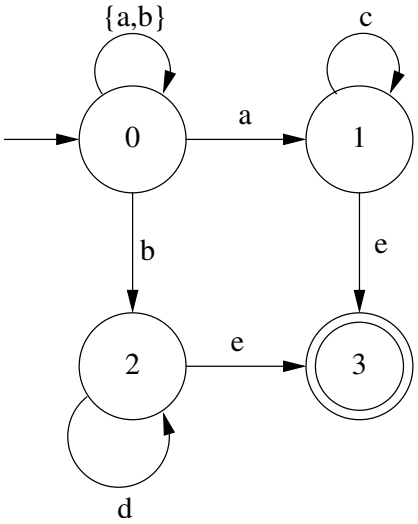
Programmazione A. A. 2004/2005

II° Appello del 10/01/2005

ISTRUZIONI: Scrivere **in stampatello** COGNOME e NOME su ogni foglio. Vanno consegnati **tutti** i fogli: brutta copia e testo compresi. Coloro che non vogliono consegnare possono andarsene, consegnando il testo, dopo un'ora dall'inizio del compito ed entro 15 minuti dalla scadenza del tempo.

ESERCIZIO 1 (5 punti)

Dato il seguente automa:



Descrivere formalmente il linguaggio accettato e disegnare un automa deterministico equivalente.

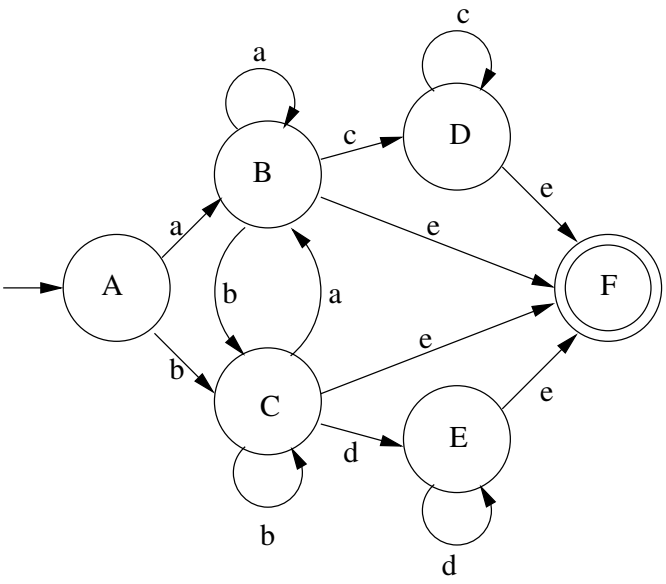
SOLUZIONE

Usando una espressione regolare il linguaggio accettato dall'automa è il seguente: $(a|b)^*\, a\, c^*\, e \;\mid\; (a|b)^*\, b\, d^* e$. Usando espressioni su insiemi lo stesso linguaggio si può scrivere $\{s\, a\, c^n\, e \mid n \geq 0, s \in \{a,b\}^*\} \cup \{s\, b\, d^n\, e \mid n \geq 0, s \in \{a,b\}^*\}$.

Utilizziamo l'algoritmo della costruzione dei sottoinsiemi per ottenere un automa deterministico equivalente. La tabella che risulta dall'applicazione dell'algoritmo è la seguente:

	a	b	c	d	e
$A = \{0\}$	B	C			
$B = \{0,1\}$	B	C	D		F
$C = \{0,2\}$	B	C		E	F
$D = \{1\}$			D		F
$E = \{2\}$				E	F
$F = \{3\}$					

L'automa deterministico risultante è il seguente:



ESERCIZIO 2 (4 punti)

Si scriva una grammatica libera dal contesto che generi tutte e sole le stringhe del seguente linguaggio:

$$L = \{a^n\, a\, b^k\, c^{n+1} \mid n \geq 0, k > 0\}$$

SOLUZIONE

$$\begin{array}{lcl} S & \rightarrow & aSc \mid aBc \\ B & \rightarrow & bB \mid b \end{array}$$

ESERCIZIO 3 (8 punti)

Si supponga di estendere la sintassi del linguaggio didattico visto a lezione come segue:

```
Com ::= IdeList = Exp;
IdeList ::= Ide = Ide | Ide = IdeList
```

L'esecuzione di un comando `x1 = x2 = x3 = ... xk = E;`, consiste nell'assegnare il valore dell'espressione `E` a tutti gli identificatori della lista.

