Άσκηση Προσομοίωσης

## Συστήματα Αναμονής

**Όνομα:** Σταύρος

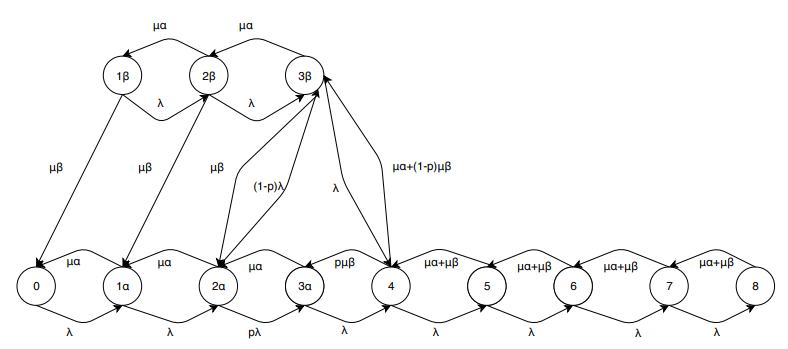
**Επώνυμο:** Σταύρου

**ΑΜ:** 03115701

**Εξάμηνο:** 6ο-ΗΜΜΥ

**Ακαδημαϊκό Έτος:** 2017-2018

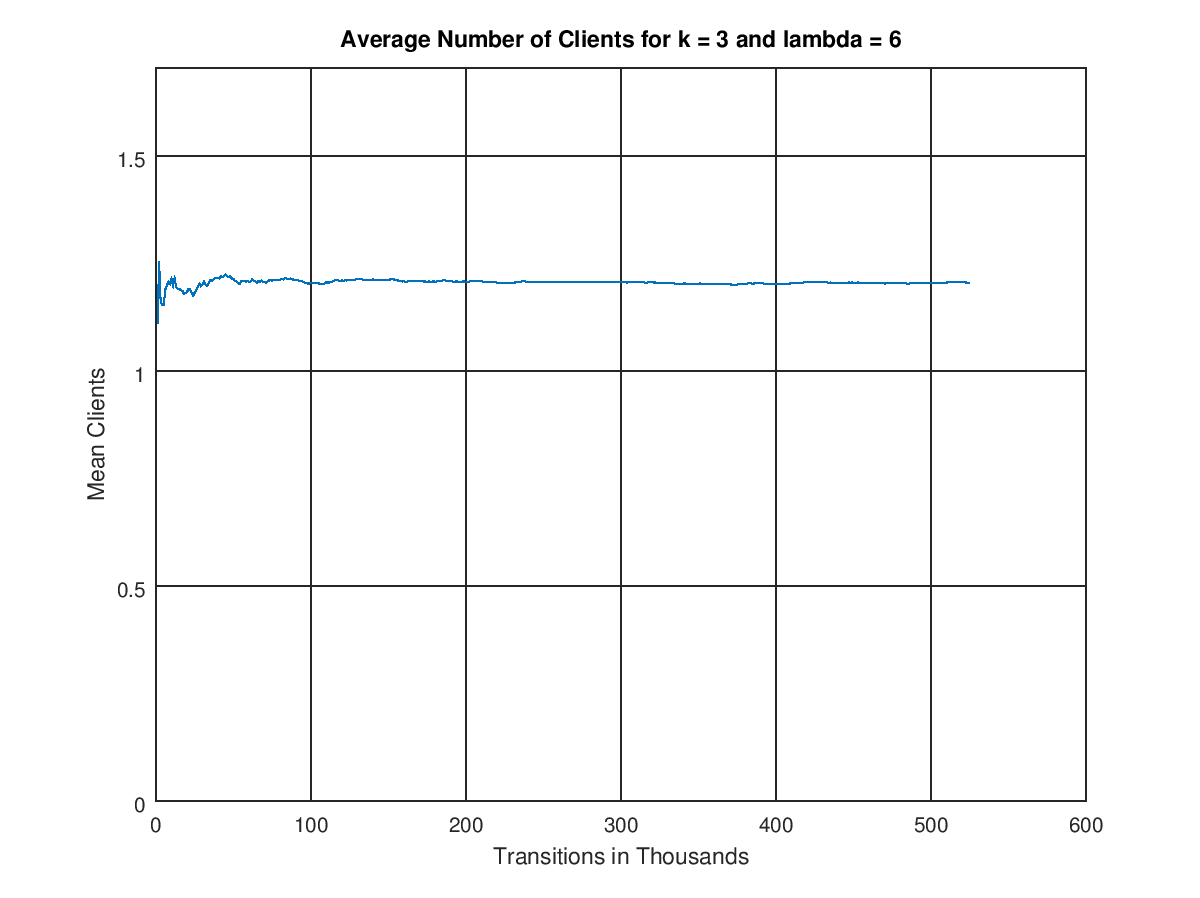
Με κατώφλι την κατάσταση 3 έχουμε το εξής διάγραμμα μεταβάσεων:

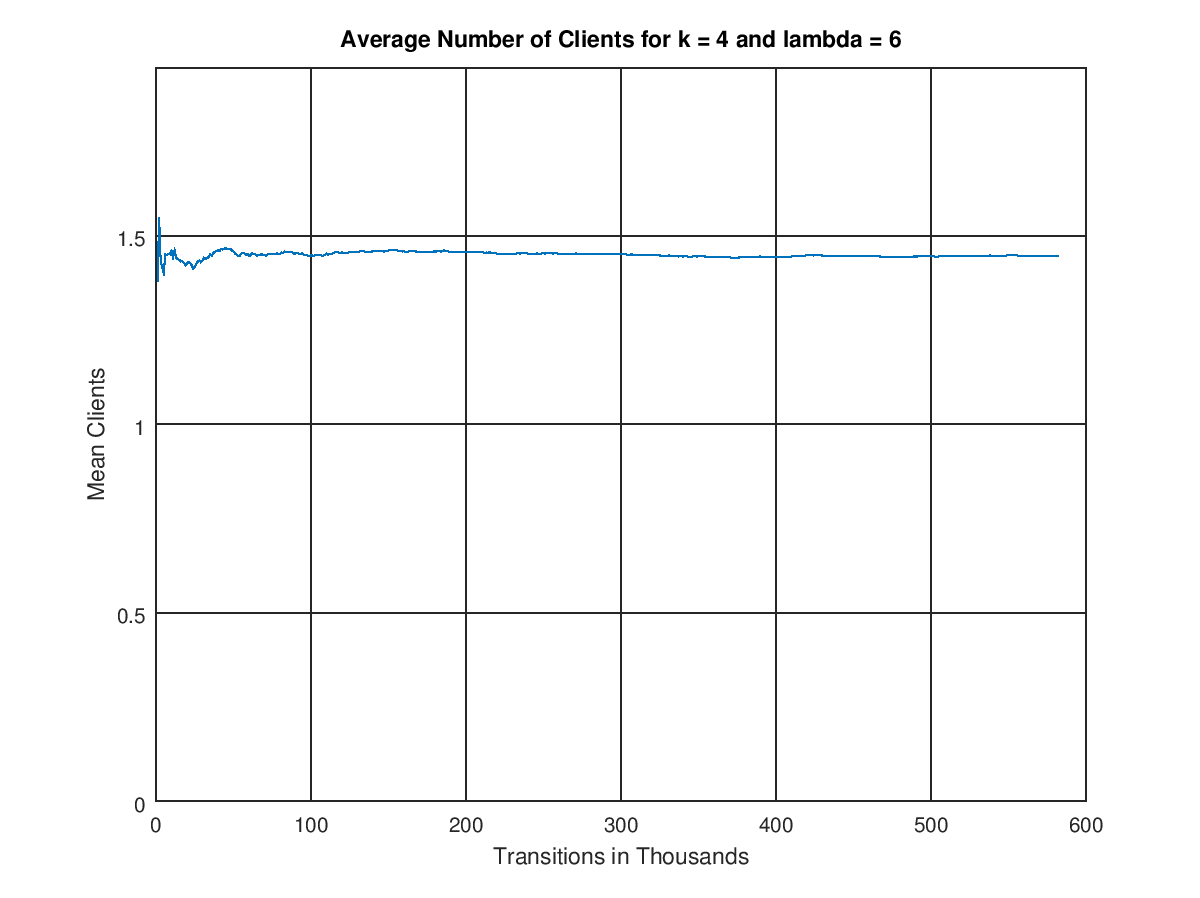


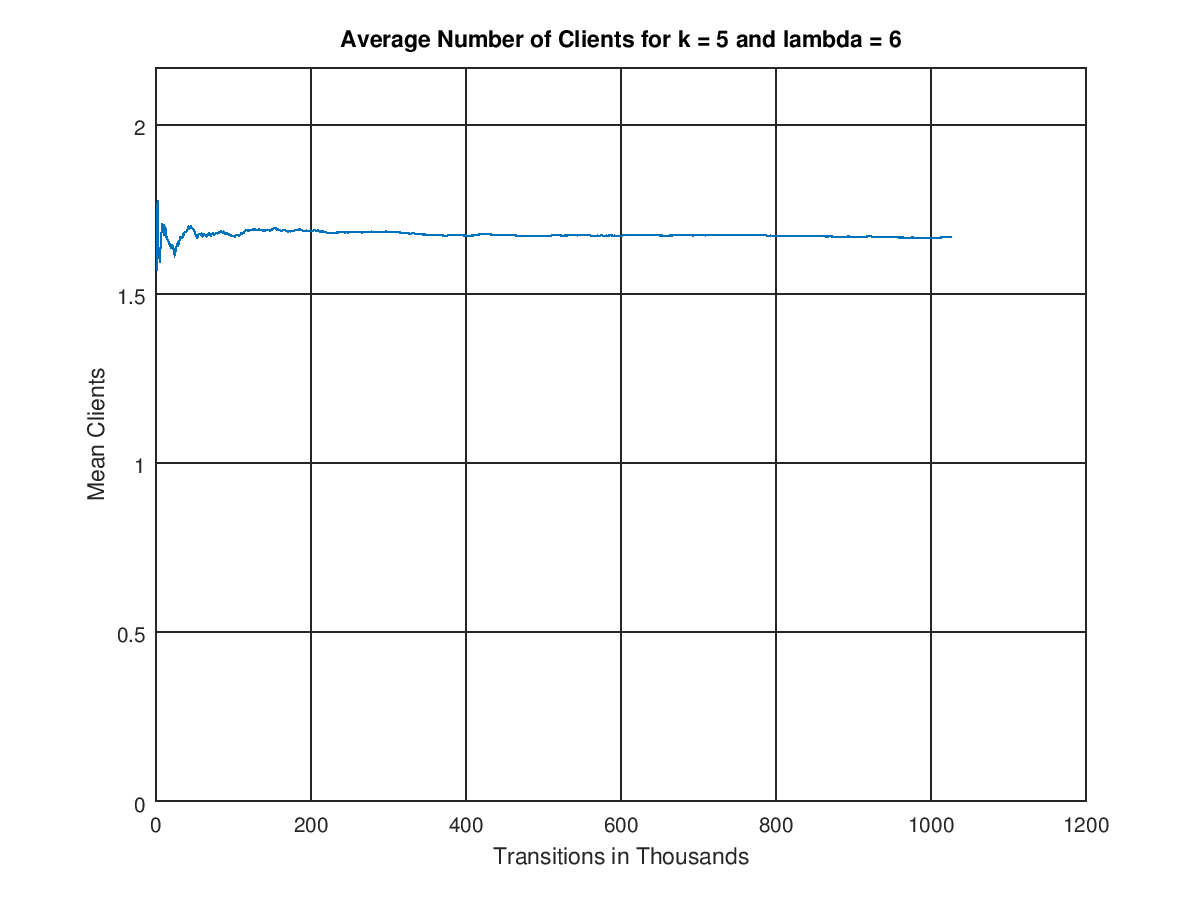
Ο κώδικας της προσομοίωσης βασίστηκε στο παραπάνω διάγραμμα μεταβάσεων. Αυτός επισυνάπτεται σε ξεχωριστό αρχείο και επίσης παρατίθεται στο παράρτημα στο τέλος. Επίσης, σαν κριτήριο σύγκλισης επιλέχθηκε η διαφορά 0.000001 ανάμεσα σε 2 διαδοχικές τιμές μέσων πελατών.

