Σειρά Προβλημάτων 4 – Λύσεις

Άσκηση 1

Να αποφασίσετε κατά πόσο οι πιο κάτω προδιαγραφές είναι ορθές σύμφωνα με την έννοια της μερικής ορθότητας και την έννοια της ολικής ορθότητας. Να αιτιολογήσετε σύντομα τις απαντήσεις σας (δεν είναι απαραίτητο να κατασκευάσετε αποδείξεις με τους κανόνες). Η εντολή skip είναι μια εντολή η οποία δεν εκτελεί τίποτα και ικανοποιεί τον κανόνα {φ} `skip` {φ}.

(α) $\{x = 12 \wedge y = 7\}$ `skip` $\{z = 12\}$

(β) $\{x > 6 \wedge y > 3\}$ `if x > y x := y; else y := x` $\{3 < x < 6\}$

(γ) $\{x \geq 28 \wedge y > 1\}$ `while (x != y) x := x+1` $\{x = y\}$

Λύση:

(α) Η προδιαγραφή δεν είναι ορθή ούτε με την έννοια της μερικής ούτε με την έννοια της ολικής ορθότητας: Η προσυνθήκη δεν προσδιορίζει την αρχική τιμή της μεταβλητής z και επομένως είναι αδύνατο να εγγυηθούμε ότι μετά την εκτέλεση του προγράμματος η τιμή της z θα είναι ίση με 12.

(β) Η προδιαγραφή δεν είναι ορθή ούτε με την έννοια της μερικής ούτε με την έννοια της ολικής ορθότητας. Για παράδειγμα, θεωρήστε το στιγμιότυπο όπου x = 7 και y = 8. Το στιγμιότυπο αυτό είναι συμβατό με την προσυνθήκη της προδιαγραφής. Εντούτοις, εκτέλεση της εντολής if δεν θα αλλάξει την τιμή της μεταβλητής x η οποία θα παραμείνει ίση με 7 που δεν ικανοποιεί τη μετασυνθήκη του προγράμματος.

(γ) Παρατηρούμε ότι το πρόγραμμα δεν τερματίζει. Επομένως η πρόταση είναι αληθής με την έννοια της μερικής ορθότητας αλλά όχι με την έννοια της ολικής ορθότητας.

Άσκηση 2 (30 μονάδες)

Να αποδείξετε την ορθότητα των πιο κάτω προδιαγραφών (ολική ορθότητα).

(α) $\models_{tot} \{x \geq 0\}$ **P** $\{y = x!\}$ **όπου ο κώδικας του P δίνεται πιο κάτω.**

```
a := x;
y := 1;
while (a > 0){
  y := y * a;
  a := a - 1;
}
```

(β) $\models_{tot} \{ n \geq 1 \}$ ***C*** $\{ m=p \cdot q,\ p = \max \{ A[i] \mid 0 \leq i < n \},\ q = \min \{ A[i] \mid 0 \leq i < n \} \}$ **όπου ο κώδικας του *C* δίνεται πιο κάτω.**

```
p := A[0];
q := A[0];
i := 0;
while (i < n){
  if  (A[i] > p)
        p := A[i];
  else
        if (A[i] < q)
              q := A[i];
        else
              skip;
  i++;
}
m := p*q;
```

Λύση:

(α) Η αμετάβλητη συνθήκη είναι η 'y = x! / a! ∧ 0 ≤ a' και η μεταβλητή έκφραση είναι η a.

{x ≥ 0}
{1 = x!/x! ∧ 0 ≤ x} Ενδυνάμωση συνθήκης
`a := x;`
{1 = x!/a! ∧ 0 ≤ a} Κανόνας ανάθεσης
`y := 1;`
{y = x!/a! ∧ 0 ≤ a} Κανόνας ανάθεσης
`while (a>0) {`
{y = x!/a! ∧ 0 ≤ a ∧ a>0 ∧ 0 ≤ a = E_0} Αμετ. συνθήκη και φρουρός
{y*a = x!/(a-1)! ∧ 0 ≤ a - 1 ∧ 0 ≤ a-1 < E_0} Κανόνας συνεπαγωγής
`y := y*a;`
{y = x!/(a-1)! ∧ 0 ≤ a - 1 ∧ 0 ≤ a-1 < E_0} Κανόνας ανάθεσης
`a := a – 1;`
{y = x!/a! ∧ 0 ≤ a ∧ 0 ≤ a < E_0} Κανόνας ανάθεσης
`}`
{y = x!/a! ∧ 0 ≤ a ∧ a ≤ 0} Κανόνας while
{y=x!}

