**Preguntas Teóricas de Taller:**

**Imperativo:**

**Ordenación de vectores  
1. ¿En qué consiste la ordenación de vectores?**

Siguiendo un patrón único intercambio los elementos de un vector ya cargado para acomodarlos con un cierto criterio.

**2. ¿Qué es y qué hace un algoritmo de ordenación?**

Serie de pasos los cuales consiguen dado un vector de elementos, intercambiar esos elementos en distintas posiciones tal que el resultado será el mismo vector con sus elementos ordenados en base a un criterio.

**3. ¿Mencione al menos dos algoritmos de ordenación de vectores? ¿Cómo funcionan? Ejemplifique**

Selección, intercambio, inserción.

Intercambio: Se comparan e intercambian datos adyacentes según corresponda

Requiere n-1 pasadas sobre el vector

Si se quiere ordenar el vector de menor a mayor, al final de cada pasada el mayor queda al final del arreglo (no es necesario re – compararlo).

Inserción: Se parte de una secuencia de dos ítems y se ordena.

En cada pasada se “agrega” un ítem y se inserta en la posición correspondiente en el arreglo ordenado.

**Recursión**

**1. Defina el concepto de Recursión. Explique las principales características que se presentan en un algoritmo recursivo.**

La recursión es una metodología para resolver problemas. Permite resolver un problema P por resolución de instancias más pequeñas P1, P2, … Pn del mismo problema.

El problema Pi es de la misma naturaleza que el problema original, pero en algún sentido es más simple.

Un algoritmo recursivo involucra:

-Un caso base.

-Al menos una condición de terminación (implícita / explícita)

-Al menos una auto invocación (llamada recursiva). Se debe garantizar que en un número finito de autoinvocaciones se alcanza la condición de terminación.

**2. ¿Una solución recursiva puede tener más de un caso base? Justifique.**

SI. Haber llegado al final, que el elemento no exista o el vector este vacío y cuando encuentro el dato a buscar.

**3. Describa dos ejemplos de problemas recursivos con más de un caso base. Implemente dicha solución.**

Búsqueda de un elemento en un árbol, vector, lista (Que no exista el elemento, encontrarlo). Ejemplo de implementación de Búsqueda de un elemento en un árbol:

Procedure Buscar (a: árbol; var p: árbol; dato: integer);

Begin

If (a=nil) then

p:= nil (Arbol Vacio)

else

if (dato= a^.dato) then

p:= a (Dato encontrado)

else

if (dato < a^.dato) then

Buscar (a^.HI,p,dato)

else

Buscar(a ^.HD,p,dato);

End;

**4. Enuncie un problema que justifique una solución recursiva e impleméntela.**

Un problema en el cual se puede resolver mediante la recursión es el factorial de un número. Su implementación es:

Function potencia (x,n: integer): real;

begin

if (n = 0) then

potencia:= 1

else

potencia := x \* potencia(x,n-1);

end;

**Merge de listas**

**1. ¿En qué consiste el Merge de listas? Mencione cuales son las precondiciones que se requieren para realizar la operación.**

La operación de Merge consiste en generar una nueva estructura de datos (arreglos, listas) ordenada a partir de la mezcla de dos o más estructuras de datos previamente ordenadas. Las estructuras que se combinan guardan el mismo orden lógico interno (por ejemplo, datos ordenados alfabéticamente). La precondición para realizar la operación es que las estructuras a mezclar estén ordenadas por la misma condición.

**2. Explique detalladamente los pasos a seguir para realizar la operación de Merge de dos listas simplemente enlazadas.**

Debemos hallar el mínimo elemento entre las dos listas, se inserta el elemento en la nueva estructura y se elimina de la anterior. Se repite él algoritmo hasta que ambas listas respondan a la condición nil.

**3. ¿En qué se diferencian las operaciones de merge y merge acumulador? Ejemplifique.**

La operación de merge consiste en agrupar en una misma estructura elementos provenientes de distintas, ordenados por un criterio. Un merge acumulador consiste en acumular elementos con un mismo criterio e insertarlos en una nueva estructura siguiendo una condición de orden.

**Árbol binario de búsqueda**

**1. ¿Qué es un árbol?**

Un árbol es una estructura de datos homogénea, dinámica, no lineal y acceso secuencia que satisface tres propiedades:

Cada elemento del árbol se relaciona con cero o más elementos (hijos).

Si el árbol no está vacío, hay un único elemento (raíz) y que no tiene padre (predecesor).

Todo otro elemento del árbol posee un único padre y es un descendiente de la raíz.

**2. ¿Qué es un árbol binario?**

Un árbol binario es una estructura de datos en la cual cada nodo puede tener un hijo izquierdo y un hijo derecho. No pueden tener más de dos hijos (de ahí el nombre "binario").

