EXAMEN DE TEORIA POO ENERO 2021

1. ¿Qué es un patrón de diseño? Define el patrón SINGLETON.

2. Indica el factor de calidad del software que no se cumple en los siguientes casos:

a) El software funciona con operaciones sencillas, pero falla cuando se usa en operaciones permitidas, pero más complejas.

b) El software no puede ejecutarse nada más que en un sistema operativo concreto.

c) El software es atacado por piratas informáticos con facilidad.

d) La página web no está adaptada a personas con discapacidad.

e) La interfaz de usuario es excesivamente compleja.

3. Para un fichero de código fuente que se llama persona escribe el código correspondiente a las guardas de inclusión que se escribe al principio del fichero y el código de dichas guardas de inclusión que se escribe al final del fichero. El resto del contenido del fichero déjalo en blanco

4. Describe el requisito para una buena descomposición modular denominado “interfaces explicitas”.

5. ¿Por qué en C++ conviene pasar objetos grandes como referencia constante?

EXAMEN RESUELTO TEORIA POO ENERO 2021

1. ¿Qué es un patrón de diseño? Define el patrón SINGLETON.

Un patrón de diseño es una descripción de las clases y objetos que lo componen, y de sus relaciones entre sí. Cada patrón está especializado en resolver un problema de diseño general en un determinado contexto.

Singleton. Es un patrón de diseño creacional. Nos asegura que solo exista una única instancia/objeto de una clase. Es útil cuando se necesita exactamente un objeto para coordinar acciones a través del sistema. Encapsula un recurso del que solo hay una instancia y lo pone a disposición de toda la aplicación. Puede ser hardware, un servicio, un dato global, etc.

Ejemplos.

- Cuando hay un único recurso con el que tener interface desde distintos lugares de la aplicación.
- En un smartphone solo hay una touch screen para todos los objetos que la usan a la vez.
- Configuración global del sistema.
- Variables globales se administran mejor en una clase con una única instancia.

2. Indica el factor de calidad del software que no se cumple en los siguientes casos:

a) El software funciona con operaciones sencillas, pero falla cuando se usa en operaciones permitidas, pero más complejas.

Integridad:(Factor externo): El producto no debe corromperse por el simple hecho de su utilización masiva o por una gran acumulación de datos, o por operaciones complejas posibles, pero no previstas al cien por cien.

b) El software no puede ejecutarse nada más que en un sistema operativo concreto.

Portabilidad:(Interno/Externo): Es la capacidad o la facilidad del producto de ejecutarse en otro hardware diferente o en otro sistema operativo diferente.

Aquí es muy importante que el programa no haga uso de características de bajo nivel del hardware o que aislé la parte dependiente del hardware para que al portarlo solo sea necesario modificar dicha parte.

c) El software es atacado por piratas informáticos con facilidad.

Seguridad:(Factor interno): Es la capacidad del producto de proteger sus componentes de usos no autorizados y de situaciones excepcionales de pérdida de información. Para ello debe prever mecanismos de control de acceso y también, desde el otro punto de

vista copias de seguridad, procesos rutinarios de comprobación, etc.

d) La página web no está adaptada a personas con discapacidad.

Accesibilidad informática:(Factor externo): Consiste en el acceso a la información sin limitación alguna por razones de deficiencia, incapacidad o minusvalía. Con el fin de poder elaborar, reproducir, manipular, etc. dicha información, así como el acceso a herramientas y opciones

e) La interfaz de usuario es excesivamente compleja.

Facilidad de uso:(Factor externo): Facilidad al introducir datos, interpretar datos, comprender errores, interpretar resultados, etc. Para usuarios con diferentes formaciones y aptitudes. Instalación, operación y supervisión. Capacidad multilingüe

3. Para un fichero de código fuente que se llama persona escribe el código correspondiente a las guardas de inclusión que se escribe al principio del fichero y el código de dichas guardas de inclusión que se escribe al final del fichero. El resto del contenido del fichero déjalo en blanco

```
#ifndef PERSONA_H
#define PERSONA_H
/*
Código
*/
#endif
```

4. Describe el requisito para una buena descomposición modular denominado "interfaces explicitas".

Las interfaces deben ser obvias a partir de su simple lectura.: La claridad del módulo, afecta en gran medida a su calidad y, por tanto, es determinante para que un módulo sea bueno. Lo mismo ocurre si un módulo debe llamar a otro, debe entenderse de forma lógica el motivo de esa llamada

5. ¿Por qué en C++ conviene pasar objetos grandes como referencia constante?

Se utiliza la referencia para evitar el proceso de copia del objeto y el consecuente gasto en memoria y tiempo de ejecución, y pasar el objeto como constante evita que este pueda ser modificado