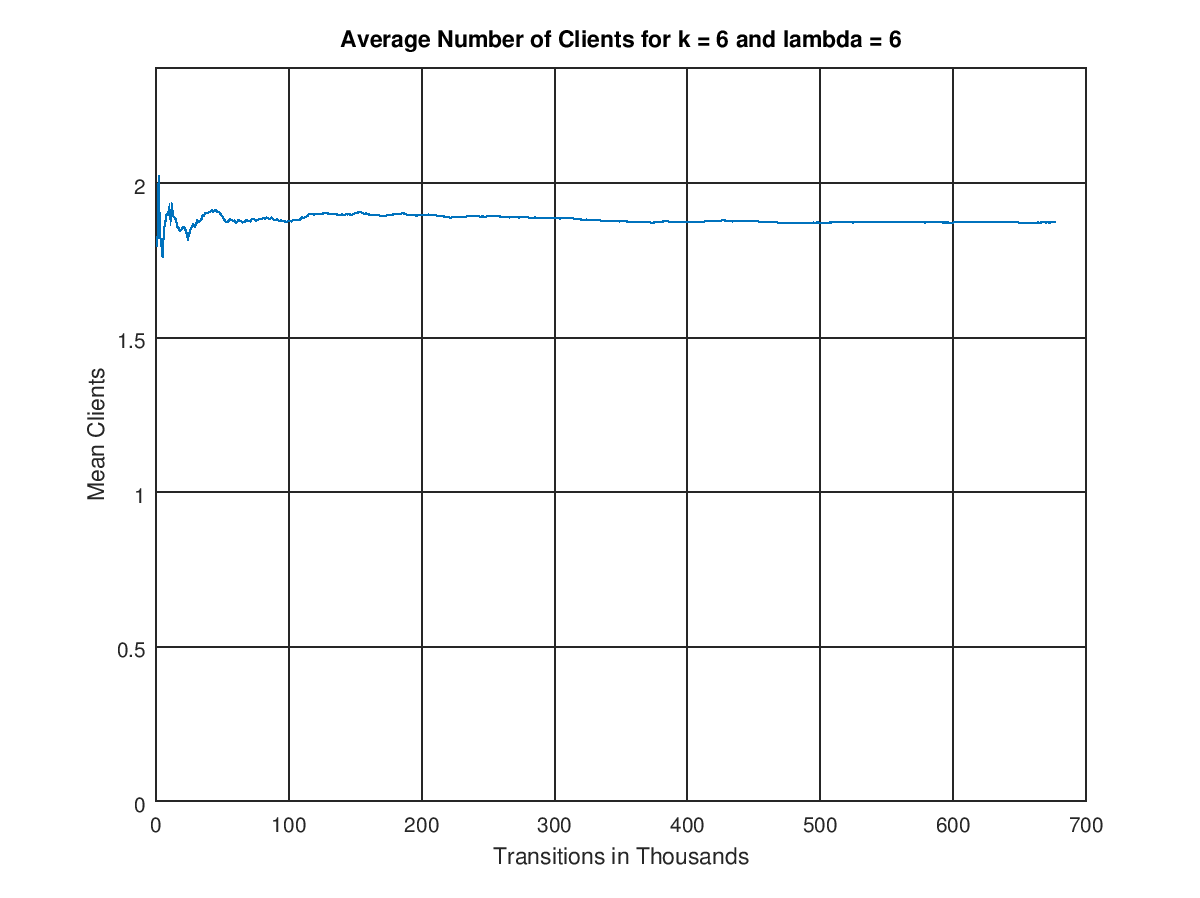
**Ερώτημα 1**

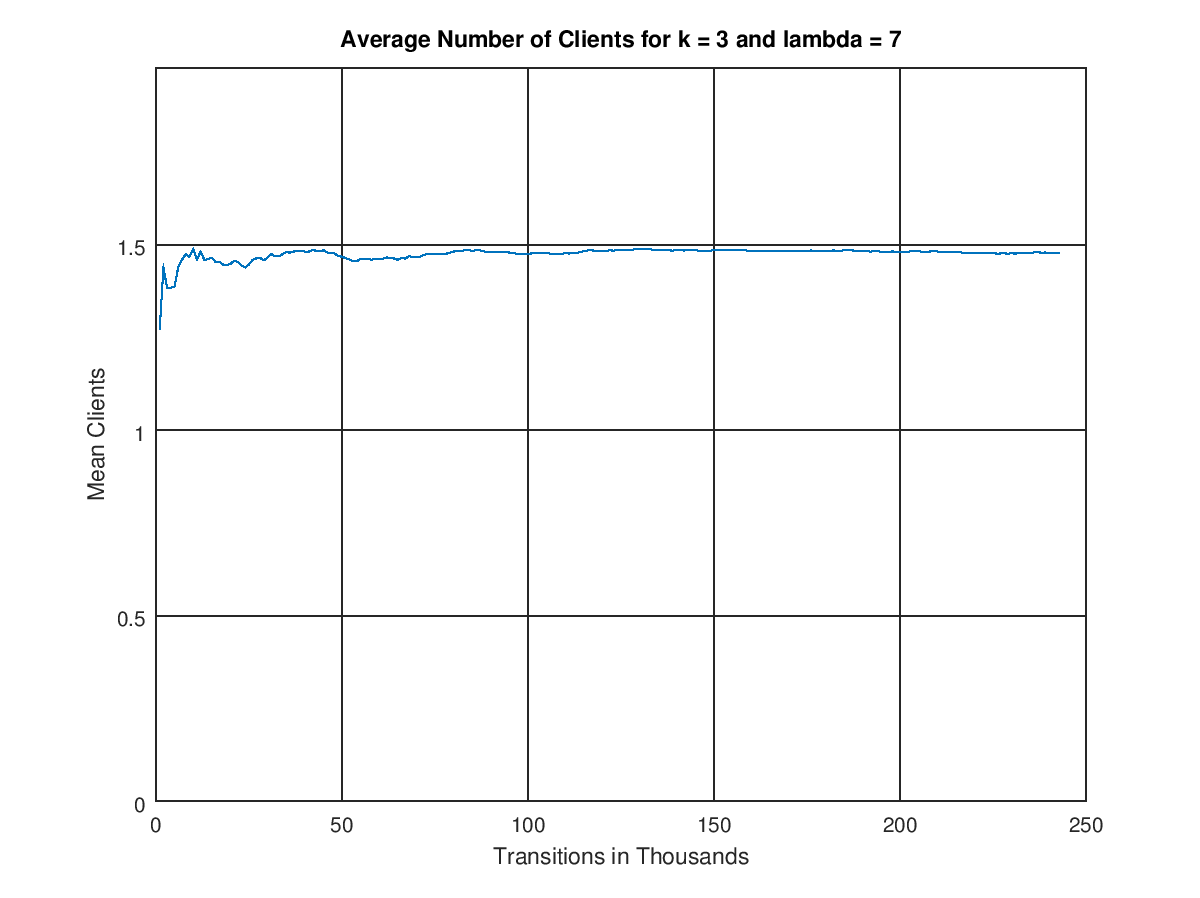
Η προσομοίωση στην Octave δίνει τις εξής 12 γραφικές παραστάσεις που απεικονίζουν την εξέλιξη του μέσου αριθμού πελατών στο σύστημα, μέχρι αυτό να φτάσει σε σύγκλιση.

