

Linguaggi Formali e Compilatori
Proff. Breveglieri, Crespi Reghizzi, Morzenti
Prova scritta¹: Domanda relativa alle esercitazioni
06/03/2009

COGNOME:
NOME: Matricola:
Iscritto a: ☐ Laurea Specialistica ☐ V. O. ☐ Laurea Triennale ☐ Altro:.....
Sezione: ☐ Prof. Breveglieri ☐ Prof. Crespi ☐ Prof. Morzenti

Per la risoluzione della domanda relativa alle esercitazioni si deve utilizzare l'implementazione del compilatore **Acse** che viene fornita insieme al compito.

Si richiede di modificare la specifica dell'analizzatore lessicale da fornire a **flex**, quella dell'analizzatore sintattico da fornire a **bison** ed i file sorgenti per cui si ritengono necessarie delle modifiche in modo da estendere il compilatore **Acse** con la possibilità di gestire le operazioni di modulo ed elevamento a potenza, come nell'esempio:

```
int x,y,z;  
  
y = 3;  
z = (3 + 8) % y;  
write(z);  
  
x = z ** 5;  
write(x);  
  
y = 3 ** 4;  
  
y = z ** z;
```

Se tale programma venisse compilato e fatto girare, allora verrebbero stampati in uscita i valori "2" ($2 = 11 \text{ modulo } 3$) e "32" ($32 = 2^5$).

La soluzione deve rispettare le seguenti specifiche:

- L'operatore di modulo deve avere priorità **strettamente** inferiore a quella dell'operatore di divisione e **strettamente** superiore a quella dell'operatore di somma.
- L'operatore di elevamento a potenza deve avere priorità **inferiore** a qualsiasi altro operatore binario.

¹Tempo 45'. Libri e appunti personali possono essere consultati.
È consentito scrivere a matita. Scrivere il proprio nome sugli eventuali fogli aggiuntivi.

- Lo studente specifichi l'associatività che ritiene più opportuna per ciascun operatore.

Una **soluzione ottimale** non deve generare codice assembly per un programma **Lance** per operazioni di modulo ed elevamento a potenza aventi *solo costanti* come operandi, ma usare direttamente il risultato dell'operazione.

Si espliciti ogni eventuale ulteriore assunzione che sia ritenuta necessaria a completare la specifica data.

Si ricorda che l'operatore modulo restituisce il resto della divisione intera. In altre parole, se x e y sono due numeri interi, allora x modulo y è uguale a: $x - y * d$, dove $d = x/y$ è il risultato della divisione *intera*.

Si ricorda che gli operandi possono essere sia costanti che variabili. Può tornare utile la funzione:

```
int gen_load_immediate(t_program_infos *program, int immediate);
```

1. Definire i token (e le relative dichiarazioni in **Acse.lex** e **Acse.y**) necessari per ottenere le funzionalità richieste. (3 punti)
2. Definire le regole sintattiche (o le modifiche a quelle esistenti) necessarie per ottenere le funzionalità richieste (per entrambi gli operatori). (8 punti)

3. Definire le azioni semantiche (o le modifiche a quelle esistenti) necessarie per ottenere *tutte* le funzionalità richieste per *uno* dei due operatori, a scelta dello studente. (13 punti per l'operatore modulo; 19 punti per l'operatore di elevamento a potenza)

4. (Bonus) Dato il seguente codice:

```
int a = 10;  
int b = 20;  
  
a = a ** b % c ;
```

scrivere l'albero sintattico partendo dalla grammatica Bison definita in Acse.y considerando le modifiche introdotte nei punti precedenti, *iniziando dal non-terminale statements*.