

Recopilación de preguntas teóricas de exámenes finales y de parciales

- Tener en cuenta que hay preguntas que son de años anteriores y se daban temas que en este año no se han dado y no se tomarían.
- Como no está depurada la lista pueden haber muchas preguntas repetidas.

Ejercicio 5 Responda: a) ¿Qué es una plantilla? ¿Qué significa instanciar una? b) Explique cómo es posible ordenar un `std::vector` de structs según alguno de sus miembros. c) ¿Para qué sirve la palabra reservada `virtual`? d) ¿Qué significa que una clase sea abstracta? e) ¿Es posible almacenar objetos de tipo `string` en un archivo binario? Explique. f) e) ¿Para qué puede utilizarse la palabra reservada `const` en la definición de un método

Ejercicio 4 (10pts): Responder a) Qué es un método virtual puro?; b) Qué significa que una clase A es contenedora de otra B?; c) Qué entiende por encapsulamiento? Se puede evitar en C++?

Ejercicio 4 (20 pts)

Responda y explique (ninguna respuesta se considerará válida sin su correspondiente justificación)

- a) (3 pts) ¿Por qué no es recomendable guardar objetos de tipo `std::string` en archivos binarios?
- b) (3 pts) Nombre y explique al menos tres diferencias entre archivos de texto y archivos binarios.
- c) (3 pts) ¿Qué significa instanciar una plantilla? ¿Puede ésta acción realizarse con cualquier tipo de dato? Explique.
- d) (3 pts) Las operaciones de inserción y remoción de un elemento, ¿se realizan de manera más eficiente en un vector o en una lista? Explique.
- e) (3 pts) ¿Qué significa que una clase sea abstracta? ¿Cómo se especifica ésta situación en C++?
- f) (3 pts) ¿Qué efectos tienen en C++ las palabras reservadas `private`, `public` y `protected`?
- h) (2 pts) ¿Qué es un iterador? ¿Para qué sirve?

Ejercicio 4 (20 pts) Responda: a) ¿Qué significa polimorfismo? b) ¿Qué significa que una clase sea abstracta? ¿Qué ventajas tiene? c) ¿En qué casos es necesario redefinir el constructor de copia? d) Proponga un ejemplo en el cual la instanciación de una plantilla de error. e) Enumere y explique los componentes de la biblioteca STL.

Ejercicio 4 (20pts) Explique: a) ¿Para qué sirve la palabra reservada `virtual`? b) ¿Qué significa que una clase sea abstracta? ¿cómo se implementa esto en C++? c) Indique dos formas de obtener una dirección de memoria válida para un puntero. d) Es posible almacenar un objeto de tipo `string` directamente en un archivo binario?

Ejercicio 4 (20 pts) Responda:

- a) Enumere y explique 3 ventajas de utilizar memoria dinámica.
- b) ¿Qué es una plantilla? Proponga un ejemplo de la sintaxis de una plantilla de clase y de una plantilla de función.

- c) ¿Qué efecto tiene la utilización de la palabra reservada `virtual`?
- d) ¿Qué ventajas ofrece el uso de archivos de texto sobre archivos binarios? ¿Qué ventajas ofrece el uso de archivos binarios sobre archivos de texto?
- e) Enumere y explique 3 de los componentes de la STL.

Ejercicio 4 (15 pts) a) Indique 3 formas de acceder al último elemento de un vector dinámico de la STL, señalando las ventajas de cada una. b) ¿Puede almacenarse un objeto `string` en un archivo? Justifique su respuesta. c) Si se necesitara almacenar el valor numérico 13789 en un archivo, ¿de qué manera ocuparía menos espacio? ¿guardando los datos como texto o como binarios?. Explique por qué. d) Defina y explique los 3 componentes principales de STL: contenedores, iteradores y algoritmos.

