Linguaggi Formali e Compilatori
Proff. Breveglieri, Crespi Reghizzi, Morzenti
Prova scritta [1]: Domanda relativa alle esercitazioni
05/01/2013

COGNOME: .....
NOME: ..... Matricola: .....
Corso: ○ 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 l'operatore di assegnamento tra vettori `:=`. Di seguito, si riporta un esempio di codice che utilizza l'operatore.

```
1 int v1[10];
2 int v2[12];
3
4 v1 := v2;
```

Figura 1: Esempio di utilizzo dell' operatore

Dati un vettore `v1` di dimensione `N1` ed un vettore `v2` di dimensione `N2`, l'operazione `v1 := v2` copia gli elementi del vettore `v2` nel vettore `v1`, i.e., `v1[i] = v2[i]` con `i` nel range `[0, min(N1,N2)]`. In altre parole, se i vettori hanno dimensione differente, il range di valori considerati è definito tra 0 e la dimensione minima tra i due vettori. Inoltre, quando la dimensione del vettore di destinazione è maggiore della dimensione del vettore sorgente, l'operazione di assegnamento `v1 := v2` introduce un valore di riempimento predefinito (il valore è scelto da chi progetta l'implementazione dell'operatore; un valore possibile è 0).

Ad esempio, sia `v2` = {`1,2,3,4,5`} un vettore di 5 elementi e `v1` un vettore di 7 elementi, l'operazione `v1 := v2` definisce `v1` come `v1` = {`1,2,3,4,5,0,0`}. Sia, invece, `v2` = {`1,2,3,4,5,6,7`} un vettore di 7 elementi e `v1` un vettore di 5 elementi, l'operazione `v1 := v2` definisce `v1` come `v1` = {`1,2,3,4,5`}.

Un errore di compilazione viene generato quando gli operandi dell'operatore `:=` non sono vettori.

[1] Tempo 60'. Libri e appunti personali possono essere consultati.
È consentito scrivere a matita. Scrivere il proprio nome sugli eventuali fogli aggiuntivi.

1. Definire i token (e le relative dichiarazioni in `Acse.lex` e `Acse.y`) necessari per ottenere la funzionalità richiesta. (3 punti)

2. Definire le regole sintattiche (o le modifiche a quelle esistenti) necessarie per ottenere la funzionalità richiesta. (4 punti)

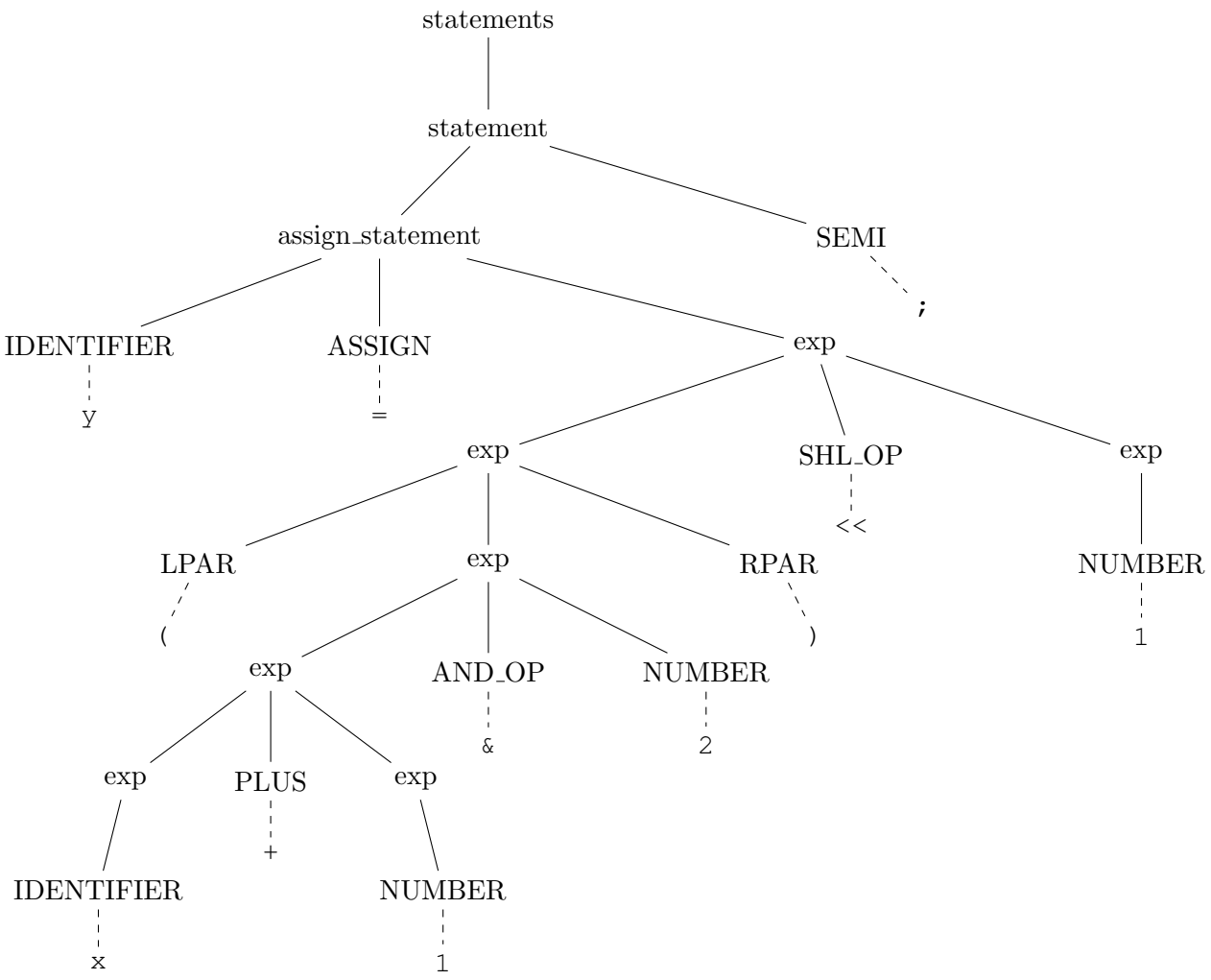
3. Definire le azioni semantiche (o le modifiche a quelle esistenti) necessarie per ottenere la funzionalità richiesta. (18 punti)

   La soluzione è riportata nella patch allegata.

4. Data il seguente snippet di codice `Lance`:

```
1 y = (x + 1 & 2) << 1;
```

Scrivere l'albero sintattico relativo partendo dalla grammatica Bison definita in `Acse.y` *iniziando dal non-terminale* `statements`. (5 punti)



5. (**Bonus**) Descrivere come modificare la soluzione proposta al punto 3. in modo che si possa specificare un range di elementi del vettore sorgente da copiare nel vettore destinazione. In questo caso, il vettore sorgente è affiancato da un range composto da due valori interi a, b definito mediante il costrutto `<a,b>`, in cui `a` è il valore minimo del range e `b` il valore massimo. Ad esempio, l'operazione `v1 := v2<1,4>` copia in `v1` (partendo dalla posizione 0) gli elementi `v2[1]`, `v2[2]`, `v2[3]`, `v2[4]`.