(β) Για σκοπούς απλούστευσης της λύσης η τρίτη εντολή του προγράμματος έχει γραφτεί ως 'i:= 1'. Σε αυτή την περίπτωση η αμετάβλητη συνθήκη και η μεταβλητή έκφραση είναι:

η = ' p = max { A[k] | 0 ≤ k < i } ∧ q = min { A[m] | 0 ≤ k <i} ∧ i≤n'

E = 'n-i'

Στην περίπτωση του ψευδοκώδικα όπως δίνεται στην εκφώνηση, η μόνη διαφορά θα ήταν ότι η αμετάβλητη συνθήκη θα ήταν:

η = ' p = max [{A[0]} ∪{ A[k] | 0 ≤ k < i }] ∧ q = min [{A[0]} ∪{ A[m] | 0 ≤ k <i}] ∧ i≤n'

Ακολουθεί η απόδειξη της προδιαγραφής.

{ n ≥ 1 }
{ A[0] = max { A[k] | 0 ≤ k <1 } , A[0] = min { A[m] | 0 ≤ k < 1} ∧ 1≤n ∧ 0≤n-1} Συνεπαγωγή
`p := A[0];`
{ *p* = max { A[k] | 0 ≤ k < 1 } , A[0] = min { A[m] | 0 ≤ k < 1} ∧ 1≤n ∧ 0≤n-1} Αξ. Ανάθεσης
`q := A[0];`
{ *p* = max { A[k] | 0 ≤ k ≤ 1 } , *q* = min { A[m] | 0 ≤ k ≤ 1} ∧ 1≤n ∧ 1≤n} Αξ. Ανάθεσης
`i := 1;`
{ *p* = max { A[k] | 0 ≤ k ≤ i } , *q* = min { A[k] | 0 ≤ k < i} ∧ i≤n ∧ 0≤n-i} Αξ. Ανάθεσης
`while (i < n){`
{ *p* = max { A[m] | 0 ≤ k < i } , *q* = min { A[m] | 0 ≤ k < i} ∧ i≤n ∧i <n ∧ 0≤n-i=E_0} Αμ. Συνθ.+ Φρ. + Μετ. Εκφρ.
{A[i]>p→A[i]=max{ A[k]|0≤k<i+1},*q*=min{A[k]|0≤k<i+1}∧i+1≤n∧0≤n-i-1<E_0
∧ A[i]≤p→ [*p*=max{A[k]|0≤k<i+1}∧i+1≤n∧0≤n-i-1<E_0
∧ A[i]<q → A[i]=min{A[k]|0≤k<i+1}∧ A[i]≥q → *q*=min{A[k]|0≤k<i+1}}] Συνεπαγωγή
`if (A[i] > p)`
{ A[i]=max{ A[k]|0≤k<i+1}, *q* = min{A[k] | 0≤m<i+1} ∧ i+1≤n ∧ 0≤n-i-1<E_0 }
`p := A[i];`
{ *p* = max{ A[k] | 0≤k<i+1} , *q* = min{A[k] | 0≤m<i+1} ∧ i+1≤n ∧ 0≤n-i-1<E_0 } Αξ. Ανάθεσης

```
        else
```
{p=max{A[k]|0≤k<i+1}∧i+1≤n∧0≤n-i-1<E_0

∧A[i]<q → A[i]=min{A[k]|0≤k<i+1}∧ A[i]≥q→ q=min{A[k]|0≤k<i+1}}

{A[i]<q →p=max{A[k]|0≤k<i+1},A[i]=min{A[k]|0≤k<i+1}∧i+1≤n∧0≤n-i-1<E_0 Συνεπαγωγή

∧

A[i]≥q→p=max{A[k]|0≤k<i+1},q=min{A[k]|0≤k<i+1} ∧ i+1≤n ∧ 0≤n-i-1<E_0}

```
        if (A[i] < q)
```
{ p=max{ A[k] | 0≤k<i+1} , A[i]=min{A[k] | 0≤k<i+1} ∧ i+1≤n ∧ 0≤n-i-1<E_0 }

```
            q := A[i];
```
{ p = max{ A[k] | 0≤k<i+1} , q = min{A[k] | 0≤k<i+1} ∧ i+1≤n ∧ 0≤n-i-1<E_0 } Αξ. Ανάθεσης

```
        else
```
{ p = max{ A[k] | 0≤k<i+1} , q = min{A[k] | 0≤k<i+1} ∧ i+1≤n ∧ 0≤n-i-1<E_0 }

```
            skip;
```
{ p = max{ A[k] | 0≤k<i+1} , q = min{A[k] | 0≤k<i+1} ∧ i+1≤n ∧ 0≤n-i-1<E_0 } Καν. if

