1. Las instrucciones dentro de una estructura de control For siempre se ejecutan al menos una vez.
   Rta: F, por ejemplo si el For es For i:= 10 to 1 do, ahi no se ejecutaria, por lo que no siempre se ejecutan.

2. Un programa modularizado es eficiente.
   Rta: F, que sea modularizado no significa que sea eficiente ya que dentro de los módulos se pueden usar algoritmos menos eficientes, es decir, que el tiempo que tarda en ejecutarse y la memoria que requiera sea mayor que la del programa sin modularizar. Dato a tener en cuenta: Modularizar y eficiencia son conceptos distintos por lo que uno no afecta al otro si o si.

3. En el acceso a los campos de un registro es necesario respetar el orden en que fueron declarados.
   Rta: F, el acceso a los campos de un registro es directo, por lo que se puede acceder sin necesidad de seguir un orden, solo se necesita especificar a que campo se quiere acceder.

4. Una variable global solo puede ser accedida y modificada desde el cuerpo del programa principal.
   Rta: F, las variables globales también pueden ser accedidas desde módulos, de hecho es uno de los métodos de comunicación entre módulos.

5. Para usar una variable de tipo puntero p siempre se debe usar new(p).
   Rta: se puede usar de todas maneras, por ejemplo se le puede asignar el valor nil u otra variable puntero.

6. Se pueden usar operaciones de entrada/salida sobre todos los campos de una variables de tipo registro.
   Rta: F, solo se puede si el campo admire las operaciones de entrada salida, por ejemplo si un campo es de tipo puntero no se puede leer.

7. La estructura de datos lista es heterogenea.
   Rta: F, todos los elementos son del mismo tipo.

8. Al asignar el valor de nil a un puntero se libera la memoria referenciada.
   Rta: F, no se libera la memoria referenciada, se corta el enlace. Para liberarla es con dispose().

9. Un módulo función no puede retornar un tipo de dato puntero a un arreglo.
   Rta: F, porque las funciones solo retornan tipos de datos simples y el puntero es uno de estos.

10. Si p es una variable de tipo puntero entonces no es valida la siguiente instruccion: readln(p^);
    Rta: F, es válida siempre y cuando el tipo de dato lo permita, por ejemplo un integer.

11. Si un programa usa variables globales entonces no pueden contener modulos que usen parametros.
    Rta: F, si pueden contener modulos que usen parametros, usar variables globales no impide su uso.

12. Un modulo con 5 lineas de codigo es mas eficiente en tiempos de ejecucion que un programa con 100 lineas de codigo.
    Rta: F, la cantidad de lineas no esta relacionado con la eficiencia del programa, el programa de 5 lineas podria tener mas tiempo de ejecucion.

13. Un tipo de dato registro puede ser homogeneo si todos sus campos son del mismo tipo de dato.
    Rta: F, es una estructura de datos heterogenea no importa que tenga los mismos tipos de datos en todos sus campos.

14. Nunca es posible acceder al nodo de la posicion N en una estructura de tipo lista.
    Rta: F, si es posible en caso de guardar la direccion del nodo en una variable.

15. Un programa que contiene errores semánticos se puede compilar.
    Rta: V, los errores semanticos no son evaluados por el compilador, se dan en tiempo de ejecucion.

16. Un programa solo permite variables globales cuyo tipo de dato es definido por el lenguaje.
    Rta: V, ya que las definidas por el programador se ponen en la seccion de type.

17. Un vector siempre es almacenado en memoria estatica.
    Rta: F, podemos declarar un puntero que apunte a un vector y de esa manera almacenar el vector en memoria dinamica.

18. Una variable de tipo puntero siempre se almacena en memoria dinamica.
    Rta: F, la variable de tipo puntero se puede almacenar en la memoria estatica, lo que siempre se almacena en memoria dinamica es el tipo de dato apuntado por el puntero.

19. Una funcion puede retornar un tipo de datos lista, siendo lista un puntero a un nodo.
    Rta: V, debido a que un puntero es un tipo de dato simple.

20. Antes de usar una variable puntero siempre se debe reservar en memoria.
    Rta: F, se puede asignar otro puntero a la variable o asignarle nil.

21. La comunicacion mediante parametros asegura que un programa sea correcto.
    Rta: F, que haya comunicacion mediante parametros no asegura que el programa ni pueda tener errores.

22. Siempre es posible eliminar el primer elemento es una lista.
    Rta: F, ya que la lista puede estar vacia.

23. Las instrucciones dentro de una estructura de control repeat until se pueden ejecutar 0, 1 o mas veces.
    Rta: F, la estructura de control repeat until se ejecuta siempre al menos una vez.

24. No es posible la utilizacion de las variables globales para la comunicacion entre modulos de un programa.
    Rta: F, de hecho es uno de los metodos de comunicacion entre modulos junto con el pasaje de parametros.

25. Siempre es posible realizar la eliminacion de un elemento de un vector.
    Rta: F, porque puede que el vector esté vacio y si es por posicion esta podria ser invalida.

26. Un programa modularizado puede no ser correcto.
    Rta: V, que sea modularizado no significa que no pueda tener errores.

27. El acceso a un elemento de una estructura de datos lineal solo es posible a traves de un recorrido secuencial.
    Rta: F, un vector es una estructura de datos lineal y no es necesario realizar un recorrido secuencial para acceder a un elemento.

28. En la tecnica de corrección de debugging es necesario analizar los casos limites del problema.
    Rta: F, los casos limites se analizan en la tecnica de correccion Testing.

29. Un vector siempre se usa teniendo en cuenta la dimension logica.
    Rta: F, si de antemano nos dicen que el vector va a ser usado en su totalidad no seria necesario tener en cuenta la dimension logica, ejemplo vector contador.

30. Una funcion puede devolver un tipo de dato registro, real, boolean, integer, etc.
    Rta: F, solo devuelve tipos de datos simples, y un registro es un tipo de dato compuesto.

31. Un programa que solo usa variables globales no requiere modularizacion.
    Rta: F, ambos se pueden usar al mismo tiempo, no impiden su uso entre si.