Trabajo Práctico 1 – Introducción a la Orientación a Objetos

1) ¿Cuál es el factor determinante en la aparición de técnicas de programación?

El problema principal es que la crisis del software sigue, y probablemente no terminará nunca. Una de las causas más fuertes es la huida hacia adelante" que se verifica por el aumento de complejidad de los sistemas. Así, cuando una técnica estaba resultando útil, el enorme crecimiento de la complejidad la hace obsoleta.

2) ¿Qué aspectos mejoró la Programación Modular?

La programación modular consistió en dividir el sistema en diferentes módulos o subprogramas autónomos que fueran individualmente programables, verificables y programables.

3) ¿Qué beneficios obtengo al utilizar componentes desarrollados por un tercero?

Comprar componentes desarrollado masivamente por terceros suele ser más económico, además de que permite que cada uno se dedique a lo que mejor sabe hacer.

4) ¿Qué es un Tipo de Datos Abstracto (TDA)?

Un tipo abstracto de datos (TAD) es un tipo de dato definido por el programador que se puede manipular de un modo similar a los tipos de datos definidos por el sistema.

Por ejemplo, el tipo de dato abstracto Pila tiene valores y operaciones definidas sobre las cuales se puede manipular el dato. Ejemplo: crear pila, pila vacía ,agregar un dato a la pila, sacar dato de una pila, ver el último dato ingresado (cima).

Los TDA tienen propiedades tales como ocultamiento tanto de los valores como de sus operaciones, son reutilizables (pueden crearse muchas instancias de ese tipo de dato), permiten utilizar operaciones propias de ese tipo de dato.

5) Ventajas de usar Tipos de Datos Abstractos.

1) Exponer una definición del tipo.
2) Hacer disponible un conjunto de operaciones que se puedan utilizar para manipular instancias de ese tipo.
3) Proteger los datos asociados con el tipo de modo que sólo se pueda actuar sobre ellas con las rutinas proporcionadas.
4) Hacer instancias múltiples del tipo.

6) ¿Por qué surge la necesidad de ocultar la implementación?

La necesidad de el uso del ocultamiento de la implementación surge de darle solución al problema del uso indebido de los datos por parte del usuario. El ocultamiento de la implementación consiste en usar interfaces para restringir el acceso a determinados datos y operaciones propios del tipo de dato, ocultando al usuario conceptos propios de la implementación que no debieran ser de su incumbencia.

7) ¿Por qué cree que las aplicaciones crecen en complejidad?

Cuanto mayor es el riesgo de error, mayor es la complejidad, pues las salvaguardias para evitar errores introducen gran cantidad de código y relaciones entre módulos.

8) ¿Cómo influyen las nuevas tecnologías en el desarrollo de software?

Cuando las computadoras eran muy caras y el costo de desarrollar software muy alto, los programas resolvían tareas elementales. A medida que el costo del hardware bajó, surgieron nuevos lenguajes de programación y hubo técnicas mejores de desarrollo de software, se pudieron crear sistemas cada vez más complejos, por lo que esos hardware, lenguajes y técnicas resultaron insuficientes. Con el tiempo el hardware siguió abaratándose, surgieron ambientes de desarrollo amigables y se crearon metodologías de desarrollo que permitiera manejar la complejidad que se estaba precisando, pero los requerimientos crecieron,en complejidad.

El auge de las interfaces gráficas de usuario, sobre todo luego de 1995, ha impulsado nuevas formas de desarrollo, entre ellas la programación visual y la programación por eventos.
La World Wide Web también ha impulsado la aparición de aplicaciones especiales para ese medio. Estas técnicas, a su vez, precisan de nuevas metodologías de desarrollo y programación.

9) Enumere y describa brevemente los factores que tienen influencia en la Calidad del Software.

A) Eficiencia: es la capacidad para hacer un buen uso de los recursos de la máquina

Transportabilidad/portabilidad: es la calidad con la que un software puede ser transportado sobre diferentes sistemas físicos o lógicos

B) Verificabilidad: es la facilidad de verificación de un software es su capacidad para soportar procedimientos de validación y de aceptar juegos de test

C) Integridad: es la capacidad de un software para proteger sus propios componentes contra los procesos que no tengan derecho de acceso

D) Fácil de utilizar: Un software es fácil de utilizar si se puede comunicar con él de manera correcta

E) Corrección: Capacidad de los productos software de realizar exactamente las tareas definidas por su especialización

F) Robustez: capacidad de los productos de software de funcionar incluso en situaciones anormales

G) Extensibilidad: Facilidad que tienen los productos de adaptarse a cambios en su especificación existen dos principios fundamentales para esto:

Diseño simple;
Descentralización;

H) Reutilización: capacidad de los productos para ser reutilizados, en su totalidad, o en parte, en nuevas apps

I) Compatibilidad: Facilidad de los productos para ser combinados por otros

10) Qué objetivos impulsó la Programación Orientada a Objetos?

Los objetivos que fueron impulsados por la programación orientada a objetos fueron los siguientes:

A) Reducción de la brecha entre el mundo de los problemas y el mundo de los modelos.
B) Aumento del nivel de complejidad de los sistemas.
C) Fomentar la reutilización y extensión del código.
D) Uso de prototipos.
E) Programación en ambientes de interfaz de usuario gráfica.
F) Programación por eventos.

Alumnos:
Gonzalo Errandonea
Sebastián Maldonado
Gabriel Ramos
Tomas Alaluf