1)

a) Verdadero, en la técnica de corrección de debugging se analizan los casos límite del problema para poder detectar el error y de esta manera solucionarlo.

b) Falso, no siempre se utiliza una dimensión lógica en un vector, ya que por ejemplo en un vector contador no tendría sentido utilizarla..

c) Falso, un registro solo puede devolver un tipo de dato simple.

d) Falso, que un programa utilice solo variables globales no quiere decir que no requiera modularizacion. Al modularizar se divide en problema en partes funcionalmente independiente que encapsulan operaciones y datos y se comunican entre sí.

2)

1. Invalida, la variable c es de tipo registro y se debe cargar campo por campo.

2. Invalida, no se le puede reservar un espacio en la memoria a una variable de tipo registro.

3. Invalida, no se puede cargar un valor en la variable cli de tipo puntero, debido a que no se le asigno espacio en la memoria dinámica anteriormente.

4. Invalida, no se le puede asignar nil a una variable de tipo registro.

5. Valida, se le asigna nil a la variable cli de tipo puntero.

6. Invalida, ya que la dirección de memoria a la que apunta quedó inaccesible.

7. Invalida, ya que para acceder al campo del registro primero hay que acceder a “datos”.

8. Valida, imprime lo guardado en el campo código.

3) Un vector es un conjunto de datos indexados (ordenados por un índice), es una estructura de datos homogénea, es decir, todos sus elementos son de un mismo tipo. Es una estructura estática, esto quiere decir que la cantidad de memoria que ocupa es fija, se define al momento de su declaración y la misma no varía en la ejecución del programa. Se puede acceder a sus elementos de forma directa.

Para realizar una búsqueda en esta estructura, si no está ordenada lo primero que hay que hacer es validar la primera posición del mismo en una variable Pos. Luego con un while bicondicional ir recorriendo dicha estructura mientras la posición sea menor o igual a la dimensión lógica y el elemento en el cual estoy parado es distinto al elemento a encontrar (pos<=dimL y v[pos]<>Elem). Dentro de este while hay que ir avanzando de posición (pos:=pos+1). Por ultimo al salir del while hay que preguntar porque salió, si la última posición es menor o igual a la dimensión lógica significa que se encontró el elemento y esta función toma el valor de true. (si pos<=dimL).

4) MEMORIA ESTATICA

v: 4\*10 = 40bytes.

i: 4bytes.

sum: 4bytes.

nota: 4bytes.

apeNom: 21bytes.

40 + 4 + 4 + 4 + 21 = 73bytes.

TIEMPO DE EJECUCION

T1 = 3 \* 10 + 2 + 2\*10 = 52ut.

for = 3 \* n + 2 + cuerpo\*n

T2 = 1ut.

T3 = n+1 + 2\*n

While = n+1 + cuerpo \* n

5) Se diferencian en que la sentencia p:=nil corta el enlace entre la variable de tipo puntero p y la dirección de memoria a la que apunta. En cambio, la sentencia new(p) le reserva un espacio en la memoria a la variable de tipo puntero p.