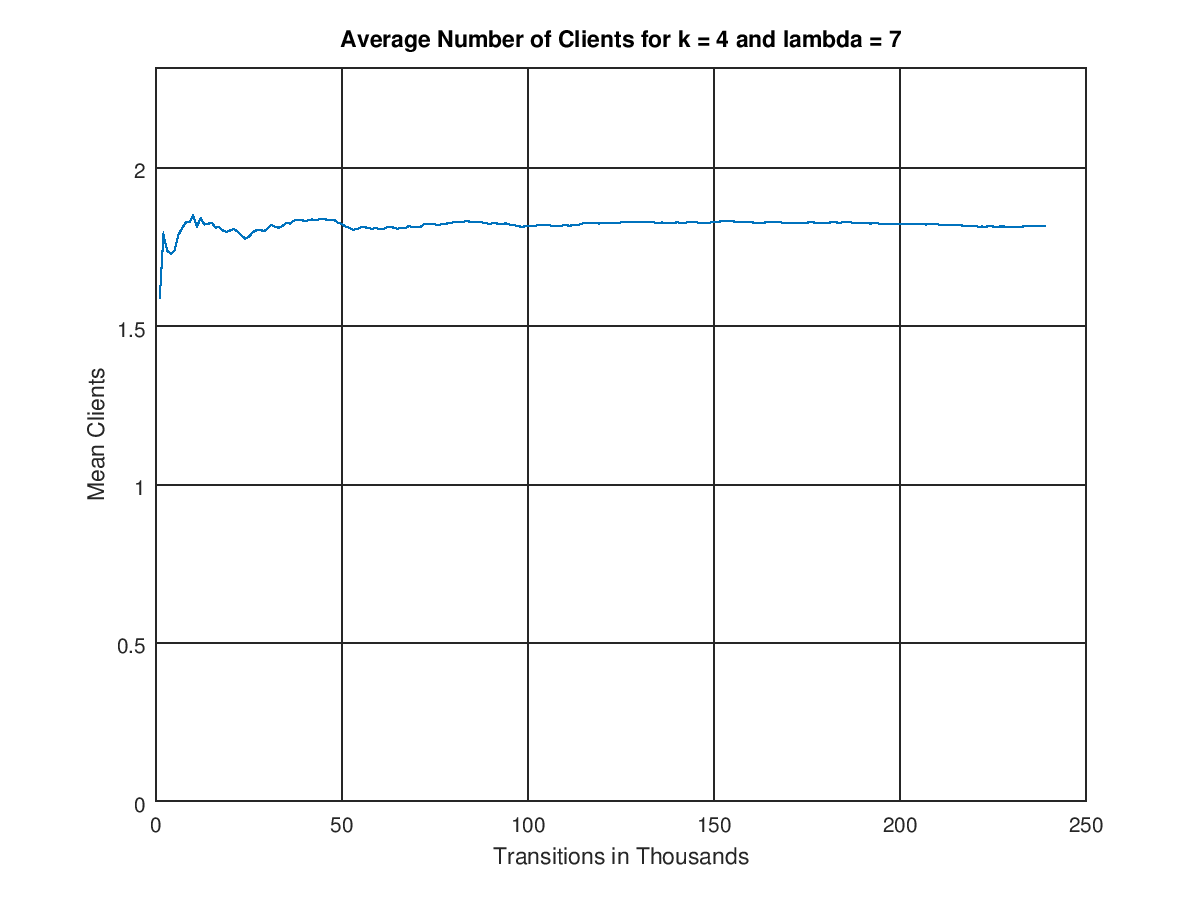


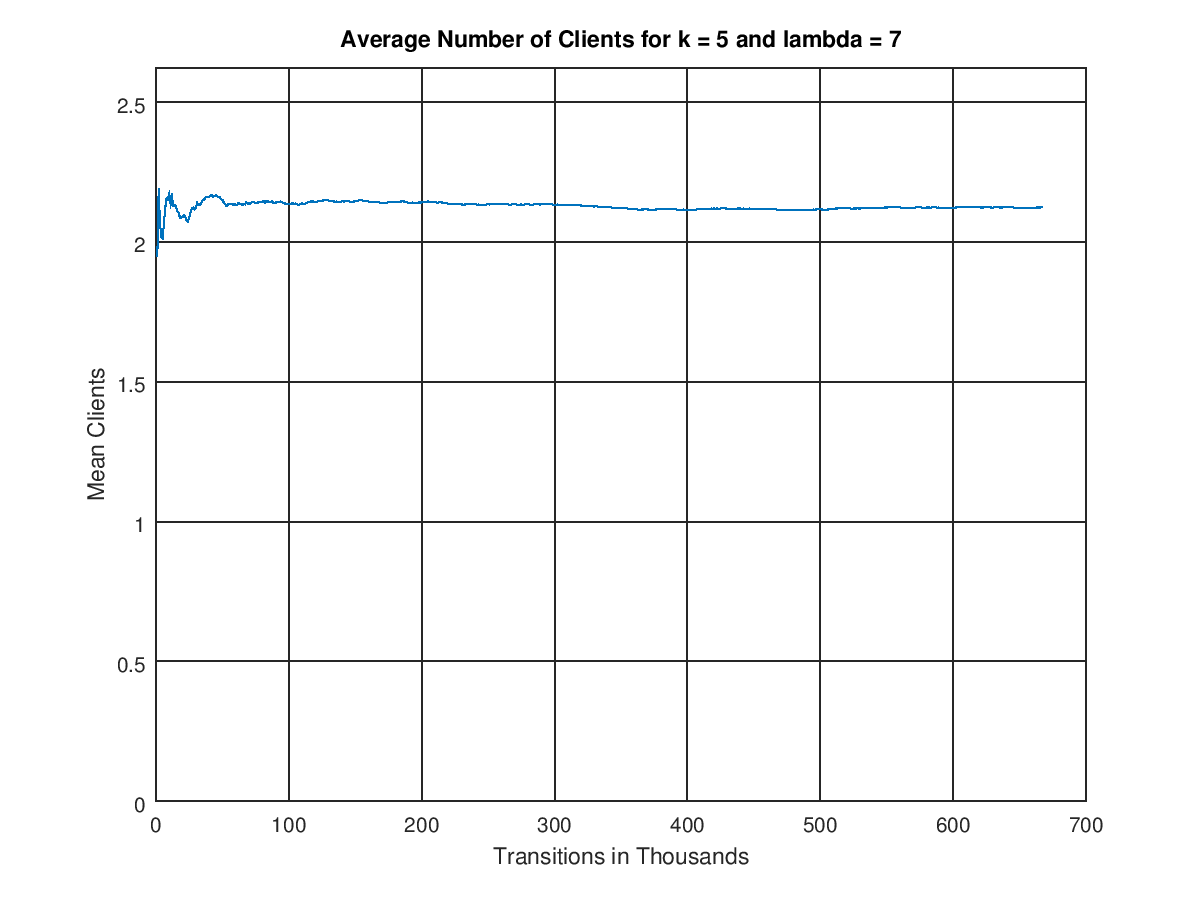


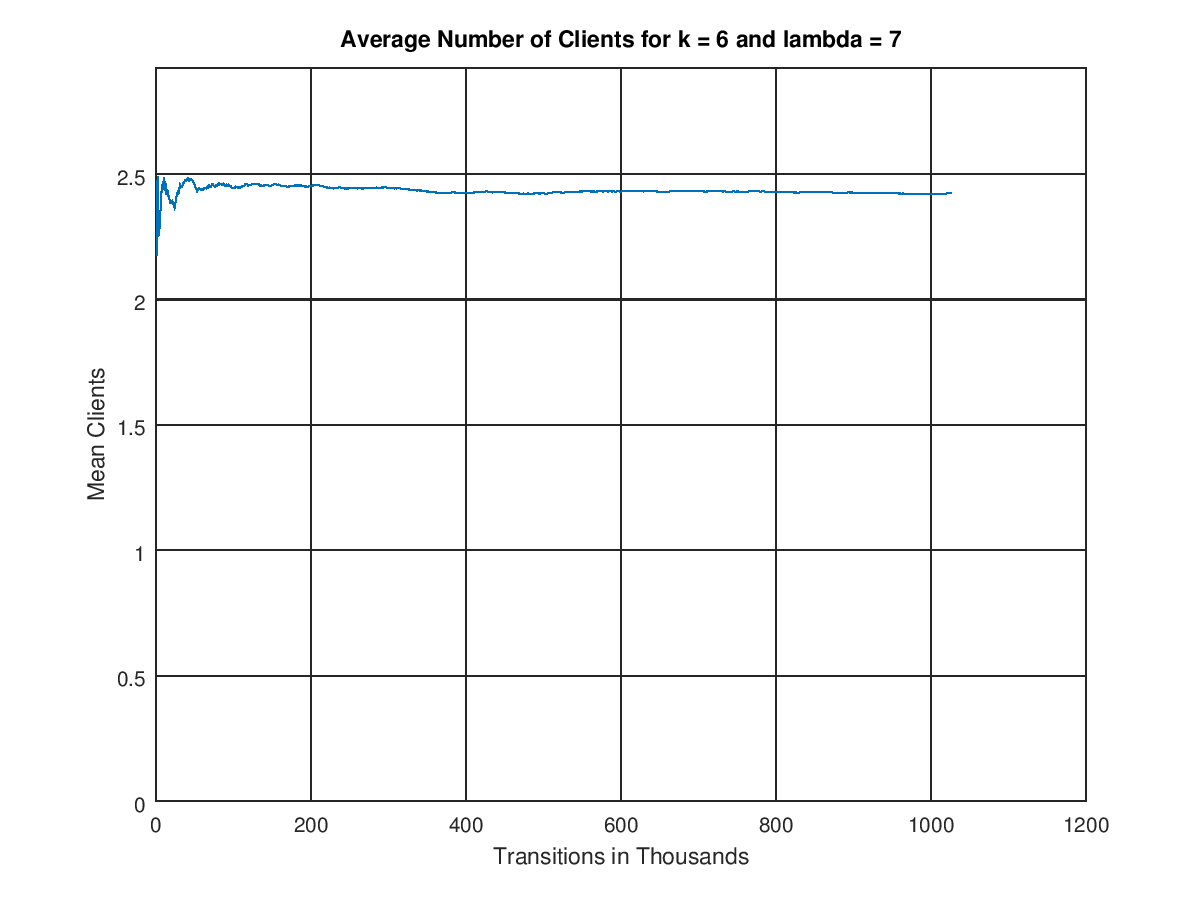


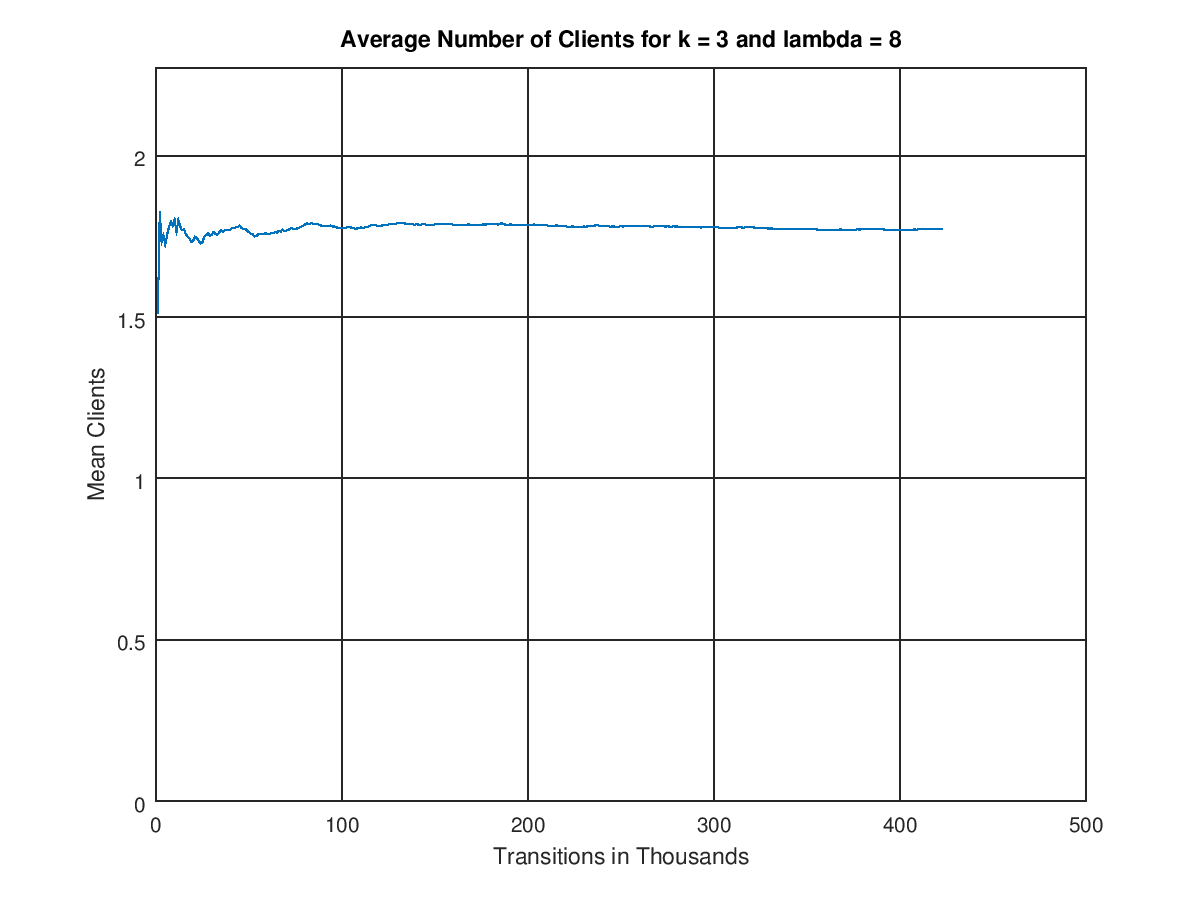


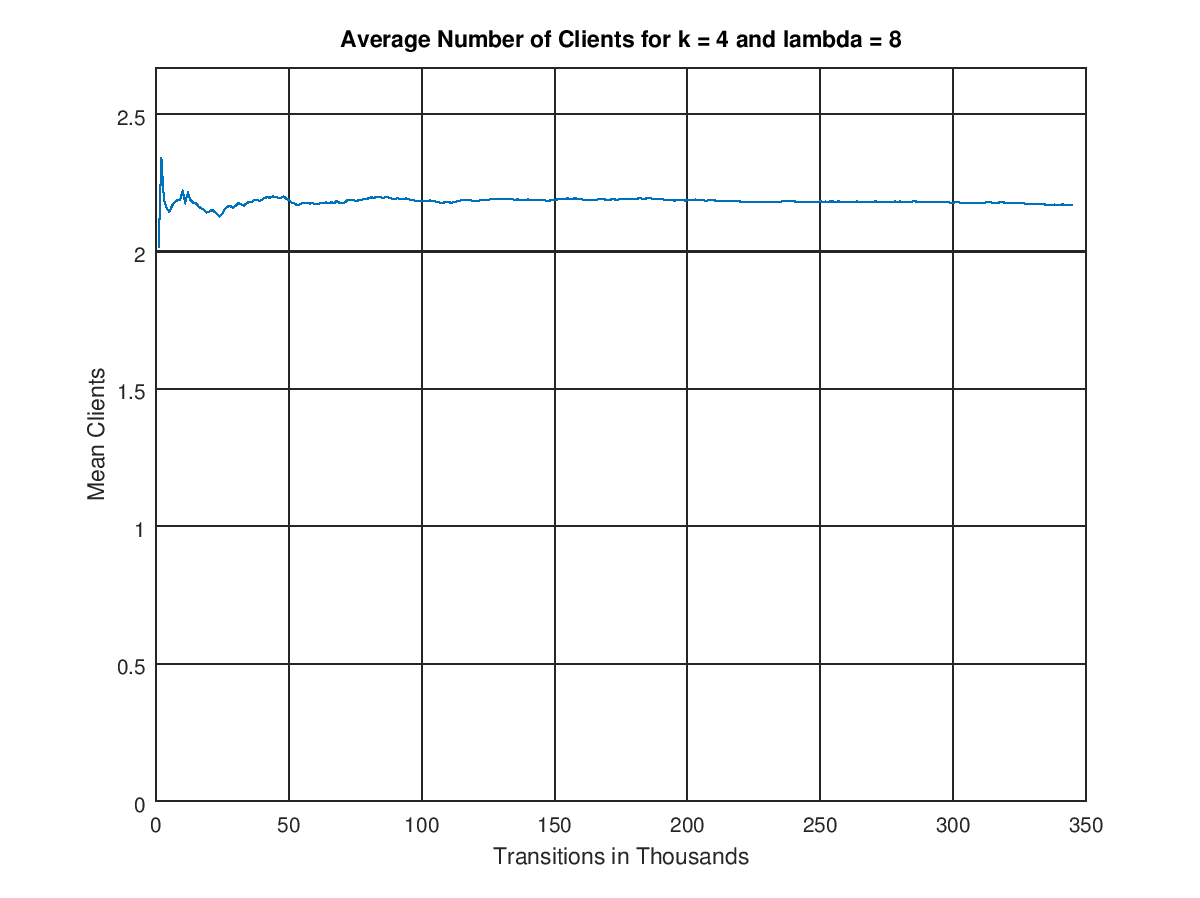


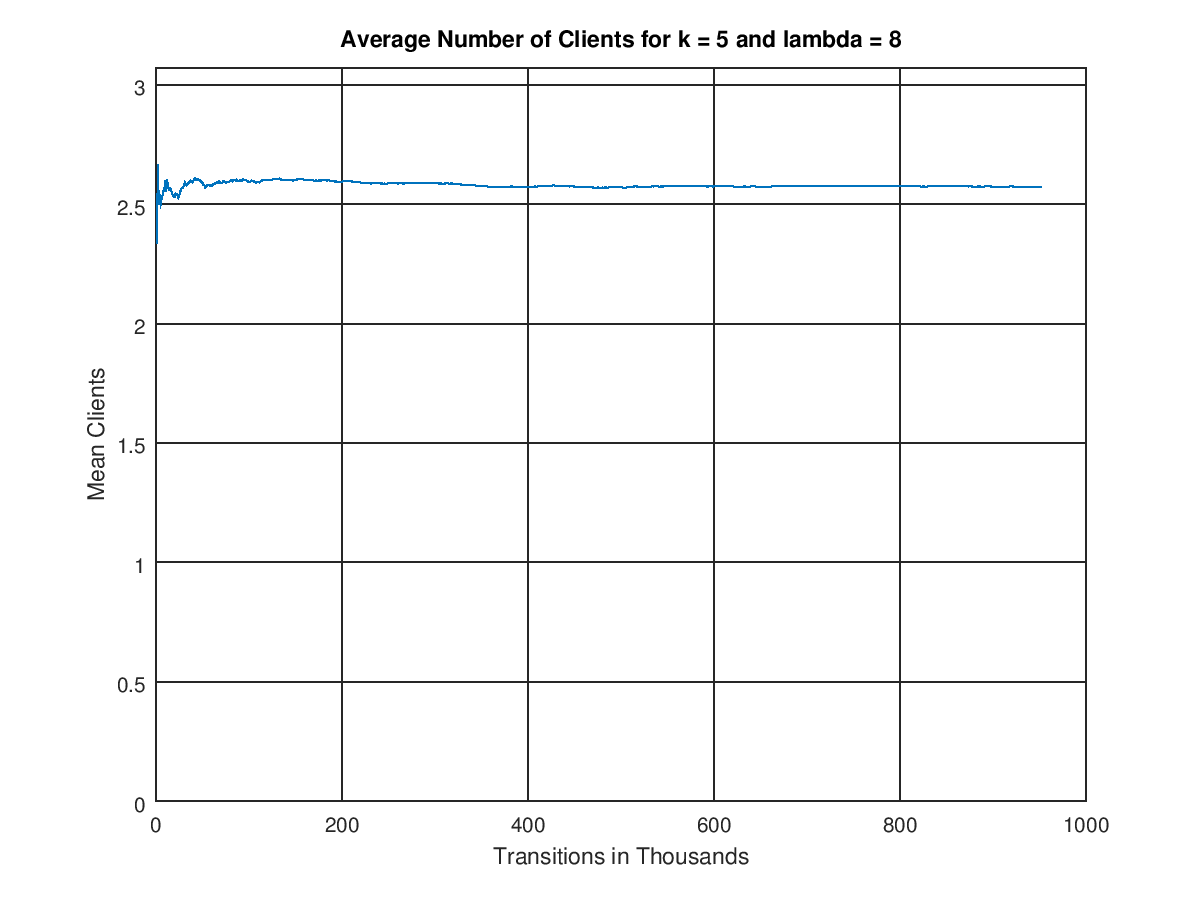


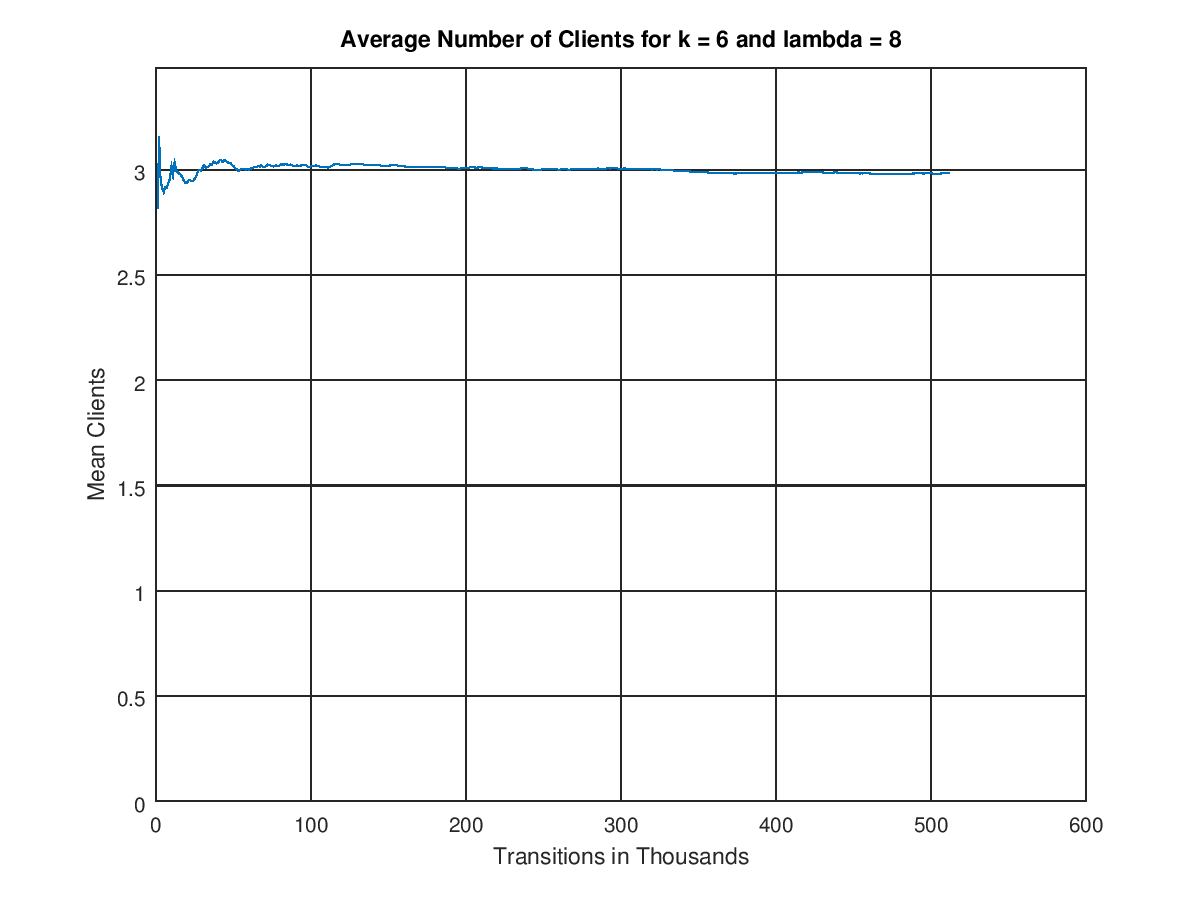




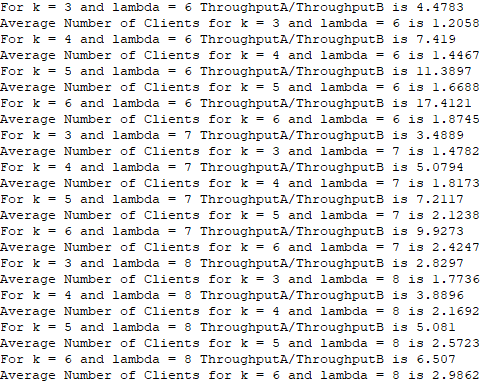