**3. ¿Qué es un árbol binario de búsqueda?**

Un ABB es un árbol binario el cual sus elementos se encuentran ordenados. Cada nodo tiene un valor que:

* Es más grande que el valor de todos los nodos del subárbol izquierdo
* Es menor que el valor de todos los nodos del subárbol derecho

Utilidad más importante ➔ Búsquedas el tiempo medio es O (log n))

**4. Defina y explique las principales características del tipo de dato Árbol Binario de Búsqueda. Características y ventajas.**

**5. Describa detalladamente la operación Insertar un elemento en un árbol binario de búsqueda.**

Primero necesito dos parámetros, uno es el árbol y el otro es el dato que quiero insertar. Utilizo una variable local de tipo árbol para guardarme los datos recibidos por parámetros y asegurarme de no perder datos. Luego pregunto si el árbol recibido como parámetro esta vacío; si lo está, guardo los datos en la variable local creada y luego le asigno esa variable local con los datos ya guardados a la variable que recibí como parámetro. En caso de que el árbol que recibí como parámetro no este vacío, invoco la recursión preguntando si el dato que recibí es mayor o menor al dato del árbol para poder ver donde lo ubico, es decir si el dato que recibo va para el hijo izquierdo o derecho.

**Árbol binario de búsqueda**

**1. ¿Cuáles son los recorridos clásicos sobre un ABB?**

En orden (hijo izquierdo, raíz, hijo derecho), pre orden (raíz, hijo izquierdo, hijo derecho), post orden (hijo izquierdo, hijo derecho, raíz).

**2. Describa detalladamente la operación de Imprimir en Ordenen un árbol binario de búsqueda.**

Para imprimir en orden en un árbol de búsqueda, puedo utilizar un método recursivo el cual se auto invoque con el HI hasta llegar al final, imprima los valores subiendo hacia la raíz, y luego recursivamente, realice lo mismo con la pata derecha.

**3. Plantee un ejemplo de árbol binario de búsqueda y explique detalladamente cuales serían los pasos a seguir para realizar un recorrido acotado en dicho árbol.**

Para realizar un recorrido acotado, debo tener en cuenta tres recursiones, que este entre la raíz, que este a la derecha, o a la izquierda. También que no sea nil.

**4. Suponga que dispone de un árbol binario con datos de alumnos (Apellido, Nombre y DNI) ordenado por DNI. Detalle los pasos necesarios para conocer el apellido y nombre del alumno que posee el mínimo valor de DNI.**

Recorro el árbol mientras no sea nil, hacia el hijo izquierdo.

**Árbol binario de búsqueda**

**1. Detalle los pasos necesarios para borrar un elemento de un árbol binario de búsqueda.**

Cuando es hoja, cuando tiene un hijo, cuándo tiene los dos.

Cuando el elemento es una hoja, la elimino. Cuando tiene dos hijos busco el más grande del sub árbol izquierdo del elemento a borrar, lo reemplazo y lo elimino. Cuando tiene un hijo lo reemplazo.

**Programación orientada a objetos**

**1. ¿Qué es un objeto? ¿Cómo se compone? Ejemplifique.**

Representación abstracta de cualquier cosa que nos rodea. Cualquier cosa que podamos nombrar y que nos sea de utilidad. Todo es un objeto.

- Características físicas, nombre, estado, atributos, comportamientos.

**2. ¿Qué es una clase? ¿Qué es una instancia?**

- Una clase es la implementación de las características, el estado y comportamiento de un objeto.

- La instancia es la representación viva de ese objeto. Quien almacena valores y se ejecuta es la instancia de la clase.

**3. ¿Cómo se crea un objeto? ¿Cuáles son los pasos que se siguen en Java para la creación de un Objeto?**

Utilizando el new, Alumno a = new Alumno (…);

**Constructores**

**1. Describa el concepto de constructor (objetivo, sintaxis de declaración, sintaxis de uso, concepto de sobrecarga).**

Es un método especial porque no se llama, sino que es invocado por el operador new al momento de crear una instancia. Su objetivo es darle un valor inicial a la instancia. En su sintaxis tiene el mismo nombre que la clase y no tiene tipo de retorno. Se diferencia por la firma.

Un Constructor es un método especial porque es el único método que no podemos invocar, es invocado por el operador new. Nos permite dar un valor inicial a todas mis variables e instancia.

-Todo constructor debería tener el mismo nombre que el de la clase, puede haber más de uno. No tiene un tipo de retorno. Respetar las firmas.

**2. ¿Es posible generar una clase sin un constructor? Justifique.**

Sí es posible, debido a que el compilador genera un constructor por defecto que inicializa las variables en valores por defecto.