Definire le regole di semantica operazionale per il nuovo comando con riferimento al modello in cui lo stato è composto da una pila di frame. (Suggerimento: basarsi sulla struttura delle produzioni di `IdeList` per scrivere le regole di $\rightarrow_{com}$ per il nuovo comando).

SOLUZIONE

$$com_{==} \qquad \frac{\langle \mathtt{E}, \sigma\rangle \rightarrow_{exp} \underline{\mathtt{v}} \quad \sigma' = \sigma[\underline{\mathtt{v}}/\mathtt{x1}, \underline{\mathtt{v}}/\mathtt{x2}]}{\langle \mathtt{x1 = x2 = E;}, \sigma\rangle \rightarrow_{com} \sigma'}$$

$$com_{==list} \qquad \frac{\langle \mathtt{E}, \sigma\rangle \rightarrow_{exp} \underline{\mathtt{v}} \quad \mathtt{Il} \in \mathtt{IdeList} \quad \langle \mathtt{Il = E;}, \sigma\rangle \rightarrow_{com} \sigma' \quad \sigma'' = \sigma'[\underline{\mathtt{v}}/\mathtt{x1}]}{\langle \mathtt{x1 = Il = E;}, \sigma\rangle \rightarrow_{com} \sigma''}$$

ESERCIZIO 4 (6 punti)

Si scriva l'implementazione del seguente metodo:

```
/** Controlla se gli elementi di un sottoarray sono ordinati
    @param a l'array di interi da controllare
    @param start la posizione del primo elemento del sottoarray
    @param stop la posizione dell'ultimo elemento del sottoarray
    @return true se gli elementi dell'array dalla posizione start
            alla posizione stop sono ordinati, false in caso
            contrario o nel caso che start e/o stop siano fuori
            dai limiti dell'array
  */
public boolean m(int a[], int start, int stop)
```

SOLUZIONE

```
public boolean m(int a[], int start, int stop){
  if (start >= a.length || start < 0 || stop >= a.length)
    return false;
  // adatto lo schema di ricerca lineare incerta per trovare
  // il primo elemento non ordinato
  int i = start;
  boolean trovato = false;
  while (i < stop && !trovato) // non si controlla l'elemento in stop
    if (a[i] > a[i+1])
      trovato = true;
    else i++;
  // Se trovato allora il sottoarray non e' ordinato
  if (trovato)
    return false;
  else return true;
}
```

ESERCIZIO 5 - PER GLI STUDENTI DA 5 CFU (7 punti)

Si dimostri formalmente che i comandi `C1: x = y - 1;` `C2: while (x > 10) x = y - 1;` sono equivalenti a partire da uno stato σ tale che $\sigma(\mathtt{x}) = \underline{\mathbf{15}}$ e $\sigma(\mathtt{y}) = \underline{\mathbf{11}}$.

SOLUZIONE

Dimostriamo, facendo due derivazioni nel sistema $\rightarrow_{com}$, che i due comandi terminano nello stesso stato se eseguiti a partire dallo stesso stato σ su cui facciamo le ipotesi date nel testo. Iniziamo dal comando `C1`:

$\langle \mathtt{x = y - 1;}, \sigma \rangle$
$\rightarrow_{com} \{(com_{=}) :$
$\quad \mathcal{E}[\![\mathtt{y - 1}]\!]_{\sigma} = \underline{\mathbf{10}}$
$\quad \sigma' = \sigma[\underline{\mathbf{10}}/\mathtt{x}]\}$
σ'

Vediamo ora il comando C2:

$\langle \texttt{while (x > 10) x = y - 1;}, \sigma \rangle$
$\rightarrow_{com} \{ \quad (com_{while-tt}) :$
$\mathcal{E}[\![\texttt{x > 10}]\!]_\sigma = \underline{\mathbf{tt}}$
$(com_{=}) : \mathcal{E}[\![\texttt{y - 1}]\!]_\sigma = \underline{\mathbf{10}}$
$\langle \texttt{x = y - 1;}, \sigma \rangle \rightarrow_{com} \sigma[\underline{\mathbf{10}}/\texttt{x}] = \sigma'$
$(d1) : \langle \texttt{while (x > 10) x = y - 1;}, \sigma' \rangle \rightarrow_{com} \sigma' \}$
σ'

dove $(d1)$ è la seguente:

$\langle \texttt{while (x > 10) x = y - 1;}, \sigma' \rangle$
$\rightarrow_{com} \{ \quad (com_{while-ff}) :$
$\mathcal{E}[\![\texttt{x > 10}]\!]_\sigma = \underline{\mathbf{ff}} \}$
σ'

Si ha che entrambi i comandi terminano nello stesso stato σ'. Pertanto i due comandi sono equivalenti date le ipotesi del testo.

