ENTORNOS DE DESARROLLO – UF1 (ACT. A)

¿Cuáles son los diferentes estados por los que pasa el código desde que el programador define el problema a resolver hasta que se entrega el producto que se soluciona?

El código, desde que se crea hasta que cumple su cometido, pasa por una serie de estados:

* **Código fuente.** Se trata del código que el programador realiza en un lenguaje de programación determinado siguiendo las normas de sintaxis que este tenga para que el compilador lo pueda transformar en código legible por el ordenador.
* **Código objeto.** Este es el código (binario) que el compilador a creado a partir del código fuente. Gracias a este código, el ordenador puede interpretar las instrucciones marcadas por el programador.
* **Código ejecutable.** Gracias al enlazador llegamos al código ejecutable, que es el código ya empaquetado y listo para ser ejecutado en cualquier computadora. Este nos permite saber que la compilación fue realizada correctamente y que el programa puede funcionar correctamente (si no tuviera errores de manejo), ya que su sintaxis es correcta.

¿Cuántos tipos de programación existen? Dar una breve definición de cada uno de ellos.

Los lenguajes de programación se pueden dividir en tres grandes grupos:

* **Lenguaje procedural**. Son como lista de acciones a realizar siempre en el mismo orden.

Se caracterizan por tener un punto de comienzo, un proceso y un final. Los procedimientos pueden ser funciones, rutinas o métodos. Dos lenguajes que forman parte de estos tipos son Basic y Fortran.

* **Lenguajes de programación estructurada.** Se trata de una variante del anterior pero con mayor organización interna. Se organizan en módulos más pequeños y manejables, diferencian las variables globales y locales y hacen uso de estructuras básicas como de secuencia y selección e iteración. Dos ejemplos de este tipo de lenguaje serían C y Pascal.
* **Lenguaje orientado a objetos.** Se crean estructuras de datos y operaciones que aceptan dichas estructuras. Es el programador el que diseña cual va a ser la estructura de los datos y el tipo de operaciones que aceptará dicha estructura. Al conjunto de la estructura dato y operación se le llama objeto. El programa pasa a ser un conjunto de objetos que cooperan entre ellos para conseguir un objetivo. Como ejemplo de estos lenguaje podemos mencionar Java y PHP.

Menciona dos lenguajes de programación orientados a objetos y menciona la diferencia que tienen con respecto a los lenguajes de procedimiento y estructurados.

Existen varios lenguajes que utilizan la programación orientada a objetos, dos de ellos son:

* **Java.** Se trata de un leguaje de programación orientado a objetos muy sencillo de usar con respecto a otros lenguajes. Permite trabajar con él por medio de varios ordenadores conectados en una misma red. Permite la portabilidad de un mismo programa, haciendo que este funcione en varios ordenadores con distintos sistemas operativos. Es un lenguaje de programación muy utilizado por la empresas.
* **C++.** Se trata de un lenguaje con el que se puede reducir mucho código gracias a que utiliza caracteres especiales en lugar de comandos, el programador codifica más rápidamente y con mayor concisión. Permite compilar en casi todos los ordenadores y en cualquier sistema operativo. Utiliza la programación modular, donde el código se compila por bloques y después lo enlaza todo.

En cuanto a las diferencias de este tipo de lenguajes (POO) con respecto a los lenguajes de procedimiento y estructurados podemos destacar lo siguiente:

* La POO es la evolución de la programación estructurada, mejorando las deficiencias y problemas que ocasionaba esta última.
* En POO, el análisis está orientado a objetos y en el estructurado está orientado a los procesos del sistema.
* En la programación procedimental, las funciones se denominan “procedimientos” y las estructuras de datos “registros”; en la POO hablamos de “métodos” para las funciones y “objetos” para las estructuras de datos.
* La programación procedimental usa una “llamada de procedimiento” para llamar a una función, mientras que en la POO se utiliza un “mensaje”.
* Con la POO se busca desarrollar los programas de una forma cercana a como pensaríamos en las cosas de la vida real, mientras que con la programación estructurada el programa se desarrolla paso a paso teniendo una secuencia lógica, resolviendo de principio a fin el problema en una sola estructura.
* La POO se basa en lenguajes que soportan sintáctica y semánticamente la unión entre los tipos abstractos de datos y sus operaciones, hablamos de lo que se denomina “clase”.
* La POO cuenta con el polimorfismo y la encapsulación, no así los otros tipos de lenguajes.
* Una de las diferencias más destacables es que la POO cuenta con la posibilidad de la “herencia”, característica muy potente y que permite reutilizar código.
* La POO también facilita la “reusabilidad”, la “mantenibilidad” y la “modificabilidad”.
* Como la POO permite dividir el problema en partes más pequeñas, los posibles errores quedan aislados y son más fáciles de encontrar. Gracias a esto se aumenta la fiabilidad del programa.

Menciona 3 características principales de un lenguaje de programación orientado a objetos.