Ακόμη παίρνουμε το εξής από την Octave (που χρησιμοποιούμε για απάντηση των ερωτημάτων 2 και 3):



**Ερώτημα 2**

Βάση του πιο πάνω έχουμε τον εξής πίνακα για τον μέσο αριθμό πελατών μετά τη σύγκλιση:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | k = 3 | k = 4 | k = 5 | k =6 |
| λ = 6 | 1.2058 | 1.4467 | 1.6688 | 1.8745 |
| λ = 7 | 1.4782 | 1.8173 | 2.1238 | 2.4247 |
| λ = 8 | 1.7736 | 2.1692 | 2.5723 | 2.9862 |

**Ερώτημα 3**

Παρατηρούμε πως για λ = 6 έχουμε για k = 4, για λ = 7 έχουμε για k = 4 και για λ = 8 έχουμε για k = 5. Θα πειραματιστώ με τις τιμές και . Έχουμε τον εξής πίνακα για τις τιμές του λόγου

|  |  |  |
| --- | --- | --- |
|  |  |  |
| λ = 6, k = 4 | 6.924 | 7.9722 |
| λ = 7, k = 4 | 4.8955 | 5.457 |
| λ = 8, k = 5 | 4.9129 | 5.3329 |

Παρατηρούμε ότι μείωση της πιθανότητας p (δηλαδή ο β ενεργοποιείται πιο συχνά) φέρνει μείωση του λόγου, που είναι λογικό, καθώς ο β εξυπηρετεί περισσότερους πελάτες από πριν. Από την άλλη αύξηση του p (δηλαδή ο β ενεργοποιείται πιο αραιά) φέρνει αύξηση του λόγου, καθώς ο β εξυπηρετεί ακόμα λιγότερους πελάτες από πριν.

