LOGICA PER LA PROGRAMMAZIONE (A,B) - a.a. 2010-2011
SOLUZIONI PROPOSTE
QUINTO APPELLO - 20/07/2011

ESERCIZIO 1
Si provi che la seguente proposizione è una tautologia:

$$(Q \wedge S) \vee ((P \Rightarrow Q) \wedge \neg Q) \Rightarrow (P \Rightarrow S)$$

Suggerimento: si usi una dimostrazione per casi su Q.
N.B. Ovviamente trattandosi di un suggerimento sono ammesse anche soluzioni diverse, purché corrette.

Soluzione
Come suggerito, usiamo una dimostrazione per casi su Q:

Ipotesi: Q
La proposizione diventa:

$(T \wedge S) \vee ((P \Rightarrow T) \wedge \neg T) \Rightarrow (P \Rightarrow S)$
$\equiv$ { Unità, T:F}
$S \vee ((P \Rightarrow T) \wedge F) \Rightarrow (P \Rightarrow S)$
$\equiv$ { Zero }
$S \vee F \Rightarrow (P \Rightarrow S)$
$\equiv$ { Unità, Elim-$\Rightarrow$ due volte }
$\neg S \vee (\neg P \vee S)$
$\equiv$ { Terzo escluso, Zero }
T

Ipotesi: $\neg Q$
La proposizione diventa:

$(F \wedge S) \vee ((P \Rightarrow F) \wedge \neg F) \Rightarrow (P \Rightarrow S)$
$\equiv$ { Zero, Elim-$\Rightarrow$, T:F }
$F \vee ((\neg P \vee F) \wedge T) \Rightarrow (P \Rightarrow S)$
$\equiv$ { Unità }
$(\neg P \wedge T) \Rightarrow (P \Rightarrow S)$
$\equiv$ { Zero }
$\neg P \Rightarrow (P \Rightarrow S)$
$\equiv$ { Elim-$\Rightarrow$, due volte }
$P \vee (\neg P \vee S)$
$\equiv$ { Terzo escluso, Zero }
T

ESERCIZIO 2
Si provi che la seguente formula è valida (P, Q, R e S contengono la variabile libera x)

$$(\exists x. S \wedge \neg Q) \wedge (\forall x. R \wedge (P \Rightarrow Q)) \Rightarrow (\exists x. \neg(S \Rightarrow P))$$

Soluzione Partiamo dalla premessa, indicando con d un generico elemento del dominio che testimonia $(\exists x. S \wedge \neg Q)$ (quindi usiamo la Regola di Skolemizzazione):

$S[d/x] \wedge \neg Q[d/x] \wedge (\forall x. R \wedge (P \Rightarrow Q))$
$\Rightarrow$ { Elim-$\forall$ }
$S[d/x] \wedge \neg Q[d/x] \wedge R[d/x] \wedge (P[d/x] \Rightarrow Q[d/x])$
$\equiv$ { Contropositiva }
$S[d/x] \wedge \neg Q[d/x] \wedge R[d/x] \wedge (\neg Q[d/x] \Rightarrow \neg P[d/x])$
$\Rightarrow$ { Modus Ponens }
$S[d/x] \wedge R[d/x] \wedge \neg P[d/x]$
$\Rightarrow$ { Simpl-$\wedge$ }
$S[d/x] \wedge \neg P[d/x]$
$\equiv$ { Doppia negazione, De Morgan }
$\neg(\neg S[d/x] \vee P[d/x])$
$\equiv$ { Intro-$\Rightarrow$ }
$\neg(S[d/x] \Rightarrow P[d/x])$
$\Rightarrow$ { Intro-$\exists$ }
$(\exists x. \neg(S \Rightarrow P)$

ESERCIZIO 3
Utilizzando il calcolo del primo ordine si formalizzi il seguente enunciato dichiarativo, indicando esplicitamente l'interpretazione intesa, *in tutte le sue componenti*:

"Alcuni studenti apprezzano tutti i professori, altri no"