**Herencia**

**1. Defina el concepto de herencia. Describa dos ejemplos donde la herencia sea de utilidad para reutilizar código.**

Es un mecanismo de POO que permite que una clase cualquiera tenga el estado interno y el comportamiento de otra clase además del propio. Permite reutilizar código y es más fácil de mantener, de decir, si hay un error, lo corrijo directamente en la clase padre.

**2. Describa los conceptos de clase abstracta y método abstracto, indicando además para qué se usan y sintaxis de declaración en Java.**

La clase abstracta es una clase que no puede ser instanciada (no se pueden crear objetos).

Uso: define características y comportamiento común para un conjunto de clases

(subclases).

**Método abstracto: dice que hace un objeto no dice como hacerlo.**

public abstract class NombreClase {}

public abstract tipoRetorno nombreMetodos ()

**3. Describa una jerarquía de clases donde aplique el uso de clase abstracta y método abstracto. Justifique la aplicación.**

**Programación concurrente**

**1. Enumere y describa tres características que generalmente existen en un programa concurrente.**

Un programa concurrente se basa en la simultaneidad en la ejecución de múltiples procesos. Sus características son: la comunicación entre procesos, la sincronización y debe haber más de un proceso (obligatorio).

**2. Para realizar la comunicación entre procesos ¿qué técnicas existen? Explíquelas.**

Para la comunicación entre procesos hay dos técnicas, memoria compartida (donde hay una variable en común, donde un proceso tiene que escribir y otro tiene que leer), memoria distribuida (se basa en la comunicación entre procesos mediante mensajes donde un proceso lo envía y otro lo recibe).

**3. Si tuviera que compartir información entre dos robots del ambiente dado en el curso. ¿Qué opciones tendría? Detalle su respuesta.**

Tendría las opciones de memoria compartida y memoria distribuida.

**4. Suponga que en el entorno R-info (el que utilizó en la  
práctica) no existiera el pasaje de mensajes, usted ¿podría afirmar que los procesos no se pueden comunicar? JUSTIFIQUE su respuesta.**

Los procesos sin el envío de mensajes podrían comunicarse igual, ya que en r-Info las esquinas son los recursos compartidos, y mediante el depósito de una flor o papel el otro robot debería hacer una u otra cosa.

**Programación concurrente**

**1. Explique detalladamente cómo funciona la recepción de un mensaje (sintaxis, variantes).**

recibirMensaje (datoQueMeEnvían, quienMeLoEnvía), la instrucción traba el robot hasta que el robot receptor reciba el mensaje que espera, esto se denomina método Sincrónico; por otro lado, el método asincrónico el robot sigue su ejecución a pesar de no recibir ningún mensaje. R-Info utiliza mensajes sincrónicos.

**2. ¿Cuántos tipos de áreas existen en el ambiente r-info? ¿Para qué se usa cada una? ¿Qué consideraciones se debe tener en cada una de ellas?**

Existes tres tipos de áreas: ***Privada*** (solo puede acceder un robot a su área privada) hay que considerar que solo un robot la acceda, ***Compartida*** (todos los robots tienen acceso), ***Parcialmente Compartida*** (solamente la pueden acceder un subconjunto de robots) tengo que considerar bloquear esquinas para evitar chocare con otros robots (semi área Compartida)

**3. ¿Qué componente de hardware de las computadoras actuales se ve relacionado con los tipos de áreas que se pueden definir en el entorno R-info? Justifique.**

Los tipos de áreas que hay en el entorno se pueden relacionar con la memoria. La cache se puede relacionar con un área parcialmente compartido y la memoria principal hace referencia a un área Compartida. Los registros de los procesadores hacen referencia a un área Privada.

**Programación concurrente**

**1. Describa un problema en el cual se justifique utilizar un área privada y un área parcialmente compartida.**

Una carrera de robots donde cada recorrido es un área privada y después deben depositarlo en una esquina, sumando la existencia de un robot jefe que no accede a la esquina

**2. Escriba un enunciado en el cual se justifique utilizar un área exclusiva y un área compartida.**

Mismo ejemplo que el anterior, pero sin el jefe.

**3. Escriba el enunciado de un problema en el cual se justifique utilizar áreas parcialmente compartidas.**

Dos áreas compartidas donde cada área tiene dos robots

**4. Escriba el enunciado de un problema en el cual se justifique utilizar SOLO áreas compartidas.**

Todos los robots manejándose en la misma ciudad

**Programación concurrente**

**1. Describa el mecanismo usado en r-info para que dos o más robots tomen objetos de una esquina para depositarlos en otra.**

Bloquear – Entrar – Hacer – Salir – Desbloquear (Siempre es así el orden)

**2. Describa el mecanismo usado en r-info para que un robot actúe como coordinador de dos o más robots (envío y recepción de datos**)

Para que un robot actúe como coordinador de dos o más robots primero debe enviar un mensaje a cada robot un identificador (id).