**Ερώτημα 4**

Για ακριβότερη απάντηση στα παρακάτω κάθε υπο-προσομοίωση είχε αρχικοποιημένο το seed τυχαίων αριθμών στο 1234. Με αυτόν τον τρόπο «απαλείψαμε» την τυχαιότητα από το σύστημα.

1. Παρόλο που αυτό δεν φαίνεται να ισχύει πάντα, εν τούτοις, φαίνεται η αύξηση του κατωφλίου να φέρνει μείωση της ταχύτητας σύγκλισης της προσομοίωσης. Αυτό ίσως να οφείλεται στο ότι αύξηση του κατωφλίου φέρνει επιπλέον καταστάσεις στο σύστημα.
2. Βλέπουμε πως αύξηση του κατωφλίου φέρνει μείωση της απόδοσης (ύπαρξη περισσοτέρων πελατών στο σύστημα). Αυτό φαίνεται λογικό για το λόγο ότι με ένα μεγαλύτερο κατώφλι, έχουμε τον 2ο εξυπηρετητή του συστήματος ανενεργό για περισσότερο χρόνο. Αυτό προκαλεί και την εξυπηρέτηση λιγότερων πελατών στο ίδιο χρονικό διάστημα.

**ΠΑΡΑΡΤΗΜΑ**

clc;

clear all;

close all;

mu = 8;

figure\_count = 1;

prob = 0.5;

% montelopoioume to sistima. Oi katastaseis einai k + 9. Exoume ti xrisi enos tixaiou arithmou\

for lambda = 6:1:8

for k = 3:1:6

% exoume sinolika 12 epanalipseis, arxikopoioume oles tis metavlites pou xreiazomaste

rand("seed",1234); % to idio seed se kathe ipo-prosomiosi gia na mporoume na elegksoume tin taxitita sigklisis

clear to\_plot;

index = 0;

transitions = 0;

sigklisi = 0;

total\_arrivals = 0;

P = zeros(1, k + 9);

arrivals = zeros(1, k + 9);

cur\_state = 1;

prev\_mean = 0;

threshold0 = lambda / (lambda + mu);

threshold1 = lambda / (lambda + 2 \* mu);

threshold2 = (lambda + mu) / (lambda + 2 \* mu);

threshold3 = threshold0 \* prob;

while !sigklisi

transitions++;

decision = rand(1);

if cur\_state == 1

arrivals(1)++;

total\_arrivals++;

cur\_state++;

elseif (cur\_state == k) && (decision > threshold0)

cur\_state--;

elseif (cur\_state == k) && (decision < threshold3)

arrivals(cur\_state)++;

total\_arrivals++;

cur\_state++;

elseif (cur\_state == k) && (decision > threshold3) && (decision < threshold0)

arrivals(cur\_state)++;

total\_arrivals++;

cur\_state = k + 9;

elseif (cur\_state == k + 2) && (decision < threshold1)

arrivals(cur\_state)++;

total\_arrivals++;

cur\_state++;

elseif (cur\_state == k + 2) && (decision > threshold1) && (decision < (lambda + prob \* mu) / (lambda + 2 \* mu))

cur\_state--;

elseif (cur\_state == k + 2) && (decision > (lambda + prob \* mu) / (lambda + 2 \* mu))

cur\_state = k + 9;

elseif (cur\_state == 9) && (decision < threshold1)

arrivals(cur\_state)++;

total\_arrivals++;

elseif (cur\_state == 9) && (decision > threshold1)

cur\_state--;

elseif (cur\_state == k + 9) && (decision < threshold1)

total\_arrivals++;

arrivals(cur\_state)++;

cur\_state = k + 2;

elseif (cur\_state == k + 9) && (decision > threshold1) && (decision < threshold2)

cur\_state = k;

elseif (cur\_state == k + 9) && (decision > threshold2)

cur\_state--;

elseif (cur\_state == 10) && (decision < threshold0)

total\_arrivals++;

arrivals(cur\_state)++;

cur\_state++;

elseif (cur\_state == 10) && (decision > threshold0)

cur\_state = 1;

elseif (cur\_state == k + 1) && (decision < threshold0)

arrivals(cur\_state)++;

total\_arrivals++;

cur\_state++;

elseif (cur\_state == k + 1) && (decision > threshold0)

cur\_state--;

elseif (cur\_state < k) && (decision < threshold0)

arrivals(cur\_state)++;

total\_arrivals++;

cur\_state++;

elseif (cur\_state < k) && (decision > threshold0)

cur\_state--;

elseif (cur\_state < 9) && (decision < threshold1)

arrivals(cur\_state)++;

total\_arrivals++;

cur\_state++;

elseif (cur\_state < 9) && (decision > threshold1)

cur\_state--;

elseif (cur\_state > 10) && (decision < threshold1)

total\_arrivals++;

arrivals(cur\_state)++;

cur\_state++;

elseif (cur\_state > 10) && (decision > threshold1) &&(decision < threshold2)

cur\_state -= 9;

elseif (cur\_state > 10) && (decision > threshold2)

cur\_state--;

endif

% kathe 1000 metavaseis elegxoume gia sigklisi

if mod(transitions,1000) == 0

index++;

for i=1:1:length(arrivals)

P(i) = arrivals(i) ./ total\_arrivals;

endfor

mean\_clients = 0;

for i=1:1:9

mean\_clients += P(i) .\* (i - 1);

endfor

for i = 10:1:length(arrivals)

mean\_clients += P(i) .\* (i - 9);

endfor

to\_plot(index) = mean\_clients;

if abs(mean\_clients - prev\_mean) < 0.000001

sigklisi = 1;

endif

prev\_mean = mean\_clients;

endif

endwhile

% ipologizoume ta throughput kai parousiazoume ta apotelesmata mas gia kathe sindiasmo lambda kai k

utilA = 1 - P(1) - P(10);

utilB = 1;

for i = 1:1:k+1

utilB -= P(i);

endfor

tptA = utilA \* mu;

tptB = utilB \* mu;

logos = tptA / tptB;

disp(cstrcat("For k = ", num2str(k), " and lambda = ", num2str(lambda), " ThroughputA/ThroughputB is ", num2str(logos)));

title\_str = cstrcat("Average Number of Clients for k = ", num2str(k), " and lambda = ", num2str(lambda));

disp(cstrcat(title\_str, " is ", num2str(mean\_clients)));

figure(figure\_count++);

plot(to\_plot);

grid on;

title(title\_str);

ylim([0 mean\_clients+0.5]);

xlabel("Transitions in Thousands");

ylabel("Mean Clients");

endfor

endfor