ESERCIZIO 5 - PER GLI STUDENTI DA 6 CFU (7 punti)

Si consideri il seguente programma nel linguaggio didattico:

```
prog {
  class Foo {
    public int x;
    public int y;
    public void m(int x, int y) {
      int z = x + y;
      y = y + 5;
      if (this.x > x) this.y = this.y * y;
      this.x = this.x + z;
    }
  }
 { Foo o1 = new Foo;
   o1.x = 30; o1.y = 5;
   Foo o2 = new Foo;
   int y = o1.y * 3;
   o2.x = o1.x + y;
   o2.y = o2.x - y * 2;
   o2.m(o1.x, o2.y + o1.y); (1)
 }
}
```

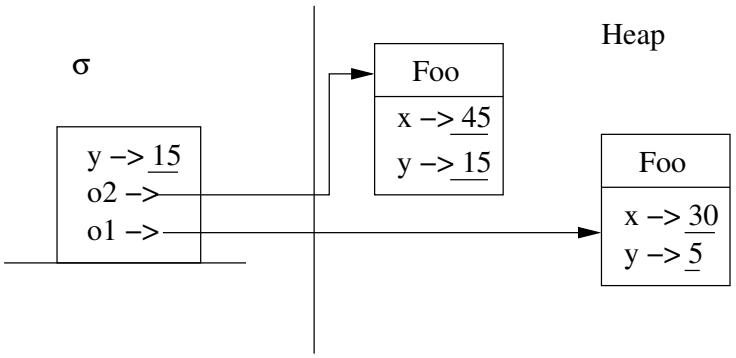
Si disegni lo stato (pila di frame dell'attivazione corrente e heap) nei seguenti punti di esecuzione:

1. immediatamente prima della chiamata al metodo del punto (1)
2. nel metodo, prima di eseguire la prima istruzione

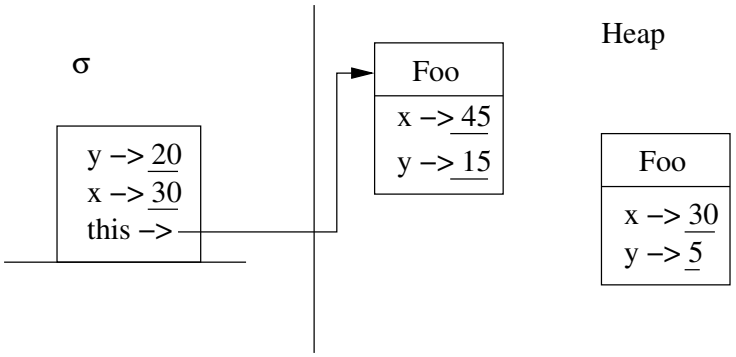
3. nel metodo, immediatamente dopo l'esecuzione dell'ultima istruzione
4. dopo la chiamata del metodo

SOLUZIONE

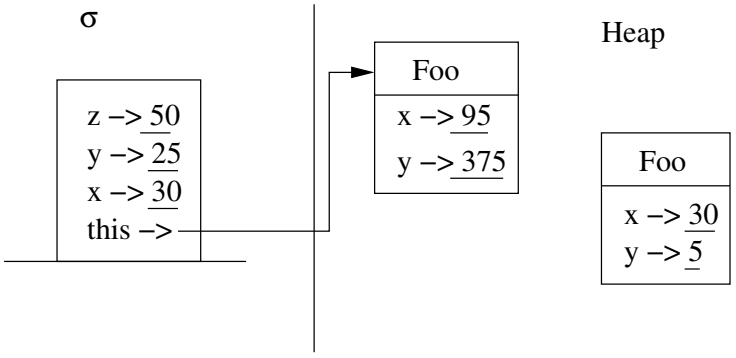
Punto 1:



Punto 2:



Punto 3:



Punto 4: