1 - Si le paso el codigo "hhhhjmm" a un analizador sintactico que usa parsing bottom-up y tiene estas reglas, ?que devuelve?

S->hhW

S->j

W->Sm

a) 11233

b) 33211

c) 32121

d) 23131

e) 13123

f) Ninguna

g) Sintaxis incorrecta

RTA: d), hacer el parsing ascendente

2- Decir a que parte de la compilacion/ejecucion pertenecen estos mensajes y justificar por que.

a) No se reconoce el simbolo recibido RTA: Lexico (Recibo un simbolo para el cual no defini ningun TOKEN)

b) El indice del vector esta fuera de rango RTA: Ejecucion o semantico (depende si valido o no limites de vectores)

c) Indice de matriz no es del tipo entero RTA: Semantico (Busco en TS y valido si el tipo es INT)

d) El TOKEN recibido no se esperaba RTA: Sintactico (Llego a un estado donde espero un TOKEN distinto al que recibo)

e) Se esta indexando un id que no es un vector RTA: Semantico (Busco en TS si el id es vector o no)

f) Puntero apunta a una direccion de memoria prohibida RTA: Ejecucion (No hay direccionamiento de memoria durante compilacion)

g) Variable fuera de rango, dentro de una union RTA: Ejecucion (no se sabe valor de variable hasta ejecutar)

3- Polaca inversa: Se hace una compilacion con polaca inversa siguiendo las etapas Lista de Reglas -> Polaca Inversa -> Codigo ASM.

Sabiendo que la polaca inversa hace uso de una pila, es esta necesaria siempre?

Decir si es necesaria o no en cada una de las siguientes etapas:

a) Lista de Reglas -> Polaca Inversa RTA: Depende del codigo, la pila guarda los comparadores principalmente, que seran usados por la tira de polaca inversa. Puede que no haya que meter nada.

b) Polaca Inversa -> Codigo Assembler RTA: Lo mismo, si esta vacia no pasa una. Si tiene algo, seguira el algoritmo de polaca inversa y desapilara comparadores a medida que sea necesario.

c) Ejecucion RTA: Cuando el codigo ASM esta listo, la pila ya no existe. El codigo ASM tiene todo lo necesario para ejecutarse.

4- Durante la etapa de creacion del compilador:

a) ? Donde se guardan los valores de las variables de usuario y quien se encarga de administrarlas? RTA: En compilacion no se saben los valores de las variables.

b) Decir cual es la regla que dispara la generacion de codigo intermedio y explicar. RTA: La regla 0, la que detecta el start symbol. Si se llega al start y no hay mas codigo para analizar, la sintaxis es correcta y ahi comienza la generacion de codigo intermedio.

5- Considere una generacion de codigo intermedio de arbol sintactico. Quiero que cuando se este por armar un nodo que divida b/1, en vez de crearlo optimice el codigo y cree solo una hoja "b". Explicar que necesito cambiar para lograr esto.

RTA: Una manera posible seria en la accion semantica que llame a crearnodo(), detectar que si el operador es "/" y la hoja derecha es "1", en vez de continuar con la ejecucion normal llame a la funcion crearhoja() para lo que iba a ser la hoja izquierda, osea cree solo la hoja "b" en este caso.