Soluzione

- **Linguaggio**
  - $\mathbf{C} = \emptyset$
  - $\mathbf{F} = \emptyset$
  - $\mathbf{P} = \{studente(\cdot), professore(\cdot), apprezza(\cdot,\cdot)\}$
- **Interpretazione**: $\mathbf{I} = (\mathbf{D}, \alpha)$
  - $\mathbf{D} = \mathcal{P}$, con $\mathcal{P}$ insieme delle persone.
  - $\alpha(studente)(p) \equiv \mathbf{T}$ se e solo la persona p è uno studente
  - $\alpha(professore)(p) \equiv \mathbf{T}$ se e solo la persona p è un professore
  - $\alpha(apprezza)(p, q) \equiv \mathbf{T}$ se e solo se la persona p apprezza la persona q

Una formalizzazione è allora

$$(\exists p.studente(p) \wedge (\forall q.professore(q) \Rightarrow apprezza(p, q))) \wedge (\exists p.studente(p) \wedge (\exists q.professore(q) \wedge \neg apprezza(p, q)))$$

ESERCIZIO 4

Assumendo **a : array [0, n) of nat** e **b : array [0, m) of nat** si formalizzi il seguente enunciato:

"Nell'array **a** ci sono almeno tre elementi distinti che non compaiono in **b**"

Soluzione

$$\#\{x : x \in \mathbb{N} \wedge (\exists i.i \in [0, n) \wedge a[i] = x) \wedge (\forall j.j \in [0, m) \Rightarrow \neg(b[j] = x))\} > 3$$

ESERCIZIO 5
Si determini una espressione E in modo che la seguente tripla sia verificata e si dimostri formalmente la correttezza della soluzione proposta.

$\{x = A \wedge y = B \wedge z = C\}$
  if $y = z$ **then** $x := y$; **else** $x := E$; **fi**
$\{x = z\}$

Soluzione Da una semplice analisi del comando si vede facilmente che una soluzione corretta è $E = z$. L'esercizio si riduce quindi a una semplice applicazione della Regola del Condizionale.
Attenzione: La soluzione $E = C$ con non è corretta, perché C è una variable di specifica e non può comparire in un comando.

ESERCIZIO 6
Si consideri il seguente programma annotato, dove '$*$' è l'operatore di moltiplicazione e A, B sono variabili di specifica.

$\{x = A \wedge y = B \wedge A \geq 0\}$
$\{\mathsf{Inv} : y = B + 2 * (A - x) \wedge x \geq 0\}\{\mathrm{t} : x\}$
while $x \neq 0$ **do**
    $x, y := x - 1, y + 2;$
endw
$\{y = B + 2 * A\}$

1. Scrivere e dimostrare la proprietà di invarianza.
2. Dimostrare che, nello stato finale raggiunto dal ciclo, vale la post-condizione del programma annotato.

Soluzione

1. La proprietà di invarianza è la tripla

   $\{y = B + 2 * (A - x) \wedge x \geq 0 \wedge x \neq 0\}$
       $x, y := x - 1, y + 2$
   $\{y = B + 2 * (A - x) \wedge x \geq 0 \wedge def(x \neq 0)\}$

   che si risolve con una semplice applicazione della Regola per l'Assegnamento.

2. La dimostrazione consiste nel mostrare che l'asserzione che risulta valida alla fine dell'esecuzione del **while**, cioè $INV \wedge \neg E$, implica la post-condizione del programma, cioè $\{y = B + 2 * A\}$, quindi occorre dimostrare:

   $$(y = B + 2 * (A - x) \wedge x \geq 0 \wedge x = 0) \Rightarrow (y = B + 2 * A)$$

   Mostriamo che vale l'uguaglianza della conseguenza usando le premesse come ipotesi:

   y
   $=$ $\{$**Ip:** $y = B + 2 * (A - x)\}$
   $B + 2 * (A - x)$
   $=$ $\{$**Ip:** $x = 0\}$
   $B + 2 * A$