{ p = max{ A[k] | 0≤k<i+1} , q = min{A[k] | 0≤k<i+1} ∧ i+1≤n ∧ 0≤n-i-1<E_0 } Καν. if

```
        i++;
```
{ p = max { A[k] | 0 ≤ k < i } , q = min { A[k] | 0 ≤ k < i} ∧ i≤n ∧ 0≤n-i < E_0 } Αξ. Ανάθεσης

```
}
```
{ p = max { A[k] | 0 ≤ k < i } , q = min { A[k] | 0 ≤ k < i} ∧ i≤n ∧ i≥n } Καν. total while

{ p·q =p·q, p = max { A[i] | 0 ≤ i < n } , q = min { A[i] | 0 ≤ i < n} } Συνεπαγωγή

```
m := p*q;
```
{ m=p·q, p = max { A[i] | 0 ≤ i < n } , q = min { A[i] | 0 ≤ i < n} } Αξ. Ανάθεσης

<u>Άσκηση 3 (11 μονάδες)</u>

Θέλουμε να προσθέσουμε στη γλώσσα WHILE (διαφάνεια 9-5), εντολές της μορφής:

case B of {1:C_1; 2:C_2; …; n: C_n}

Οι εντολές αυτές εκτελούνται ως εξής:

(1) Πρώτα υπολογίζεται η έκφραση B για να δώσει μια τιμή x.

(2) Αν το $x = i$, $1 \le i \le n$, τότε εκτελείται η εντολή C_i.

(3) Διαφορετικά, η εντολή δεν εκτελεί καμιά ενέργεια και η ροή του προγράμματος προχωρεί στην επόμενη εντολή (αν υπάρχει).

(α) Θεωρήστε τον πιο κάτω κανόνα για την εντολή.

$$\frac{\{\varphi \wedge B = 1\}C_1\{\psi\} \dots \{\varphi \wedge B = n\}C_n\{\psi\}}{\{\varphi\}\ case\ B\ of\ \{1: C_1; \dots; n: C_n\}\{\psi\}}$$

Να εξηγήσετε γιατί ο κανόνας αυτός είναι λανθασμένος.

(β) Να προτείνετε διορθωμένη εκδοχή του πιο κανόνα από το μέρος (α) και να την χρησιμοποιήσετε για να αποδείξετε την ορθότητα της πιο κάτω προδιαγραφής.

{$1 \le x \wedge x \le 3$} case x of {1: y:= x – 1; 2: y:= x – 2; 3: y:= x – 3} {y = 0}

(α) Ο κανόνας αυτός είναι λανθασμένος γιατί αγνοεί την περίπτωση που η τιμή του B δεν ανήκει στο πεδίο {1,…,n}. Για να ικανοποιείται η προδιαγραφή από τον συγκεκριμένο τύπο

εντολής ο κανόνας θα πρέπει να λαμβάνει υπόψη και αυτή την επιπρόσθετη περίπτωση, όπως διατυπώνεται στην πιο κάτω διορθωμένη μορφή του κανόνα.

$$\frac{\{\varphi \wedge B = 1\}C_1\{\psi\}, \dots, \{\varphi \wedge B = n\}C_n\{\psi\}, [\,\varphi \wedge (B \notin \{1, \dots, n\})] \rightarrow \psi}{\{\varphi\}\ case\ B\ of\ \{1\colon C_1; \dots; n\colon C_n\}\{\psi\}}$$

(β) Για να αποδείξουμε την προδιαγραφή

$\{1 \le x \wedge x \le 3\}$ case x of {1: y:= $x - 1$; 2: y:= $x - 2$; 3: y:= $x - 3$} $\{y = 0\}$

Πρέπει να αποδείξουμε τις επιμέρους προδιαγραφές

1. $\{1 \le x \wedge x \le 3 \wedge x = 1\}\ y := x - 1\ \{y = 0\}$
2. $\{1 \le x \wedge x \le 3 \wedge x = 2\}\ y := x - 2\ \{y = 0\}$
3. $\{1 \le x \wedge x \le 3 \wedge x = 3\}\ y := x - 3\ \{y = 0\}$
4. $(1 \le x \wedge x \le 3 \wedge x \notin \{1,2,3\}) \Rightarrow y = 0$

Οι τρεις πρώτες επιβεβαιώνονται εύκολα μέσω του κανόνα της ανάθεσης και το κανόνα της ενδυνάμωσης προσυνθήκης.