Cuando hablamos de un lenguaje de programación orientado a objetos podemos decir que cuentan con unas características propias que son las que los hacen tan potentes. Hablamos de características como la “abtracción”, el “encapsulamiento”, el “principio de ocultación”, el concepto de “clase”, el “polimorfismo”, la “herencia”, los “objetos”, los “mensajes”, las “instancias” y los “métodos”. A continuación detallaré tres de ellas:

* **Abstracción.** Cada objeto es un modelo abstracto que puede realizar un trabajo, informar y cambiar su estado, y comunicarse con otros objetos en el sistema sin revelar cómo se implementan estas características. Los procesos, las funciones o los métodos también pueden ser abstraídos y cuando lo están se puede recurrir a técnicas que permiten ampliar dicha abstracción.
* **Encapsulamiento.** Con esta particularidad podemos hacer que los atributos de una clase sólo sean accesibles desde el interior de dicha clase, favoreciendo, así, la cohesión del programa.
* **Polimorfismo.** Es la característica que permite que objetos de diferente tipo puedan recibir el mismo mensaje y puedan responder de diferente forma.

Menciona las etapas de transformación por las que pasa el código objeto hasta llegar a ser código ejecutable.

El código objeto, también llamado código ensamblador, es el código binario que se ha originado a partir de traducir el código fuente con la ayuda del compilador. Esta interpretación es la traducción y ejecución simultánea del programa línea a línea.

El código objeto no lo podrá interpretar ni el ser humano ni la computadora, se trata de un código intermedio entre el código fuente y el ejecutable y sólo si existe el programa se compila, ya que si se interpreta (haciendo la traducción línea a línea del código) se traduce y se ejecuta en un solo paso.

Una vez tenemos dicho código (objeto), este pasa al enlazador, que es un programa que generará el código máquina. El enlazador interpreta los nombres simbólicos para direcciones de memoria y otros elementos.

Llegados a este punto, tenemos el código ejecutable (o código máquina) que ya es entendible por la computadora. Dicho código es binario y es el resultado de enlazar los archivos de código objeto con las rutinas y bibliotecas necesarias. Será el sistema operativo el encargado de cargar el código ejecutable en la memoria RAM y proceder a ejecutarlo.

¿Cómo se llama el programa traductor de código fuente a código objeto?

El programa encargado de traducir el código fuente a código objeto se llama compilador. El compilador puede hacer dicha traducción del lenguaje de programación usado en el código fuente a otro lenguaje de programación que suele ser un lenguaje máquina pero en ocasiones puede hacerlo en uno de bytecode o texto.

Generalmente este proceso realiza dos tareas:

* ANÁLISIS. Hace la comprobación de que el código fuente está correcto e incluye las fases que corresponden al análisis léxico, sintáctico y semántico.
* SÍNTESIS. Genera la salida del código en el lenguaje objeto y suele estar formado por una o varias combinaciones de fases de generación de código y de optimización del mismo.

La fase en que los objetos pasan a explotación, ¿se considera dentro de las fases del desarrollo de una aplicación? ¿Si, no y por qué?

Sí se considera que la fase en que los objetos pasan a explotación esté dentro de las fases de desarrollo de una aplicación. A pesar de las pruebas previas que el usuario ha podido hacer por su cuenta, la implementación en un entorno de explotación real abrirá un necesario periodo de ventana de monitorización de la instalación y de posibles ajustes o incluso vuelta atrás por mal funcionamiento.

Dicho esto, cabe destacar que una aplicación informática pasa por otras fases, a continuación las detallo:

* Planificación del proyecto. Estudio de viabilidad.
  + Documento del presupuesto. Después de la planificación, el responsable del proyecto debe valorarlo económicamente.
  + Firma del documento de inicialización del proyecto.
  + Análisis del sistema y recogida de requerimientos.
  + Diseño del sistema.
* Codificación.
* Pruebas técnicas.
* Pruebas de usuario.
* Documentación.
* Implementación en entorno de explotación (real).
* Plan de mantenimiento.

Define qué es una máquina virtual y para qué se puede utilizar.

Se trata de un software que puede simular tanto un sistema operativo como un programa dentro de un ordenador y haciendo sus funciones como si de otra máquina independiente se tratase, como un ordenador simulado dentro de otro ordenador.

Hoy en día la virtualización es algo muy usado porque cuenta con numerosas ventajas: reduce costes y riesgos, aumenta la calidad y el aprovechamiento de los equipos, hace posible y mejor aislamiento y administración de los recursos. A pesar de todo lo mencionado, la virtualización también cuenta con desventajas: un fallo en el software podría ocasionar la caída de todos los sistemas virtuales que se estén ejecutando y, para que esta virtualización sea óptima es necesario un hardware potente y, por lo tanto, más caro.

Menciona tres de las principales funciones habilitadas en un entorno de desarrollo.

Son muchas la funciones que nos ofrece un entorno de desarrollo, tres de ellas serían:

* Detección de errores de sintaxis en tiempo real.
* Identificación automática del código.
* Asistentes y utilidades de gestión y generación de código.

Menciona y define brevemente las áreas en las que se divide una pantalla básica de desarrollo.

Los IDE nos ofrecen una interfaz gráfica a los usuarios que facilitan en buena medida la elaboración del código. En esa interfaz encontramos 5 áreas:

* **Área 1. Menú principal.** Aquí se realizan las principales acciones del entorno, como la depuración, los fichero o el control del proyecto.
* **Área 2.** **Ventana de proyecto.** Muestra una división de conjunto del proyecto.
* **Área 3.** **Ventana de navegación.** Permite acceder de forma rápida a los elementos de las clases que tengamos seleccionadas.
* **Área 4.** **Ventana de edición.** Es el editor de texto a través del cual entramos en el código del desarrollo de nuestro proyecto.
* **Área 5.**  **Ventana de salida de tareas.** Muestra los errores de compilación e información seleccionada por el programador mediante palabras clave del código.