

Sessions avaluació continuada CP1 (2011-2012)

Tema 5-6: Generació de Codi

1. Descripció de l'exercici

Donada la gramàtica d'un llenguatge LS+ que varem crear a les sessions anteriors i la seva anàlisi semàntica, volem crear el generador de codi d'aquest llenguatge LS+, per tenir tot el compilador implementat. Per fer-ho tenim el compilador del llenguatge LS que és la base del nou llenguatge LS+.

Primer farem un estudi de: Quina informació nova haurem d'afegir a la TS per poder fer la generació de codi (desplaçaments, mides... Organització de memòria) i quines instruccions de codi pseudoassembler s'han de generar i en quins punts de la gramàtica.

Alguns consells per fer el lliurament previ i el disseny de la solució:

- Feu les insercions sobre la gramàtica LS+ que tingueu sense els controls semàntics perquè es pugui llegir millor.
- Tingueu en compte que la generació de codi que ja es fa en el llenguatge LS no cal afegir-la.
- Tampoc cal afegir les instruccions en CoSel que seran necessàries per crear aquest codi.

Un cop fet aquest estudi previ es presentarà al professor i es corregirà, durant la quarta sessió tutoritzada de la setmana del 19 de desembre. Un cop presentada la solució i comentada per tots els components del grup i pel professor s'implementarà el generador de codi (el compilador complet) del llenguatge LS+ utilitzant el llenguatge CoSel i es passarà el joc de proves donat.

2. Descripció Llenguatge LS

2.1 Descripció general

- El llenguatge LS és un llenguatge imperatiu fortament tipat. Això significa que, per exemple si tenim la suma d'un enter i un real transformarem l'enter en real i farem una suma de reals. El mateix ens passarà amb totes les operacions de suma, resta, multiplicació, potència, divisió i comparacions.
- Pot tenir declaracions de tipus de dades, variables, funcions i procediments aniuats: per tant al bloc d'activació de les funcions hi ha display.
- Els tipus de dades de les expressions booleanes de if, i while han de ser enters, per tant en aquests casos els enters s'avaluaran com a valors lògics.
- ... (veure descripció de classe de teoria)

2.2 Consideracions sobre les noves característiques del llenguatge LS+:

- En general fer la generació de codi i les modificacions de la TS per totes les **NOVES** funcionalitats que heu afegit al llenguatge LS.
- Instruccions:
 - La instrucció **with** element_record..., ha de contenir un element que sigui de tipus record. Quan entrem dins seu hem de ser capaços d'accedir als camps del record i al sortir de la instrucció han de quedar modificats tal com els hem deixat dins les instruccions del with.
 - La instrucció **for**, te tres parts a la seva capçalera: inicialització, comprovació i increment. Hem de veure en quins punts del codi hem de fer aquestes accions.
- Declaracions de llistes de variables, paràmetres o camps d'un record de la forma: id,id,id: tipus: Vigilar quins són els seus desplaçaments o adreces per tal de poder accedir a ells. Hem de vigilar que els camps d'un record definits així continuïn estant en l'ordre definit.
- Mètodes: funcions i procediments de l'estil function id:tipus.f() ...
 - Els objectes que passem com a objectes d'un mètode id.f() els tractarem com si fossin paràmetres per referència i només poden rebre elements que tinguin una adreça associada, com les variables, i no podran tenir expressions de l'estil 1+2 o a+b. Hem de ser capaços de calcular el seu desplaçament per poder accedir a ells,

Sessions avaluació continuada CP1 (2011-2012)

i a l'hora de cridar-los i accedir-hi mirar com podem fer-ho per tal que al final de la funció les modificacions realitzades sobre l'objecte es mantinguin (és com un paràmetre per referència)..

- Els noms dels mètodes poden estar repetits sempre que es tracti de mètodes per objectes de diferent tipus, però hem de triar el correcte per tal de fer la crida corresponent.
- Els arrays definits ara com array [1,2,3] of array [5] of integer han de tractar-se com si fossin els antics arrays: array [1] of array [2] of array [3] of array [5] of integer, tant per reservar espai per a ells com per accedir-hi.
- Les declaracions amb inicialitzacions de variables i camps, de l'estil: **var a:integer=10; var b:array[5] of integer={1,2,3,c+d,5}; var c:r1=r1(4,"joan",{1,2,3});** poden assignar-se directament en bloc a la zona de memòria reservada per la variable corresponent. Recordeu que si hi ha més d'una variable a la definició i una sola assignació aquesta afecta a totes les variables definides a una mateixa instrucció de definició.
- Hem d'afegir els nous codis per a l'operació matemàtica de potència **.
- En general, heu d'afegir totes les actualitzacions de la TS que afectin a zones de memòria de les variables i paràmetres i a l'inici de funcions, procediments i mètodes, així com el codi associat a les diferents instruccions i funcionalitats afegides a la gramàtica LS per crear el nou llenguatge LS+.