Όσον αφορά το τέταρτο σκέλος της προδιαγραφής, είναι εύκολο να δούμε ότι $1 \le x \wedge x \le 3 \wedge x \notin \{1,2,3\} \equiv$ False και επομένως False $\Rightarrow y = 0$.

<u>Άσκηση 4 (14 μονάδες)</u>

Θεωρήστε το μοντέλο Kriple M=(W, R, L), όπου

W = {a,b,c,d,e}
R = {(a,c), (a,e),(b,a),(b,c),(d,e),(e,a)}, και
L(a) = {p}, L(b)={p,q}, L(c)={p,q}, L(d) = {q} και L(e) = {}.

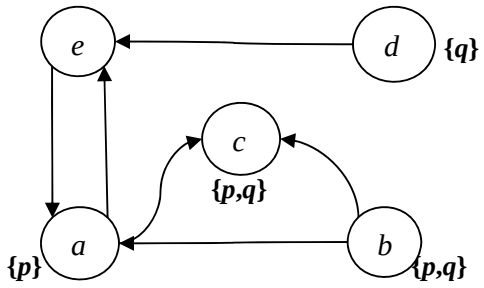
(α) Να παρουσιάσετε το μοντέλο M γραφικά.

(β) Για κάθε μια από τις πιο κάτω ιδιότητες να αποφασίσετε κατά πόσο υπάρχει κόσμος του μοντέλου M που να τις ικανοποιεί.

i. $\Box\neg p \wedge \Box\Box\neg p$
ii. $\Diamond q \wedge \neg\Box q$
iii. $\Diamond p \vee \Diamond q$
iv. $\Diamond(p \vee \Diamond q)$
v. $\Box p \vee \Box\neg p$
vi. $\Box(p \vee \neg p)$

<u>Λύση:</u>

(α) Ακολουθεί το δοσμένο μοντέλο σε γραφική αναπαράσταση.



(β)

	$\Box\neg p \wedge \Box\Box\neg p$	$\Diamond q \wedge \neg\Box q$	$\Diamond p \vee \Diamond q$	$\Diamond(p \vee \Diamond q)$	$\Box p \vee \Box\neg p$	$\Box(p \vee \neg p)$
a	False	True	True	True	False	True
b	False	True	True	True	True	True
c	True	False	False	False	True	True
d	False	False	False	False	True	True
e	False	False	True	True	True	True

Άσκηση 5

Να δείξετε ότι οι πιο κάτω προτάσεις του βασικού τροπικού λογισμού είναι έγκυρες.

(α) $\Diamond(\phi \vee \psi) \leftrightarrow (\Diamond\phi \vee \Diamond\psi)$

(β) $\Diamond T \rightarrow (\Box\phi \rightarrow \Diamond\phi)$

Λύση:

(α) Η πρόταση είναι αληθής. Η απόδειξη έχει ως εξής:

Έστω κόσμος w σε κάποιο μοντέλο Kripke M. Τότε

$w \Vdash \Diamond(\phi \vee \psi)$ αν και μόνο αν $x \Vdash \phi \vee \psi$ για κάποιο x τέτοιο ώστε $(w,x) \in R$

αν και μόνο αν $x \Vdash \phi$ ή $x \Vdash \psi$ για κάποιο x τέτοιο ώστε $(w,x) \in R$

αν και μόνο αν $x \Vdash \phi$ για κάποιο x τέτοιο ώστε $(w,x) \in R$

ή $x \Vdash \psi$ για κάποιο x τέτοιο ώστε $(w,x) \in R$

αν και μόνο αν $w \Vdash \Diamond\phi$ ή $w \Vdash \Diamond\psi$

αν και μόνο αν $w \Vdash \Diamond\phi \vee \Diamond\psi$

Αυτό ολοκληρώνει την απόδειξη.