Ejercicio 4 (15 pts) Explique: a) Porqué la secuencia `list` tiene su propio método `sort` (ya que el algoritmo `sort` no está disponible para listas) b) ¿A qué se denomina “instanciación” o “especialización” de una plantilla? c) ¿Se puede almacenar cualquier objeto (de cualquier tipo) en un archivo binario? d) ¿Cuáles son las diferencias entre `list<T>::iterator` y `vector<T>::iterator`?

Ejercicio 4 (20pts) Explique: a) ¿Cuáles son las principales diferencias entre los contenedores `std::list` y `std::vector`? ¿Cuándo convendría usar cada uno? b) ¿Qué es un iterador? c) ¿Para qué sirve la programación genérica? d) ¿Cuál es la diferencia entre un objeto de la clase `std::string` y un `c-string`?

Ejercicio 4 (15 pts) Explique: a) ¿Qué es y para qué sirve un objeto de tipo `istream_iterator`? b) ¿Se puede almacenar cualquier objeto (de cualquier tipo) en un archivo binario? c) ¿Cuáles son las diferencias entre `list<T>::iterator` y `vector<T>::iterator`? d) ¿A qué se denomina “instanciación” o “especialización” de una plantilla?

Ejercicio 4 (15 pts)

Explique: a) ¿Qué devuelve y para qué sirve la función `back_inserter()`? b) ¿Qué devuelve el operador `sizeof()` aplicado a un objeto de tipo `string`? ¿Por qué? c) Los datos generados por un programa C++ deben registrarse en un archivo, el cual debe utilizarse como entrada de un programa escrito en Java, ¿qué tipo de archivo sugiere emplear? d) Mencione dos métodos que permitan controlar la apertura de un archivo en C++ y explique su función.

Ejercicio 4 (25 pts) Explique a) ¿Qué entiende por programación genérica? b) Al instanciar una plantilla, ¿cuándo se puede omitir el tipo de dato, y cuándo debe explicitarse? Dé ejemplos. c) ¿Qué es un iterador? ¿Por qué las funciones de la biblioteca `<algorithm>` reciben iteradores en lugar de contenedores? d) ¿Es correcto escribir un arreglo `int V[10]` en un archivo binario B con la sentencia `"B.write(reinterpret_cast<char*>(V),10*sizeof(int));"`? ¿Y si V fuese un `vector<int> V`, sería correcto utilizar `"B.write(reinterpret_cast<char*>(&V),V.size()*sizeof(int));"`? e) ¿Cuáles son las ventajas y desventajas de utilizar archivos binarios frente a utilizar archivos de texto?

Ejercicio 4) a) considere las acciones siguientes: `int *p; p=new int; *p=25;` si `p` almacena la dirección `0x3a0f30`, ¿qué se obtiene de `cout<<p+1<<" "<<*p+1<<" "<<sizeof(*p)<<endl;` b) Señale las diferencias entre las clases contenedoras que representan secuencias en la STL; c) ¿Qué es una clase amiga? Y una función amiga? d) Proponga un ejemplo de empleo de `this`.

Teoría [15 pts] Explique: a) ¿Para qué sirven los operadores *new* y *delete*? b) ¿Qué es un *constructor de copia*? ¿En qué casos es necesario definirlo? ¿Qué ocurre con el mismo en las clases en las que no se define? c) Se declara un arreglo mediante "double v[10]"; indique dos formas de obtener la dirección del 5to elemento.

Ejercicio 4 (10 pts) Responda: a) ¿Qué es una plantilla? ¿Qué significa instanciar una? b) Explique cómo es posible ordenar un `std::vector` de structs según alguno de sus miembros. c) ¿Es posible almacenar objetos de tipo string en un archivo binario? Explique. d) ¿Qué es un iterador y para qué sirve?

Ejercicio 4 (15pts) Explique a) Puede abrir un archivo binario en modo texto? y viceversa? b) ¿Es posible escribir un objeto de tipo `std::string` en un archivo binario? Explique su respuesta. c) Explique cómo es posible ordenar un `std::vector` de structs según alguno de sus miembros.

Ejercicio 4 (20pts) a) Señales las diferencias entre un `std::list` y un `std::vector`. ¿Cuándo utilizaría uno y otro? b) ¿Qué entiende por "plantilla" de clase o de función? ¿Cómo se utiliza una plantilla de clase? ¿Y una de función? c) ¿Se puede almacenar cualquier clase o struct en un archivo binario? ¿Por qué? d) ¿Qué es un puntero? ¿Para qué se utiliza en C++? ¿Qué ventajas tiene su uso?

Ejercicio 5. Responda: a) ¿Qué es una plantilla? ¿Qué significa instanciar una? b) Explique cómo es posible ordenar un `std::vector` de structs según alguno de sus miembros. c) ¿Para qué sirve la palabra reservada *virtual*? d) ¿Qué significa que una clase sea abstracta? e) ¿Es posible almacenar objetos de tipo string en un archivo binario? Explique. f) e) ¿Para qué puede utilizarse la palabra reservada *const* en la definición de un método

Ejercicio 4 (20 pts) a) Puede abrir un archivo binario en modo texto? y viceversa? b) ¿Es posible escribir un objeto de tipo `std::string` en un archivo binario? Explique su respuesta. c) Explique las diferencias entre archivos de texto y archivos binarios. d) Por qué usaría `list` o `vector` en un programa? Explique. e) ¿Qué significa instanciar una plantilla o template? f) ¿Qué relación existe entre arreglos estáticos y punteros? g) ¿Qué hace el operador *new*?

Ejercicio 4. (20 pts) a) ¿Qué es un iterador en la STL? ¿Es lo mismo que un puntero? b) Mencione y explique brevemente al menos tres diferencias entre archivos de texto y archivos binarios. c) Antes de codificar en C++, como reconoce si 2 clases pueden relacionarse a través de composición o agregación? d) ¿Cómo se implementa el encapsulamiento en C++?

Ejercicio 4 (20 pts) Responda: a) ¿Qué significa polimorfismo? b) ¿Qué significa que una clase sea abstracta? ¿Qué ventajas tiene? c) ¿En qué casos es necesario redefinir el constructor de copia? d) Proponga un ejemplo en el cual la instanciación de una plantilla de error. e) Enumere y explique los componentes de la biblioteca STL.

Ejercicio 4 (15 pts) Responda: a) ¿Qué es una plantilla? ¿Qué significa instanciar una? b) Explique cómo es posible ordenar un `std::vector` de structs según alguno de sus miembros. c) ¿Para qué sirve la palabra reservada *virtual*? d) ¿Qué significa que una clase sea abstracta? e) ¿Es posible almacenar objetos de tipo string en un archivo binario? Explique. f) ¿Qué es un iterador? ¿Para qué sirve?

Ejercicio 4 (20 pts) Responda y justifique: a) ¿Qué entiende por polimorfismo? ¿Para qué sirve? b) ¿Qué es un iterator? c) ¿Cuáles son las ventajas de utilizar memoria dinámica (con new y delete)? d) ¿Qué significa que una función F sea amiga de una clase C? ¿Cómo se declara dicha amistad?

Ejercicio 4 (20 pts)

a-¿Qué es un objeto? ¿Qué es una clase? b-¿Cuándo se invoca al constructor de un objeto? ¿Cuándo al destructor? c-¿Cuál es la diferencia, en C++, entre una clase y un struct? ¿Cuándo se utiliza cada una? d-¿Qué implica que un atributo se declare con el modificador static? Ejemplifique.

Ejercicio 4 (20 pts)

a) ¿Qué es herencia múltiple? b) ¿Para qué sirve la palabra reservada virtual? c) ¿Qué es una clase abstracta? d) ¿En qué se diferencian agregación y herencia? e) Indique dos formas de obtener una dirección de memoria válida para un puntero.

Ejercicio 4 (10pts): Responder a) Qué es un método virtual puro?; b) Qué significa que una clase A es contenedora de otra B?; c) Qué entiende por encapsulamiento? Se puede evitar en C++?

Ejercicio 4) 20 pts

a) ¿En qué se diferencia un método virtual de un método virtual puro? b) Explique en qué consiste el constructor por defecto, y cuándo es necesario definirlo explícitamente. c) Mencione dos ventajas y dos desventajas del uso de arreglos dinámicos. d) ¿Qué es el puntero this? Mencione un caso en el que sea necesario el uso de este puntero.

Ejercicio 4 (20 pts) Responda:

a) ¿Cuándo una clase es abstracta?
b) Enumere y explique 3 de los componentes de la STL.
c) ¿Es posible guardar un objeto de tipo `vector<int>` en un archivo binario? Explique.
d) ¿Qué significa instanciar una plantilla?
e) ¿Qué ventajas ofrece el uso de archivos de texto sobre archivos binarios? ¿Qué ventajas ofrece el uso de archivos binarios sobre archivos de texto?

Ejercicio 4 (20 pts) a) ¿Cuándo utilizaría una relación de composición entre clases? b) Escriba un ejemplo sencillo utilizando la función `sort(iterator, iterator, función)` de la STL. c) Se tiene un puntero `int *p` a un arreglo de `n` elementos. Muestre por pantalla los elementos del arreglo, sin utilizar corchetes. d) ¿En qué caso es *necesario* sobrecargar un operador como función global?

Ejercicio 4 (20 pts) Explique: a) ¿Cómo identifica cuándo debe usar herencia y cuándo composición? b) ¿Qué significa que una clase sea “abstracta”? c) ¿Puede sobrecargar al operador `>>`, para leer una instancia de una clase mediante `cin`, como método de la clase? ¿Por qué? d) Señale ventajas y desventajas de utilizar `new` y `delete` para gestionar la memoria? e) ¿Qué es y por qué es importante el “principio de ocultación”?

Ejercicio 4 (20pts) a) ¿Qué es un `cstring` (cadena estilo C)? ¿Cuándo conviene usarlo en lugar de un `std::string`? b) ¿Qué es un iterator? ¿Qué diferencia hay entre un iterator de lista y uno de vector? c) ¿Cómo se determina el compilador con qué tipos de datos debe especializar una plantilla? d) Explique cómo se podría ordenar un `std::vector` de structs según alguno de sus miembros.

Ejercicio 4 (20pts) a) ¿Qué entiende por programación genérica? b) ¿Cuáles son las ventajas y desventajas de utilizar archivos binarios frente a utilizar archivos de texto? c) ¿Cuáles son las diferencias entre un vector y una lista? ¿En qué casos conviene usar uno o el otro? d) ¿Es correcto escribir un arreglo "int V[10]" en un archivo binario B con la sentencia "B.write(reinterpret_cast<char*>(V),10*sizeof(int));"? ¿Y si V fuese un "vector<int> V", sería correcto utilizar "B.write(reinterpret_cast<char*>(&V),V.size()*sizeof(int));"?

Ejercicio 4 (20 pts) Explique: a) ¿Cómo determina el compilador con qué tipo en particular debe especializar un template de clase? ¿Y uno de función? b) ¿Qué es un cstring (cadena estilo C)? c) ¿Qué es un iterador? ¿Qué diferencia hay entre un iterador de lista y uno de vector? d) ¿Es correcto escribir un arreglo "int V[10]" en un archivo binario B con la sentencia "B.write(reinterpret_cast<char*>(V),10*sizeof(int));"? ¿Y si V fuese un "vector<int> V", sería correcto utilizar "B.write(reinterpret_cast<char*>(&V),V.size()*sizeof(int));"?

Ejercicio 4 [10 pts] a) Puede abrir un archivo de texto como binario? Si responde sí, para qué serviría? b) la siguiente función genérica tiene el prototipo: **bool compara_intercambia(T &x,T &y);** y determina qué dato es mayor e intercambia ambos parámetros; es posible pasarle 2 objetos de una clase? explique; c) Explique la clase contenedora map de STL y ejemplifique; d) Puede escribir una programa C++ que acceda a un archivo de textos que solo sabe su nombre y que no sabe cómo están dispuestos los datos? explique.