2.3 Preguntes per reflexionar i portar contestades per escrit al primer lliurament

En general reflexionar com afecten les següents qüestions a la generació de codi i a quines regles gramaticals pot afectar.

- Definició de variables, camps de record i paràmetres com a llistes de id,id,id... : tipus: Com ens afecta a l'hora de fer la generació de codi?
- Inicialització de variables de l'estil: id:tipus={ , , , } Com fem la generació de codi?
- Inicialització de variables de l'estil: id:r1=r1(, , ,) a on r1 és un tipus de dades record. Com fem la generació de codi?
- I si en els dos casos anteriors tenim més d'un id? Id,id,id:tipus={ , , , } o id,id,id:r1=r1(, , ,). Totes les variables declarades s'han d'inicialitzar amb el valor de la dreta de l'igual.
- With exp do : Com solucionem el problema que dins del with les modificacions dels camps afectin immediatament al camp del record? Que fem amb l'adreça calculada a l'expressió del with? Que passa si tenim dos with aniuats?
- For (exp,exp,exp) : a on hem d'afegir el codi associat a cada una de les expressions?
- Com generem codi per la potència? A**b ?
- Com solucionem el tema dels mètodes associats a tipus de dades concrets, tenint en compte que els mètodes, independentment que siguin funcions o procediments, si modifiquen alguna part de l'objecte passat com a paràmetre per referència ha de quedar modificat fora del mateix mètode. És a dir els objectes es passen per referència.
- Com solucionem el tema dels mètodes a l'hora d'utilitzar-los?, com escollim la crida correcta a un objecte d'un tipus concret? Com generem la seva crida?

3. Lliuraments

Hi ha dues sessions associades a aquest enunciat:

a) Sessió de la setmana del 19 de desembre:

Per aquesta setmana s'ha de portar a la sessió el següent:

- Disseny de la Generació de Codi sobre la gramàtica del llenguatge LS+ que heu creat a les sessions anteriors.

Per exemple:

<decvar> ::= identificador : <tipus(t) @R2 [= <expressio> @R3];

Sessions avaluació continuada CP1 (2011-2012)

R2, cas1: Si estem a àmbit local: Afegir el desplaçament, modificar la mida de l'espai de variables del bloc d'activació.

R2, cas2: Si estem a àmbit global: Crear etiqueta, associar-li un espai de memòria de la mida de la variable en el segment de dades.

R3, cas1: Si estem a àmbit local: POPBVar mida(variable), desplaçament(variable).

R3, cas2: Si estem a àmbit global: POPGVar mida(variable), etiqueta(variable).

b) Sessió de la setmana del 16 de Gener:

Per aquesta setmana s'ha de portar feta la implementació del compilador del llenguatge LS+. Aquesta definició ha de complir els següents requisits:

- Adaptar-se a les característiques descrites
- Passar el joc de proves associat.

NOTA IMPORTANT: Aquesta part és OBLIGATÒRI tenir-la aprovada per poder aprovar l'assignatura.

4. Implementació en CoSel

Què es necessita per implementar l'analitzador sintàctic en CoSel:

- Descarregar-se el mòdul del generador (Com.csm), nova versió per semàntic.
- Descarregar-se el mòdul del generador (Semàntic.csm), té la definició de la Taula de símbols i funcions per crear-la i consultar-la.
- Descarregar-se el mòdul del generador (GenCod.csm), té la definició de les instruccions de codi màquina i la implementació de la màquina.
- Descarregar-se la plantilla de la implementació de la generació de codi del llenguatge LS (gencod.csl) i completar-la.
 - Substituir els valors del NIA i el Nom i cognoms per les dades dels components del grup.
("nom1","Cognoms1","0000001"),
("nom2","Cognoms2","0000002"),
("nom3","Cognoms3","0000003"),
("nom4","Cognoms4","0000004"),
("nom5","Cognoms5","0000005")
 - Completar les regles que vareu introduir al sintàctic i semàntic.
 - Passar l'autotest:
 - Modificar el fitxer gencod.csl com hem indicat abans creant un nou fitxer gencod_solucio.csl
 - Executar el cosel: cosel.cip
 - Carregar el fitxer: load gencod_solucio.csl
 - Executar autotest: autotest()

Us donem els següents fitxers:

- Enunciat: enunciatGenCod1112.pdf
- Compilador del llenguatge LS: gencod.csl
- Com.csm, semantic.csm i GenCod.csm
- Joc de proves: Estan inclosos a gencod.csl, però recordeu que vosaltres hauríeu de fer més per tal de passar el màxim nombre de test el dia del lliurament.

Nota: Els fitxers associats a CoSel (gencod.csl, gencod.csm) els tindreu disponibles a partir de **dijous dia 15 de desembre**.