(β) Η πρόταση είναι αληθής. Η απόδειξη έχει ως εξής:

Έστω κόσμος w σε κάποιο μοντέλο Kripke M. Έχουμε τα εξής:

Έστω $w \Vdash \Diamond T$ και $w \Vdash \Box\phi$

τότε $x \Vdash T$ για κάποιο x τέτοιο ώστε $(w,x) \in R$

και $y \Vdash \phi$ για κάθε y τέτοιο ώστε $(w,y) \in R$

τότε True για κάποιο x τέτοιο ώστε $(w,x) \in R$

και $y \Vdash \phi$ για κάθε y τέτοιο ώστε $(w,y) \in R$

τότε υπάρχει κάποιο x τέτοιο ώστε $(w,x) \in R$

και $y \Vdash \phi$ για κάθε y τέτοιο ώστε $(w,y) \in R$

τότε $x \Vdash \phi$ για κάποιο x τέτοιο ώστε $(w,y) \in R$

τότε $w \Vdash \Diamond\phi$

Αυτό ολοκληρώνει την απόδειξη.

Άσκηση 6 (20 μονάδες)

Να αποδείξετε τα πιο κάτω λογικά επακόλουθα του Τροπικού Λογισμού KT45.

(α) $\Box(\Box p \rightarrow \Box q) \vee \Box(\Box q \rightarrow \Box p)$

(β) $\Box(\Diamond p \rightarrow q) \leftrightarrow \Box(p \rightarrow \Box q)$

Λύση:

(α)

1.	$\Box p \vee \neg\Box p$	LEM
2.	$\Box p$	πρ. υπ.
3.	$\Box q$	πρ. υπ
4.	$\Box p$	copy 2
5.	$\Box q \rightarrow \Box p$	$\rightarrow$ 3-4
6.	$\Box(\Box q \rightarrow \Box p)$	$\Box$ i 3-5
7.	$\Box(\Box p \rightarrow \Box q) \vee \Box(\Box q \rightarrow \Box p)$	$\vee$ i 6
8.	$\neg\Box p$	πρ. υπ.
9.	$\Box\neg\Box p$	Καν. 5, γραμμή 2
10.	$\Box p$	πρ. υπ
	$\neg\Box p$	$\Box$ e 3
	$\bot$	$\neg$e 3,5
	$\Box q$	$\bot$ e 6
	$\Box p \rightarrow \Box q$	$\rightarrow$ 4-7
	$\Box(\Box p \rightarrow \Box q)$	$\Box$ i 4-8
	$\Box(\Box p \rightarrow \Box q) \vee \Box(\Box q \rightarrow \Box p)$	$\vee$ i 9
11.	$\Box(\Box p \rightarrow \Box q) \vee \Box(\Box q \rightarrow \Box p)$	$\vee$e 1, 2-10

(β) Θα αποδείξουμε τις δύο κατευθύνσεις ξεχωριστά αφού αντικαταστήσουμε τον όρο $\Diamond p$ με τον ισοδύναμο $\neg\Box\neg p$.

1.	$\Box(\neg\Box\neg p \rightarrow q)$	πρ. υπόθεση
2.	p	πρ. υπόθεση
3.	$\neg\Box\neg p \rightarrow q$	$\Box$ e 1
4.	$\neg q$	πρ. υπόθεση
5.	$\neg\neg\Box\neg p$	MT 3, 4
6.	$\Box\neg p$	$\neg\neg$ e 5
7.	$\neg p$	T 6
8.	$\bot$	$\neg$ e 7, 2
9.	q	RAA 4-8
10.	$\Box q$	$\Box$ i 4-9
11.	$p \rightarrow \Box q$	$\rightarrow$ i 2-10
12.	$\Box(p \rightarrow \Box q)$	$\Box$ i 2-11
13.	$\Box(\neg\Box\neg p \rightarrow q) \rightarrow \Box(p \rightarrow \Box q)$	$\rightarrow$ i 1-12

1.	$\Box(p \to \Box q)$	πρ. υπόθεση
2.	$\neg\Box\neg p$	πρ. υπόθεση
3.	$\neg q$	πρ. υπόθεση
4.	$\Box q$	πρ. υπόθεση
5.	q	T 4
6.	$\bot$	$\neg$e 3, 5
7.	$\neg\Box q$	$\neg$i 4-6
8.	$p \to \Box q$	$\Box$ e 1
9.	$\neg p$	MT 8, 7
10.	$\Box\neg p$	$\Box$ i 3-9
11.	$\bot$	$\neg$e 2, 10
12.	q	RAA 3-11
13.	$\neg\Box\neg p \to q$	$\to$ i 2-12
14.	$\Box(\neg\Box\neg p \to q)$	$\Box$ i 2-13
15.	$\Box(p \to \Box q) \to \Box(\neg\Box\neg p \to q)$	$\to$i 1-14