog.

- C: es un lenguaje de programación con núcleo del lenguaje simple, con funcionalidades matemáticas, y bibliotecas. Es estructurado. Tiene un acceso a memoria de bajo nivel mediante el uso de punteros, además tiene un conjunto reducido de palabras clave. Es un lenguaje compilado y de medio nivel.

- C#: es un lenguaje de programación orientado a objetos, la sintaxis deriva de C y C++. Es un lenguaje compilado y de alto nivel además de definirse como un lenguaje imperativo.

- Java: es un lenguaje de programación orientado a objetos, de alto nivel. Tiene un conjunto amplio de bibliotecas, es simple, distribuido y compilado a la vez, además es portable, indiferente a la arquitectura y de alto rendimiento.

- JavaScript: es un lenguaje interpretado, orientado a objetos, es imperativo y estructurado, es de tipo dinámino, está formado en su totalidad por objetos, que son arrays asociativos.

- PHP: es un lenguaje de programación de propósito general. Es interpretado y del lado del servidor. Es ampliable mediante módulos. Es libre y permite aplicar técnicas de programación orientada a objetos. Es un lenguaje que se interpreta en ejecución, es decir, es interpretado.

- Prolog: es un lenguaje de programación lógico e interpretado usado en el campo de la inteligencia artificial. Está basado en el concepto de razonamiento, mediante una base de datos el sistema es capaz de hacer razonamientos.

6. ¿Qué es un intérprete?

Es un programa informático utilizado para traducir los programas escritos en lenguaje de alto nivel, que nos da la apariencia de ejecutar directamente las operaciones especificas en el programa fuente con las entradas proporcionadas por el usuario.

Mediante el uso de un interprete un solo archivo fuente puede producir resultados iguales solo si es compilado a distintos ejecutables a cada sistema.

Los programas interpretados suelen ser más lentos debido a la necesidad de traducción del programa en ejecución, pero son más flexibles de programar y depurar.

7. Realiza un resumen/esquema de las fases del desarrollo de software.

Las fases del desarrollo de software son las siguientes:

- Análisis: Se extraen los requisitos del producto de software. Donde podemos utilizar técnicas como, entrevistas, desarrollo conjunto de aplicaciones, planificación conjunta de requisitos, brainstorming, prototipos o casos de uso.

- Diseño: Consiste en el diseño de los componentes del sistema que dan respuesta a las entidades de negocio. Existen dos tipos de diseño, estructurado, formado por diseño de datos, arquitectónico e interfaz, y el diseño a nivel de componentes.

- Codificación: Se traduce el diseño a código. Es la parte más obvia del trabajo de ingeniería de software y la primera en que se obtienen resultados “tangibles. Existe una serie de praxis orientada a la legibilidad de código para el posterior desarrollo del proyecto.

- Pruebas: Aquí comprobamos que el software realice correctamente las tareas indicadas en la especificación. Es una buena praxis realizar pruebas a distintos niveles y por equipos diferenciados del de desarrollo. Se fijan las tareas de verificación y validación.

- Documentación: Se lleva a cabo el manual de usuario, y posiblemente un manual técnico con el propósito de mantenimiento futuro y ampliaciones al sistema. Las tareas de esta etapa se inician ya en el primera fase pero sólo finalizan una vez terminadas las pruebas. La documentación se puede dividir en, documentación del proceso y documentación del producto.

- Explotación: Está etapa consiste en la instalación y puesta en marcha del producto y se realizará , la estrategia para la implementación, las pruebas de operación, el uso operacional del sistema y el soporte al usuario.

- Mantenimiento: Se realiza un mantenimiento correctivo de errores y un mantenimiento evolutivo para mejorar la funcionalidades y/o dar respuesta a nuevos requisitos.

8. Las fases de desarrollo siguen diferentes modelos de ciclo de vida. Investiga por internet sobre los modelos en cascada y evolutivos.

Lo que conocemos como modelo en cascada es un enfoque meteorológico que ordena de forma rigurosa las etapas del desarrollo de software de tal manera que el inicio de cada etapa debe esperar a la finalización de la anterior.

De esta forma cuando se da un error en una etapa es necesario corregir y rediseñar todo lo posterior a esa etapa además de ésta. Las fases de este modelo son:

- Análisis de requisitos del software.
- Diseño del sistema.
- Diseño del programa.
- Codificación.
- Pruebas.
- Verificación.
- Mantenimiento.

Los modelos evolutivos son aquellos que permiten desarrollar versiones cada vez mas completas hasta legar al objetivo deseado. La idea principal de este modelo de desarrollo es una implantación de un sistema inicial, e ir posteriormente refinándola en una serie de versiones hasta que se desarrolle el sistema adecuado. Existen dos tipos de desarrollo evolutivo:

- Desarrollo exploratorio: El objetivo es explorar con el usuario los requisitos hasta llegar a un sistema final. Comienza con las partes que se tiene más claras. El sistema evoluciona conforme se añaden nuevas características propuestas por el usuario.

- Enfoque de prototipos: El objetivo es entender los requisitos del usuario y trabajar para mejorar la calidad de los mismos. Se comienza por definir los requisitos que no están claros para el usuario y se utiliza un prototipo para experimentar con ellos.

9. **Realiza el pseudocódigo y el diagrama de flujo para que, dada una nota entre 0 y 10, el programa muestre su nota en modo texto**

   - **Entre 5 y 6.99 -> APROBADO**
   - **Entre 7 y 8.99 -> NOTABLE**
   - **Entre 9 y 10 -> SOBRESALIENTE**

Pseudocódigo:

```
Inicio
        Leer NOTA
        Si NOTA < 5 Entonces
                Escribir “Suspenso”;
        Sino
                Si NOTA <6.99 entonces
                Escribir “Aprobado”;
                Sinosi NOTA <8.99 entonces
                Escribir “Notable”;
                Sinosi NOTA >8.99 entonces
                Escribir “Sobresaliente”;
        FinSi;
Fin;
```

Diagrama de flujo: