Υπολογιστικά Νέφη
ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 2
«Προσομοίωση διακριτών γεγονότων – Ουρά Άπειρων Γεγονότων»
Οκτώβριος 2024

Συμπληρώστε τα στοιχεία σας πριν την αποστολή της εργασίας	
Ονοματεπώνυμο	Δημήτριος Γκούμας
Αριθμός Μητρώου	4502

Δημιουργία τυχαίων γεγονότων και υπολογισμός αναλυτικής έκφρασης

Η προσομοίωση διακριτών γεγονότων (discrete-event simulation) αφορά τη **μοντελοποίηση ενός συστήματος καθώς αυτό εξελίσσεται με την πάροδο του χρόνου**.

Στην αναπαράσταση ενός τέτοιου συστήματος οι μεταβλητές κατάστασης μεταβάλλονται στιγμιαία σε διακεκριμένες χρονικές στιγμές. Χρησιμοποιώντας τη μαθηματική ορολογία, θα μπορούσαμε να πούμε ότι **το σύστημα μπορεί να αλλάζει μόνον σε μετρητό / διακριτό αριθμό σημείων στο χρόνο**.

Ένα γεγονός (event) μπορεί να συμβεί στα σημεία αυτά, όπου σαν γεγονός ορίζεται ένα στιγμιαίο συμβάν το οποίο μπορεί να αλλάξει την κατάσταση του συστήματος. Εξ' αιτίας της δυναμικής φύσης των μοντέλων προσομοίωσης διακριτών γεγονότων, θα πρέπει να παρακολουθούμε την τρέχουσα τιμή του χρόνου προσομοίωσης, καθώς **η προσομοίωση προχωρά και εξελίσσεται σε διακριτά γεγονότα** (διακριτά σημεία στο χρόνο). Επίσης χρειαζόμαστε κάποιον μηχανισμό που να προχωρά το χρόνο προσομοίωσης από τη μια τιμή στην άλλη.

Σε ένα μοντέλο προσομοίωσης, ονομάζουμε **ρολόι προσομοίωσης (simulation clock)** τη μεταβλητή που δίνει την τρέχουσα τιμή του χρόνου προσομοίωσης. Η μονάδα χρόνου για το ρολόι προσομοίωσης δεν ορίζεται ρητώς όταν το μοντέλο έχει γραφεί σε μια γλώσσα γενικού σκοπού (όπως είναι η C, Matlab) και θεωρείται ότι είναι η ίδια με τις μονάδες των παραμέτρων εισόδου. Θα πρέπει να έχουμε υπόψιν ότι γενικά ο χρόνος προσομοίωσης δεν έχει καμία σχέση με το χρόνο που απαιτείται για να τρέξει το μοντέλο προσομοίωσης στον υπολογιστή.

Η πιο γνωστή μέθοδος για να προχωρήσει το ρολόι προσομοίωσης είναι η μέθοδος: **"να προχωρήσει ο χρόνος στο επόμενο γεγονός" (next-event time advance).**

- Με τη μέθοδο αυτή το ρολόι προσομοίωσης ***παίρνει αρχική τιμή μηδέν*** και υπολογίζονται οι χρόνοι που θα συμβούν τα μελλοντικά γεγονότα.
- Στη συνέχεια ***το ρολόι προχωράει και δείχνει το χρόνο του πιο επικείμενου (πρώτου) μελλοντικού γεγονότος***. Στο σημείο αυτό ενημερώνεται η κατάσταση του

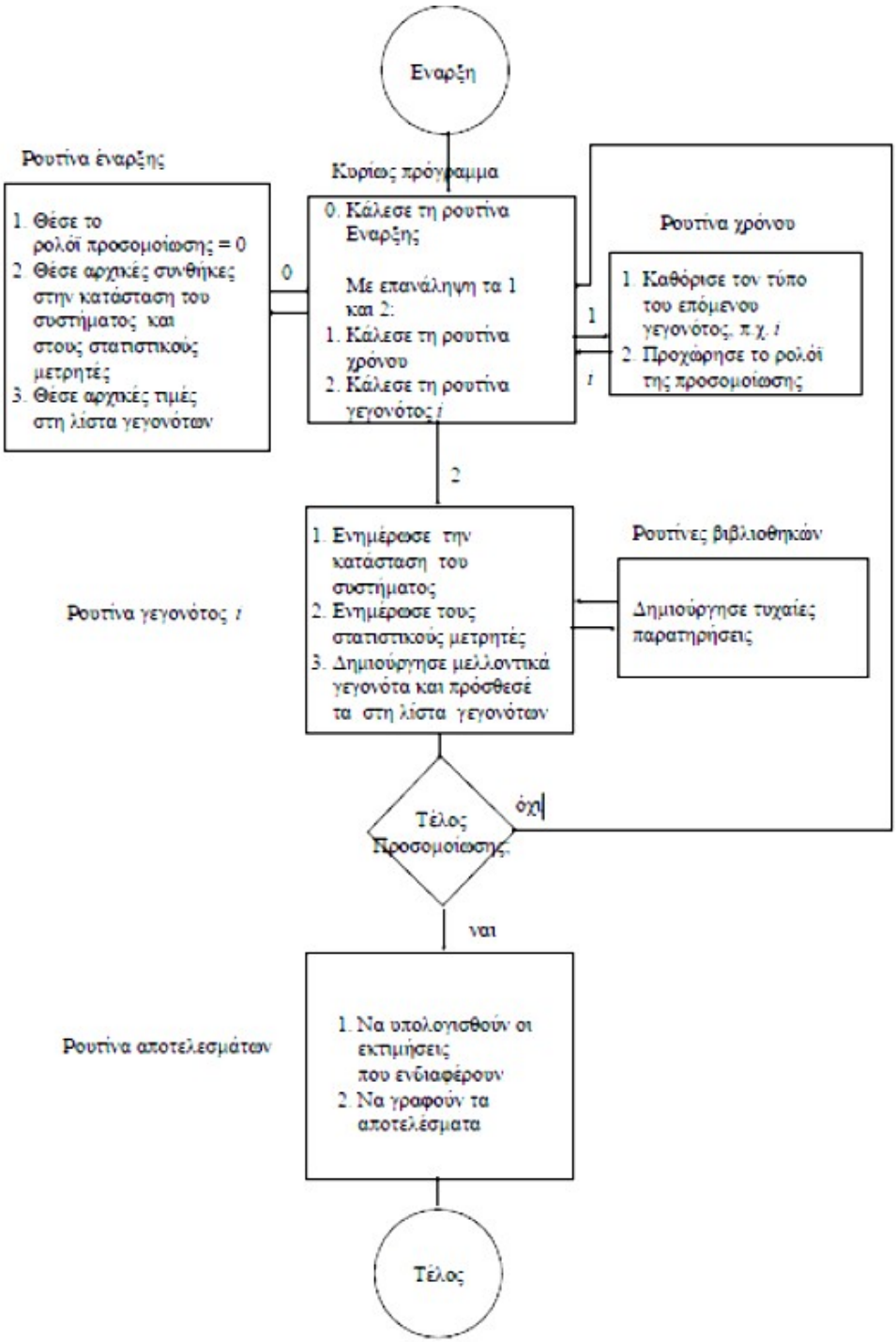
συστήματος ότι έχει συμβεί κάποιο γεγονός, αλλά πρέπει και εμείς να ενημερώσουμε τη γνώση μας για τον αριθμό των γεγονότων που πρόκειται να συμβούν στο μέλλον.
• Τότε το ***ρολόι προσομοίωσης προχωράει και δείχνει το χρόνο του καινούργιου πιο επικείμενου γεγονότος***, ενημερώνεται η κατάσταση του συστήματος και ορίζονται οι χρόνοι που θα συμβούν τα μελλοντικά γεγονότα.
• Η διαδικασία αυτή που προχωρά το ρολόι προσομοίωσης από το ένα γεγονός στο άλλο συνεχίζεται ***έως ότου τελικά ικανοποιηθεί κάποια συνθήκη τέλους.***

Παράγοντες και Οργάνωση ενός Μοντέλου Προσομοίωσης Διακριτών Γεγονότων
Όλα τα μοντέλα προσομοίωσης διακριτών γεγονότων περιέχουν ορισμένους παράγοντες οι οποίοι είναι οργανωμένοι με κάποια συγκεκριμένη λογική για την εύκολη κωδικοποίηση, εύρεση λαθών και για μελλοντικές μεταβολές του προγράμματος του μοντέλου προσομοίωσης. Πιο συγκεκριμένα, οι επόμενοι παράγοντες συναντώνται στα περισσότερα μοντέλα προσομοίωσης διακριτών γεγονότων:

- ***Κατάσταση συστήματος:*** Μια συλλογή μεταβλητών κατάστασης που είναι απαραίτητες για την περιγραφή του συστήματος κάποια συγκεκριμένη χρονική στιγμή.
- ***Ρολόι προσομοίωσης:*** Μια μεταβλητή που δίνει την τρέχουσα τιμή του χρόνου προσομοίωσης.
- ***Λίστα γεγονότων:*** Μια λίστα που περιέχει τον επόμενο χρόνο που θα συμβεί ο κάθε τύπος γεγονότος.
- ***Στατιστικοί μετρητές:*** Είναι μεταβλητές που αποθηκεύουν στατιστικές πληροφορίες σχετικά με την απόδοση του συστήματος.
- ***Ρουτίνα έναρξης:*** Ένα υπο-πρόγραμμα το οποίο ξεκινάει το μοντέλο προσομοίωσης στο χρόνο μηδέν.
- ***Ρουτίνα χρόνου:*** Είναι ένα υπο-πρόγραμμα το οποίο καθορίζει ποιο θα είναι το επόμενο γεγονός που θα συμβεί από την λίστα γεγονότων και το οποίο στη συνέχεια προχωρά το ρολόι προσομοίωσης για να δείξει το χρόνο που θα συμβεί το γεγονός αυτό.
- ***Ρουτίνα γεγονότος:*** Ένα υπο-πρόγραμμα το οποίο ενημερώνει την κατάσταση του συστήματος όταν συμβαίνει ένα γεγονός κάποιου συγκεκριμένου τύπου (υπάρχει μια ρουτίνα γεγονότος για κάθε τύπο γεγονότος).
- ***Ρουτίνες βιβλιοθηκών:*** Ένα σύνολο υπο-προγραμμάτων τα οποία χρησιμοποιούνται για την παραγωγή τυχαίων παρατηρήσεων από κατανομές πιθανοτήτων που έχουν οριστεί σαν τμήμα του μοντέλου προσομοίωσης.
- ***Ρουτίνα αποτελεσμάτων:*** Είναι ένα υπο-πρόγραμμα το οποίο υπολογίζει εκτιμήτριες των παραμέτρων της απόδοσης που αποτελούν το στόχο της μελέτης (από τους στατιστικούς μετρητές) και παράγει ένα σύνολο αποτελεσμάτων όταν τελειώσει η προσομοίωση.
- ***Κυρίως πρόγραμμα:*** Ένα υπο-πρόγραμμα το οποίο καλεί τη ρουτίνα χρόνου να καθορίσει το επόμενο γεγονός και στη συνέχεια μεταφέρει τον έλεγχο στην αντίστοιχη ρουτίνα γεγονότος για να ενημερωθεί κατάλληλα η κατάσταση του συστήματος. Το κυρίως πρόγραμμα επίσης ελέγχει την συνθήκη τέλους και καλεί τη ρουτίνα αποτελεσμάτων όταν τελειώσει η προσομοίωση.

Η ροή ελέγχου ανάμεσα στους παράγοντες αυτούς φαίνεται στο σχήμα παρακάτω.

- ***Το κυρίως πρόγραμμα καλεί τη ρουτίνα έναρξης.*** Στη ρουτίνα αυτή το ρολόι προσομοίωσης παίρνει την τιμή μηδέν οπότε και η προσομοίωση ξεκινάει στο χρόνο 0, και παίρνουν αρχικές τιμές οι στατιστικοί μετρητές καθώς και η λίστα γεγονότων.
- ***Μόλις επιστρέψει ο έλεγχος στο κυρίως πρόγραμμα***, καλεί τη ρουτίνα χρόνου για να αποφασιστεί ποιος είναι ο τύπος του πιο επικείμενου γεγονότος.
    - *Εάν το γεγονός που θα συμβεί είναι π.χ. τύπου i*, τότε ο χρόνος προσομοίωσης προχωράει και δείχνει το χρόνο που θα συμβεί το γεγονός του τύπου i, και ο έλεγχος επιστρέφει στο κυρίως πρόγραμμα.
    - Τότε το κυρίως πρόγραμμα καλεί τη ρουτίνα γεγονότος i, όπου συμβαίνουν τα ακόλουθα:
      1) η κατάσταση του συστήματος ενημερώνεται με το συμβάν του γεγονότος τύπου i,
      2) ενημερώνονται οι στατιστικοί μετρητές, και
      3) υπολογίζονται οι χρόνοι που θα συμβούν τα μελλοντικά γεγονότα και ενημερώνεται η λίστα γεγονότων.



Εικόνα 1 Ροή ελέγχου ανάμεσα στους παράγοντες

Πολλές φορές πρέπει να δημιουργήσουμε τυχαίες παρατηρήσεις από κατανομές πιθανοτήτων ούτς ώστε να καθοριστούν οι χρόνοι αυτών των μελλοντικών γεγονότων. Μια τέτοια παρατήρηση που δημιουργείται ονομάζεται τυχαία παρατήρηση (random variate), και ακολουθεί τις διαδικάσίες του προηγούμενου εργαστηρίου.

Αφού εκτελεστεί η απαραίτητη λειτουργία, είτε στο κυρίως πρόγραμμα ή στη ρουτίνα γεγονότος i, ***γίνεται έλεγχος εάν πληρείται κάποια συνθήκη τέλους*** ούτως ώστε να σταματήσει η προσομοίωση.

- *Εάν πρέπει να σταματήσει η προσομοίωση αυτή τη χρονική στιγμή*, καλείται από το κυρίως πρόγραμμα η ρουτίνα αποτελεσμάτων για να υπολογίσει τις εκτιμήσεις (από τους στατιστικούς μετρητές) εκείνων των παραμέτρων της απόδοσης που μας ενδιαφέρουν και παράγει ένα σύνολο αποτελεσμάτων.
- *Εάν δεν πληρείται η συνθήκη τέλους της προσομοίωσης, ο έλεγχος επιστρέφεται στο κυρίως πρόγραμμα* και εκτελείται συνεχώς ο κύκλος "κυρίως πρόγραμμα - ρουτίνα χρόνου - κυρίως πρόγραμμα - ρουτίνα γεγονότος - έλεγχος τέλους κύκλου" μέχρις ότου ισχύσει η συνθήκη τέλους.

Στα προηγούμενα ορίσαμε ότι ένα σύστημα είναι μια συλλογή καλώς ορισμένων οντοτήτων. Οι οντότητες χαρακτηρίζονται από τιμές δεδομένων που ονομάζονται χαρακτηριστικά (attributes). Αυτά τα χαρακτηριστικά είναι μέρος της κατάστασης του συστήματος για ένα μοντέλο προσομοίωσης διακριτών γεγονότων.

Οντότητες με κάποια κοινή ιδιότητα ομαδοποιούνται σε λίστες (lists) (ή αρχεία (files) ή σύνολα (sets)). Για κάθε οντότητα υπάρχει μια εγγραφή (record) στη λίστα που αποτελείται από τα χαρακτηριστικά της οντότητας. Η σειρά με την οποία οι εγγραφές τοποθετούνται στη λίστα εξαρτάται από κάποιο συγκεκριμένο κανόνα.

Η οργάνωση ενός προγράμματος προσομοίωσης διακριτών γεγονότων που χρησιμοποιεί το μηχανισμό "να προχωρήσει ο χρόνος στο επόμενο γεγονός"

Προσομοίωση ενός συστήματος ουρών με έναν εξυπηρετητή / διακομιστή

Στην παράγραφο αυτή θα εξετάσουμε την προσομοίωση ενός συστήματος ουρών με έναν εξυπηρέτη, που επεξεργάζεται **εντολές (tasks ή instructions) που ζητώνται (δημιουργούνται) να εκτελεστούν σε ένα υπολογιστικό σύστημα**, όπως απεικονίζεται στην Εικόνα 2.

Η βασική σχεδίαση της προσομοίωσης ενός τέτοιου συστήματος απαιτεί 4 κυρίως δραστηριότητες

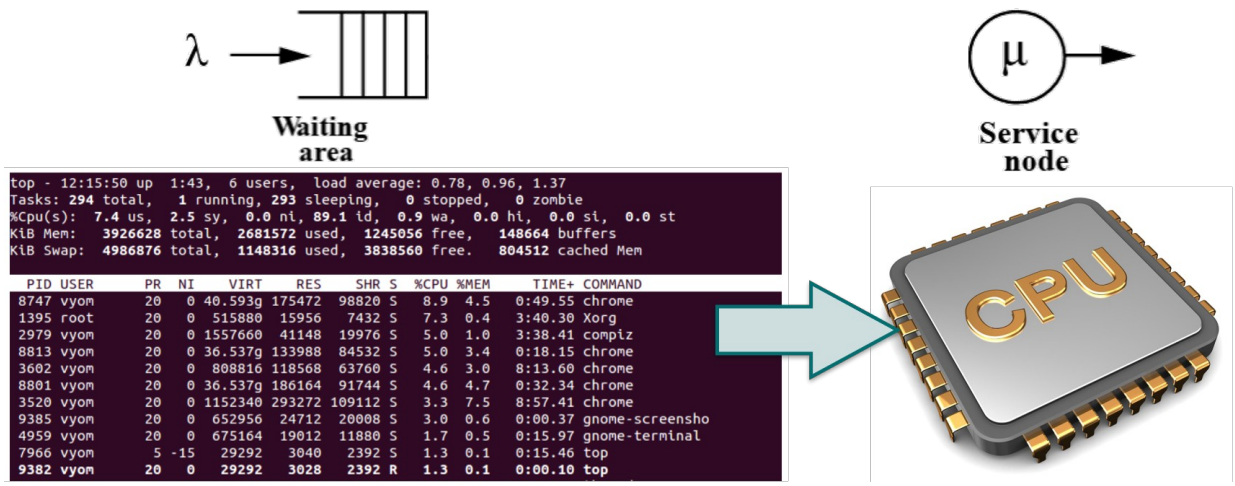
- **Μελέτη και χρονική απεικόνιση των τυχαίων νέων γεγονότων**, πότε δηλαδή ακριβώς χρονικά προκύπτει ένα νέο task (εν προκειμένω με ρυθμό άφιξης λ)
- **Πότε ξεκινάει και πόσο διαρκεί η επεξεργασία ενός task**, πότε δηλαδή ακριβώςχρονικά ξεκινάει ένα καινούριο task και άρα πότε θα αναχωρίσει / φύγει από τον επεξεργαστής (εν προκειμένω με ρυθμό εξυπηρέτησης μ)
- **Η λειτουργία του προσομοιωτή του discrete event simulator**, δηλαδή του πως προχωράει το ρολόι της προσομοίωσης, που απαιτεί ουσιαστικά να «πρσομοιώσουμε» την τωρινή κατάσταση (π.χ. αν έρχεται άφιξη νέου task τώρα ή αναχώρηση/εξυπηρέτηση) και να προχωρήσουμε στην επόμενη κατάσταση.
- **Η ανανέωση και αποθήκευση των δικών μας μετρικών** και στατιστικών παραμέτρων που θέλουμε να μελετήσουμε, π.χ. για το χρόνο επεξεργασίας, αναμονής, πλήθος task που εξυπηρετήθηκαν κτλ.

Στην παρούσα άσκηση θα μελετηθεί το πιο απλό σύστημα, δηλαδή:

- δημιουργία τυχαίων νέων tasks
- η τοποθέτησή τους σε μια ουρά με τα νέα tasks
- η εξυπηρέτησή τους σειριακά ένα-ένα από το παλιότερο προς το νεότερο (first in first out) από έναν επεξεργαστή και αναχώριση από το σύστημα

Στις επόμενες εργαστηριακές ασκήσεις θα προσεγγίσουμε πιο σύνθετα συστήματα, αλλάζοντας

- το πλήθος των παράλληλων επεξεργαστών με στόχο την παράλληλη επεξεργασία,
- την πειθαρχία εξυπηρέτησης ουράς, δηλαδή τον τρόπο/σειρά εξυπηρέτησής των νέων tasks με στόχο την εξυπηρέτηση αυτών που έχουν προτεραιότητα, καθώς και
- τις τεχνικές ανάθεσης των tasks στους επεξεργαστές για εφαρμογή τεχνικών load-balancing.

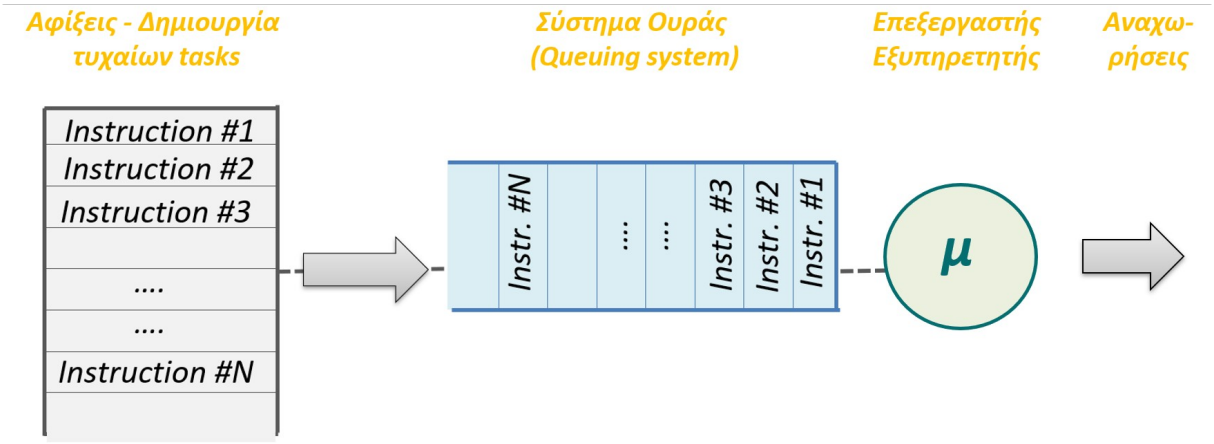


Εικόνα 2 Προσομοίωση ενός συστήματος ουράς με έναν εξυπηρετητή / διακομιστή.

Εστω ότι το σύστημα αυτό παριστάνει τον επεξεργαστή (processor) ενός υπολογιστικού συστήματος. Εστω για παράδειγμα ότι οι εργασίες (jobs) ή προγράμματα φθάνουν στον επεξεργαστή και ζητούν εξυπηρέτηση (επεξεργασία, processing).

- **Οι μεταξύ των αφίξεων χρόνοι των προγραμμάτων *A1, A2,* ..... είναι ανεξάρτητες και όμοια κατανεμημένες τυχαίες μεταβλητές** (independent and identically distributed random variables, IID). Ο όρος “όμοια κατανεμημένες” σημαίνει ότι οι μεταξύ των αφίξεων χρόνοι έχουν την ίδια κατανομή.
- **Ενα πρόγραμμα το οποίο φθάνει και βρίσκει τον εξυπηρέτη (τον επεξεργαστή) ελεύθερο (idle) αρχίζει να εξυπηρετείται (εκτελείται) αμέσως**.
- **Οι χρόνοι εξυπηρέτησης *S1, S2,* .....των διαδοχικών προγραμμάτων είναι IID τυχαίες μεταβλητές** και είναι ανεξάρτητες από τους μεταξύ των αφίξεων χρόνους.
- **Ενα πρόγραμμα το οποίο φθάνει και βρίσκει τον εξυπηρέτη απασχολημένο, μπαίνει στο τέλος της ουράς**. Μόλις ο εξυπηρέτης τελειώσει με την εξυπηρέτηση ενός προγράμματος, διαλέγει ένα πρόγραμμα από αυτά που περιμένουν στην ουρά (εφ’ όσον η ουρά δεν είναι άδεια) σύμφωνα με την αρχή “όποιος φθάνει πρώτος εξυπηρετείται πρώτος” (first-come, first-served, FCFS).

Η προσομοίωση ξεκινάει με άδεια ουρά και ελεύθερο εξυπηρέτη. Κανένα πρόγραμμα δεν έχει φθάσει ακόμη. Στο χρόνο 0 θα αρχίσουμε να περιμένουμε την άφιξη του πρώτου προγράμματος που θα συμβεί μετά από χρόνο *A1*. (*Θα μπορούσαμε βέβαια να υποθέσουμε ότι η πρώτη άφιξη συμβαίνει σε χρόνο 0.*)



Εικόνα 3 Σύστημα ουράς με έναν εξυπηρετητή / διακομιστή και άπειρη ουρά.

Εστω ότι επιθυμούμε να προσομοιώσουμε το **σύστημα αυτό έως ότου τελειώσουν οι καθυστερήσεις (delays) στην ουρά ενός συγκεκριμένου αριθμού προγραμμάτων n,** δηλαδή η προσομοίωση θα σταματήσει όταν το n οστό πρόγραμμα αρχίσει να εξυπηρετείται. Συνεπώς ο χρόνος (time) που τελειώνει η προσομοίωση είναι μια τυχαία μεταβλητή που εξαρτάται από τις παρατηρηθείσες τιμές των τυχαίων μεταβλητών των μεταξύ των αφίξεων χρόνων και των χρόνων εξυπηρέτησης.

Για να μετρήσουμε την απόδοση αυτού του συστήματος, θα βρούμε εκτιμήσεις τριών ποσοτήτων.

- **Πρώτα θα εκτιμήσουμε την αναμενόμενη (expected) μέση καθυστέρηση στην ουρά των n προγραμμάτων**

Τη μέση καθυτέρηση θα τη δηλώσουμε σαν d(n). Ο όρος “αναμενόμενη” στον ορισμό του d(n) σημαίνει το εξής:

• ***Σε ένα τρέξιμο (run) της προσομοίωσης***, η πραγματική παρατηρηθείσα μέση καθυστέρηση των n προγραμμάτων εξαρτάται από τις παρατηρήσεις των τυχαίων μεταβλητών των μεταξύ των αφίξεων χρόνων και των χρόνων εξυπηρέτησης που έτυχε να σημειώσουμε.

• ***Σε ένα άλλο τρέξιμο της προσομοίωσης*** μπορεί να προκύψουν διαφορετικοί χρόνοι αφίξεων και εξυπηρετήσεων οπότε και θα προκύψει διαφορετική τιμή για τη μέση καθυστέρηση των n προγραμμάτων.

Επομένως η μέση καθυστέρηση για ένα τρέξιμο της προσομοίωσης θεωρείται και αυτή ως τυχαία μεταβλητή. Αυτό που ζητάμε να εκτιμήσουμε, δηλαδή το d(n), είναι η αναμενόμενη τιμή αυτής της τυχαίας μεταβλητής. Θα μπορούσαμε να πούμε ότι d(n) είναι ο μέσος όρος από έναν μεγάλο αριθμό μέσων καθυστερήσεων n προγραμμάτων. Εστω ότι από ένα τρέξιμο της προσομοίωσης προκύπτουν οι καθυστερήσεις προγραμμάτων D1, D2, ..., Dn. Ενας προφανής εκτιμητής του d(n) είναι

$$\hat{d}(n) = \frac{\sum_{i=1}^{n} D_i}{n}$$

ο οποίος είναι ο μέσος όρος των n καθυστερήσεων Di που παρατηρήθηκαν στην προσομοίωση. Θα πρέπει να σημειωθεί ότι οι καθυστερήσεις κάποιων προγραμμάτων μπορεί να είναι ίσες με μηδέν. Αυτό ισχύει όταν ένα πρόγραμμα τη στιγμή της άφιξής του βρίσκει τον επεξεργαστή ελεύθερο και τιν ουρά άδεια. Τότε ο εκτιμητής βασίζεται σε ένα δείγμα το οποίο στην περίπτωση αυτή είναι ένα σύνολο τρεξιμάτων προσομοίωσης μεγέθους 1.

- **Ενα δεύτερο μέτρο της απόδοσης του συστήματος είναι ο αναμενόμενος μέσος αριθμός προγραμμάτων που περιμένουν εξυπηρέτηση στην ουρά**

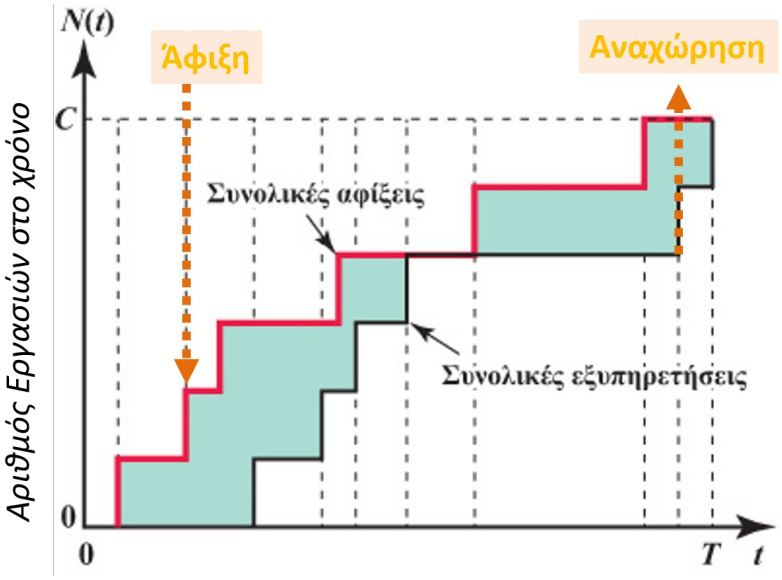
Το μέσο αριθμό προγραμμάτων που περιμένουν εξυπηρέτηση τον δηλώνουμε συνήθως ως q(n). Το n είναι απαραίτητο στη δήλωση αυτή, γιατί δείχνει ότι αυτός ο “μέσος όρος” υπολογίζεται κατά τη διάρκεια του χρόνου που χρειάζεται να παρατηρηθούν οι n καθυστερήσεις που ορίζουν τη συνθήκη τέλους.

- **Το τρίτο μέτρο της απόδοσης του συστήματος αφορά το πόσο απασχολημένος είναι ο εξυπηρέτης.**

Η αναμενόμενη χρησιμοποίηση (utilization) του εξυπηρέτη είναι το αναμενόμενο τμήμα του χρόνου κατά τη διάρκεια της προσομοίωσης (από 0 μέχρι T(n)) που ο εξυπηρέτης είναι απασχολημένος. Η χρησιμοποίηση είναι προφανώς ένας αριθμός μεταξύ 0 και 1 και τη συμβολίζουμε με u(n). Από ένα τρέξιμο της προσομοίωσης, εκτιμητής του u(n) ισούται με το παρατηρηθέν τμήμα του χρόνου κατά τη διάρκεια της προσομοίωσης που ο εξυπηρέτης είναι απασχολημένος.

Το u(n) μπορεί να υπολογιστεί από την προσομοίωση σημειώνοντας τους χρόνους κατά τους οποίους ο εξυπηρέτης αλλάζει κατάσταση και κάνοντας τις απαραίτητες αφαιρέσεις και τη διαίρεση.

Η εικόνα παρακάτω παριστάνει μια ενδεικτική χρονική εξέλιξη ενός συστήματος ουράς επεξεργαστή διακομιστή, υποδεικνύοντας την άφιξη μιας εντολής (task) στο σύστημα, την αναχώρηση ενός task από το σύστημα καθώς και τον τρόπο υπολογισμού των συνολικών αφίξεων/εξυπηρετήσεων και πόσα task βρίσκονται στο σύστημα χωρίς να έχουν εξυπηρετηθεί ακόμα.



Εικόνα 4 Άφιξη και ολοκλήρωση εργασιών

Οι μεταξύ των αφίξεων χρόνοι και οι χρόνοι εξυπηρέτησης θα μοντελοποιηθούν σαν ανεξάρτητες τυχαίες μεταβλητές από ***εκθετικές κατανομές με μέση τιμή λ για τους μεταξύ των αφίξεων χρόνους*** και ***μέση τιμή μ για τους χρόνους εξυπηρέτησης***.

Στην παρούσα εργασία, ζητείται να υλοποιηθεί το σύστημα προσομοίωσης ενός επεξεργαστή με ρυθμό επεξεργασίας μ=6 εκθετικής κατανομής στον οποίο φτάνουν εργασίες με ρυθμό άφιξης λ=5 εκθετικής και να μελετηθούν τα διαγράμματα που απεικονίζουν το σύστημα, καθώς και κάποια μετρικά, όπως χρόνος αναμονής, ο αναμενόμενος μέσος αριθμός που βρίσκονται στην ουρά, το ποσοστό χρησιμοποίοησης (utilization) του διακομιστή κτλ.

1η σειρά από ερωτήματα προσομοίωσης ενός επεξεργαστή :

1) Με βάση τις γνώσεις του προηγούμενου εργαστηρίου, να δημιουργήσετε μια λίστα τυχαίων παρατηρήσεων-αφίξεων tasks που δημιουργούνται για παράδειγμα σε ένα σύστημα διακομιστή. Η δημιουργία αυτών των γεγονότων να ακολουθεί εκθετική μέση τιμή με ρυθμό άφιξης λ=5 και ο χρόνος προσομοίωσης μπορεί να είναι σχετικά μικρός π.χ. 2, για χάριν απλότητας. Να δημιουργήσετε ένα μονοδιάστατο διάνυσμα που καταχωρεί το πλήθος των συνολικών αφίξεων που έχουν δημιουργηθεί στο σύστημα και έπειτα να απεικονίσετε τις αφίξεις κατά τη διάρκεια εξέλιξης του χρόνου προσομοίωσης.

ΑΠΑΝΤΗΣΗ:

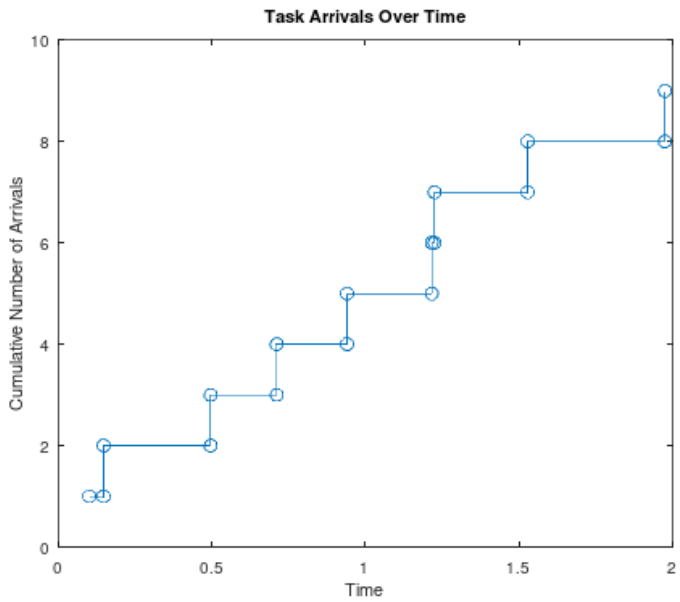
```
lambda = 5;
simulation_time = 2;
N = 1000;

interarrival_times = exprnd(1/lambda, 1, N);

arrival_times = cumsum(interarrival_times);

arrival_times = arrival_times(arrival_times <= simulation_time);

subplot(2, 2, 1);
stairs(arrival_times, 1:length(arrival_times), 'o-');
title('Task Arrivals Over Time');
xlabel('Time');
ylabel('Cumulative Number of Arrivals');
```



2) Να δημιουργήσετε μια λίστα τυχαίων παρατηρήσεων-αναχωρήσεων για την εξυπηρέτηση ενός task από το σύστημα του διακομιστή που να ακολουθεί εκθετική μέση τιμή με ρυθμό αναχώρησης μ=6.

ΠΡΟΣΟΧΗ: Πλέον οι αναχωρήσεις δεν είναι ανεξάρτητα γεγονότα για να μπουν σε ένα διάνυσμα αυθαίρετα ΑΛΛΑ μπορούν να εμφανιστούν προφανώς μόνο μετά από αντίστοιχο αριθμό αφίξεων στο σύστημα. Με άλλα λόγια η πρώτη αναχώρηση στο χρόνο $t_{departure}$ μπορεί να εμφανιστεί μόνο μετά από την πρώτη άφιξη $t_{arrival}$. Αντίστοιχα, η δεύτερη αναχώρηση μετά τη δεύτερη άφιξη κοκ., οπότε χρειάζεται να υλοποιήσετε πρώτα ένα μονοδιάστατο διάνυσμα στο χρόνο που ελέγχει και καταχωρεί αν το επόμενο πλησιέστερο χρονικά γεγονός που συμβεί στο μέλλον είναι αναχώρηση ή άφιξη.

Έπειτα να απεικονίσετε σε ένα γράφημα τις αναχωρήσεις / εξυπηρετήσεις που συμβαίνουν ως προς το χρόνο.

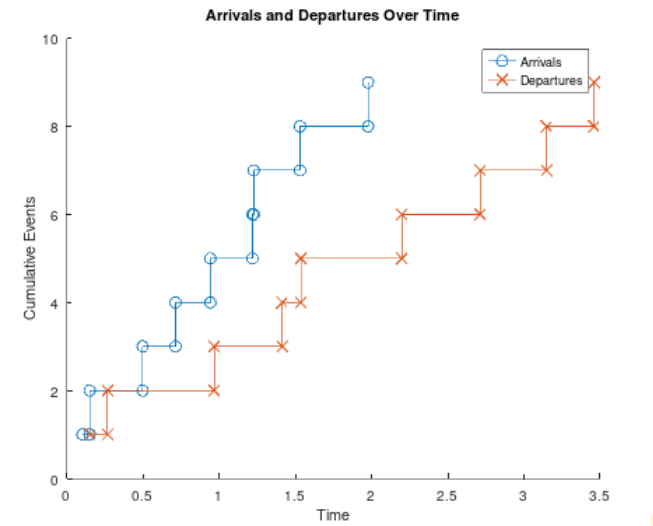
ΑΠΑΝΤΗΣΗ:

```
mu = 6;

service_times = exprnd(1/mu, 1, length(arrival_times));

departure_times = zeros(1, length(arrival_times));
for i = 1:length(arrival_times)
   if i == 1
      departure_times(i) = arrival_times(i) + service_times(i);
   else
      departure_times(i) = max(arrival_times(i), departure_times(i-1)) + service_times(i);
   end
end

subplot(2, 2, 2);
hold on;
stairs(arrival_times, 1:length(arrival_times), 'o-', 'DisplayName', 'Arrivals');
stairs(departure_times, 1:length(departure_times), 'x-', 'DisplayName', 'Departures');
title('Arrivals and Departures Over Time');
xlabel('Time');
ylabel('Cumulative Events');
legend;
hold off;
```



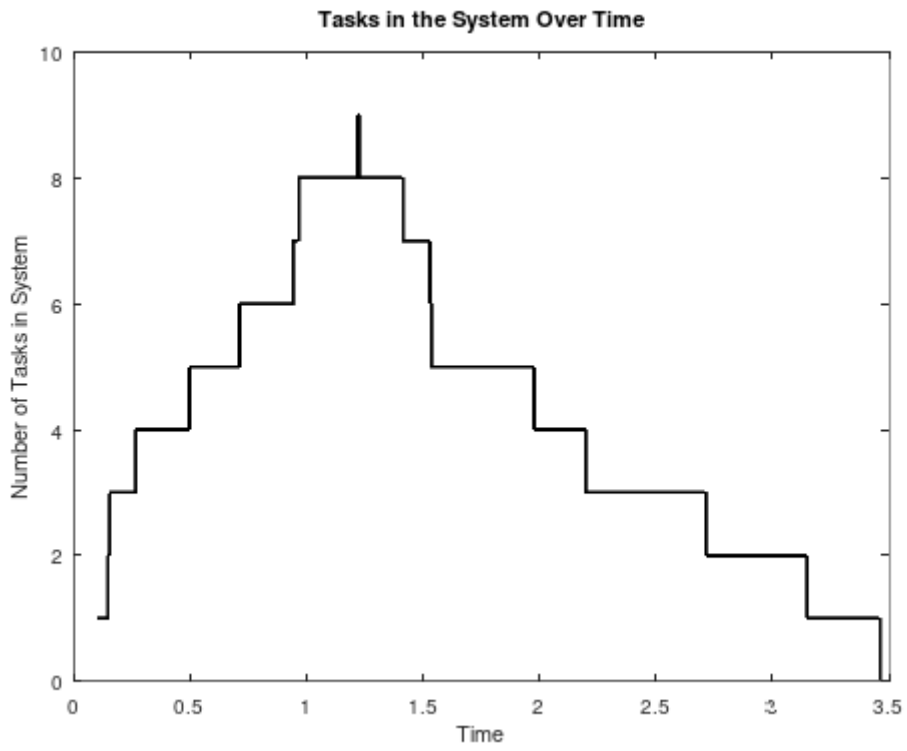
3) Να συνδυάσετε τις αφίξεις και τις αναχωρήσεις για να απεικονίσετε τα tasks που βρίσκονται στο σύστημα ουράς του επεξεργαστή κατά τη διάρκεια εξέλιξης του χρόνου προσομοίωσης.

ΑΠΑΝΤΗΣΗ:

```
events = sort([arrival_times, departure_times]);
event_type = [ones(1, length(arrival_times)), -ones(1, length(departure_times))]; % 1 for
arrival, -1 for departure
[event_timeline, idx] = sort(events);
event_type = event_type(idx);

tasks_in_system = cumsum(event_type);

subplot(2, 2, [3, 4]);
stairs(event_timeline, tasks_in_system, 'k-', 'LineWidth', 1.5);
title('Tasks in the System Over Time');
xlabel('Time');
ylabel('Number of Tasks in System');
```



4) Να δημιουργήσετε ένας μετρικό / μεταβλητή που να αυτοματοποιεί και να εκτιμά με βάση την προσομοίωσή σας:
   a. Το μέσο χρόνο αναμονής στην ουρά των tasks
   b. Τον συνολικό-αθροιστικό αριθμό αφίξεων νέων tasks στο σύστημα
   c. Το συνολικό-αθροιστικό αριθμό αναχωρήσεων-εξυπηρετήσεων των tasks στο σύστημα.
   d. Το μέσο συντελεστή χρησιμοποίησης του επεξεργαστή που δείχνει τι ποσοστό του χρόνου ο επεξεργαστής ήταν κατειλημμένος.

ΑΠΑΝΤΗΣΗ:

```
waiting_times = departure_times - arrival_times - service_times;
avg_waiting_time = mean(waiting_times);

total_arrivals = length(arrival_times);

total_departures = length(departure_times);

total_service_time = sum(service_times);
processor_utilization = total_service_time / simulation_time;

fprintf('Simulation Time %d, λ = %.1f, μ = %.1f\n', simulation_time, lambda, mu);
fprintf('  Average Waiting Time: %.2f\n', avg_waiting_time);
fprintf('  Total Arrivals: %d\n', total_arrivals);
fprintf('  Total Departures: %d\n', total_departures);
fprintf('  Processor Utilization: %.2f\n', processor_utilization);
```

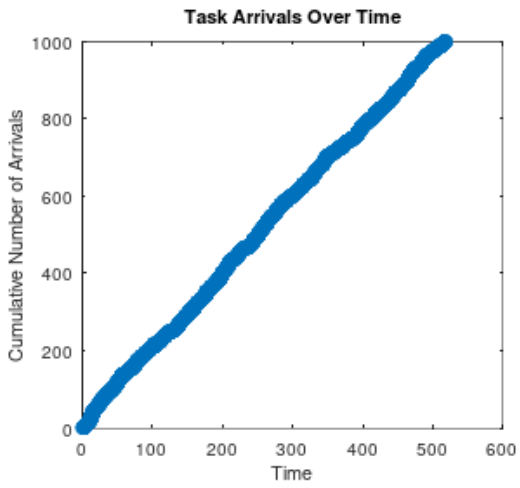
```
Simulation Time 2, λ = 5.0, μ = 6.0
  Average Waiting Time: 0.49
  Total Arrivals: 9
  Total Departures: 9
  Processor Utilization: 1.56
>>
```

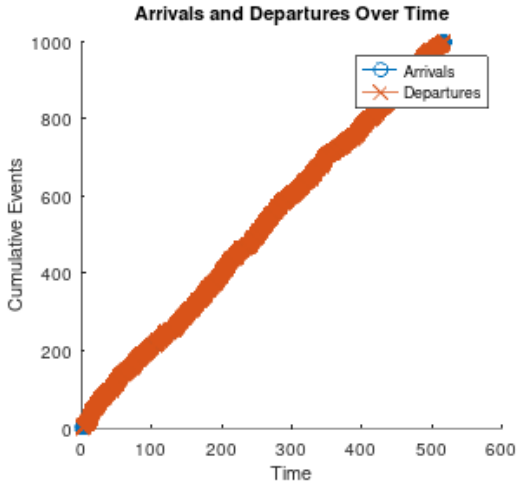
2η σειρά από ερωτήματα προσομοίωσης ενός επεξεργαστή άπειρης ουράς :

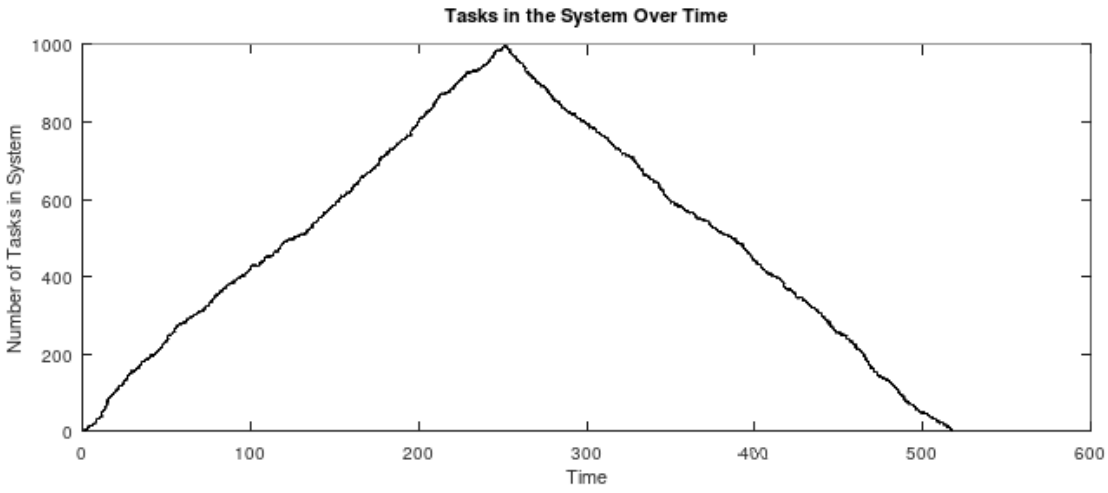
Εφόσον έχετε αναπτύξει την παραπάνω προσομοίωση ενός συστήματος ουράς, να αυξήσετε το χρόνο προσομοίωσης σε 1000 (αντί για 1 ή 2) και να εκτελέσετε τις παρακάτω προσομοιώσεις με τις μέσες τιμές άφιξης - αναχώρησης που ζητώνται, παράγοντας κάθε φορά τα αποτελέσματα-απαντήσεις των τεσσάρων προηγούμενων ΥΠΟ ερωτημάτων.

1) Για μέσο χρόνο άφιξης λ = 2 και αναχώρησης μ = 6

ΑΠΑΝΤΗΣΗ:



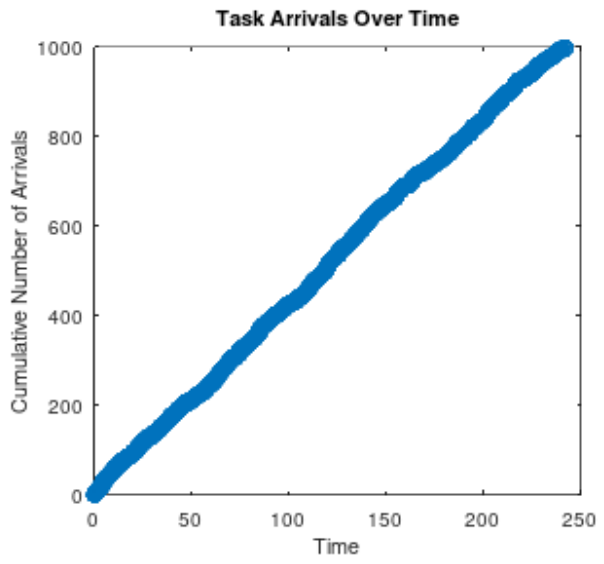


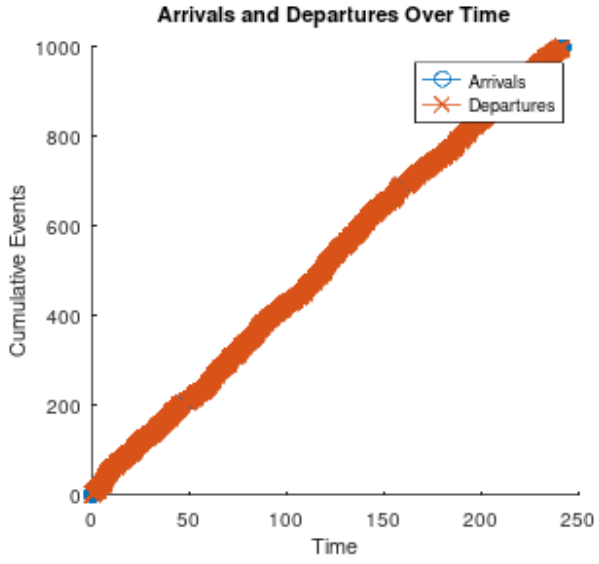


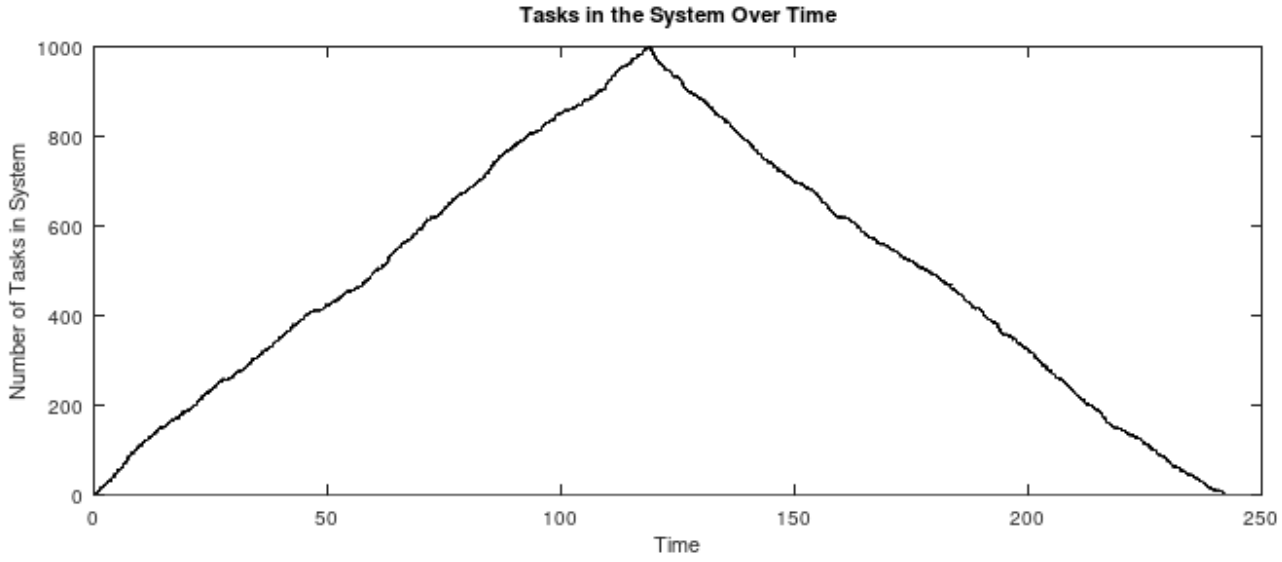

```
Simulation Time 1000, λ = 2.0, μ = 6.0
  Average Waiting Time: 0.08
  Total Arrivals: 1000
  Total Departures: 1000
  Processor Utilization: 0.17
>>
```

2) Για μέσο χρόνο άφιξης λ = 4 και αναχώρησης μ = 6

ΑΠΑΝΤΗΣΗ:



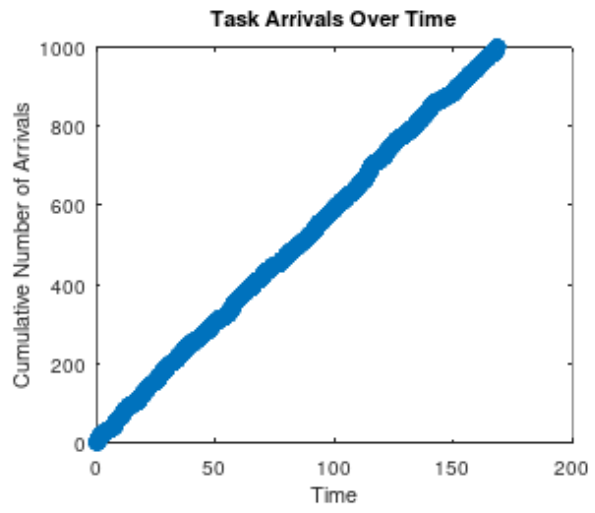


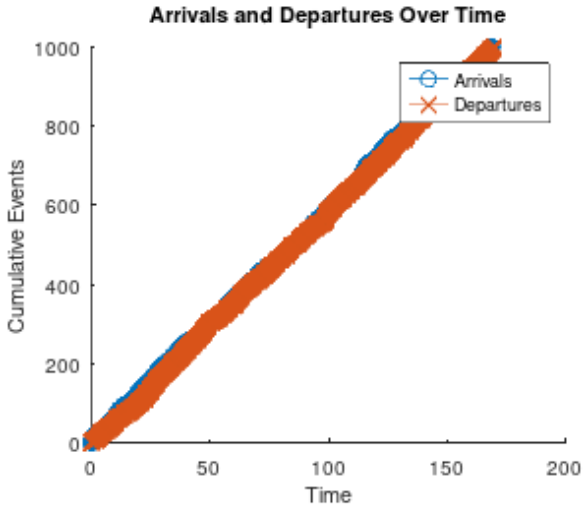


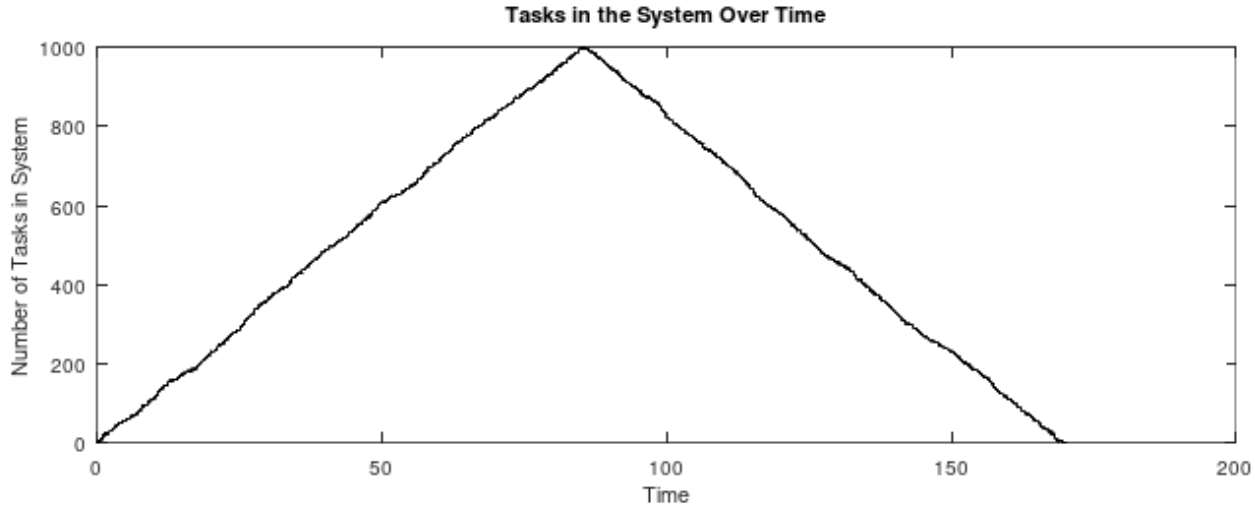

```
Simulation Time 1000, λ = 4.0, μ = 6.0
  Average Waiting Time: 0.32
  Total Arrivals: 1000
  Total Departures: 1000
  Processor Utilization: 0.16
>>
```

3) Για μέσο χρόνο άφιξης λ = 5.9 και αναχώρησης μ = 6

ΑΠΑΝΤΗΣΗ:



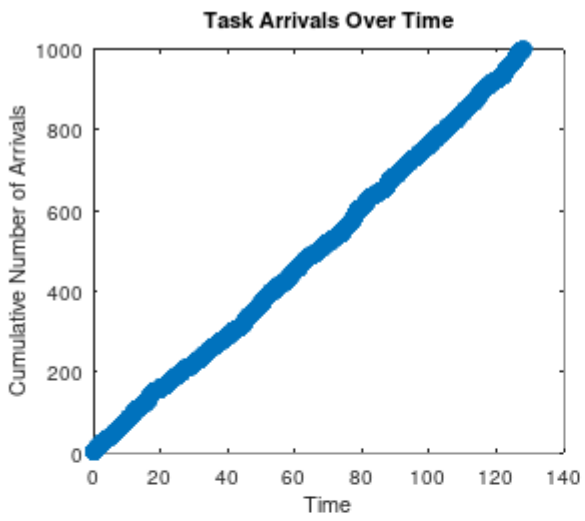


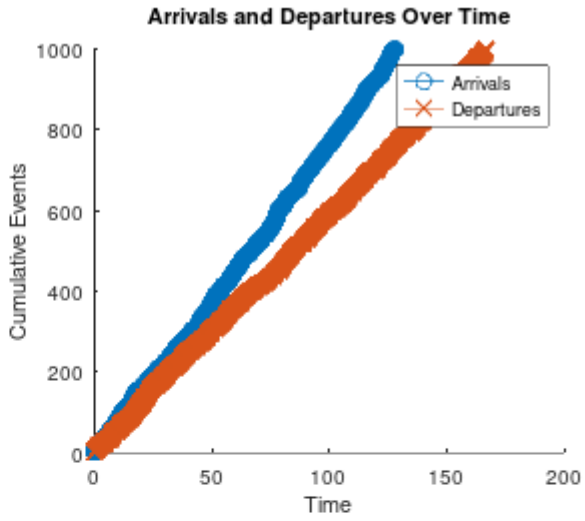


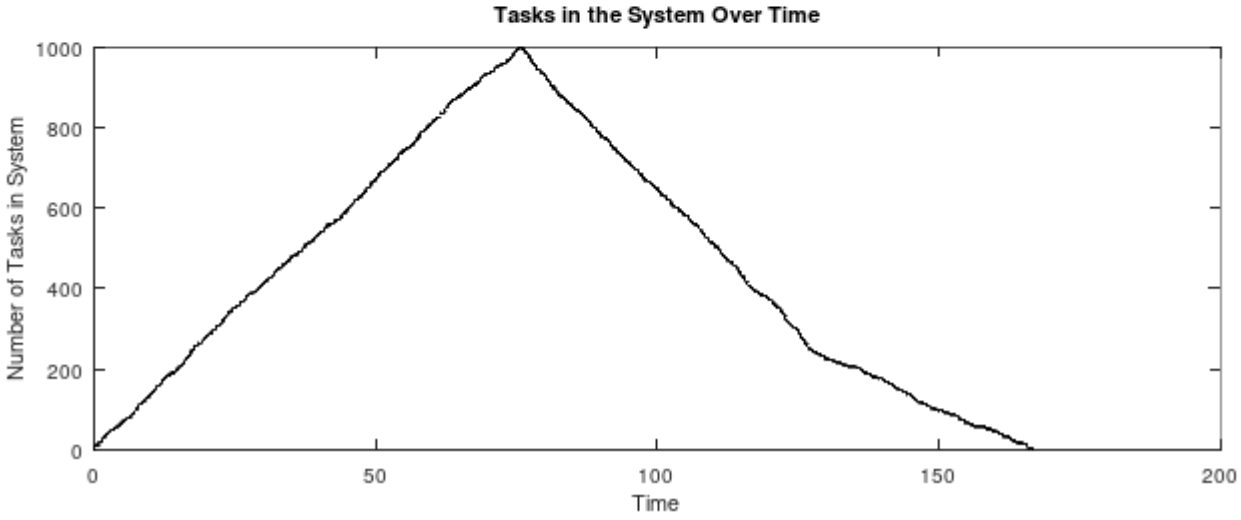

```
Simulation Time 1000, λ = 5.9, μ = 6.0
  Average Waiting Time: 1.98
  Total Arrivals: 1000
  Total Departures: 1000
  Processor Utilization: 0.17
>>
```

4) Για μέσο χρόνο άφιξης λ = 8 και αναχώρησης μ = 6

ΑΠΑΝΤΗΣΗ:



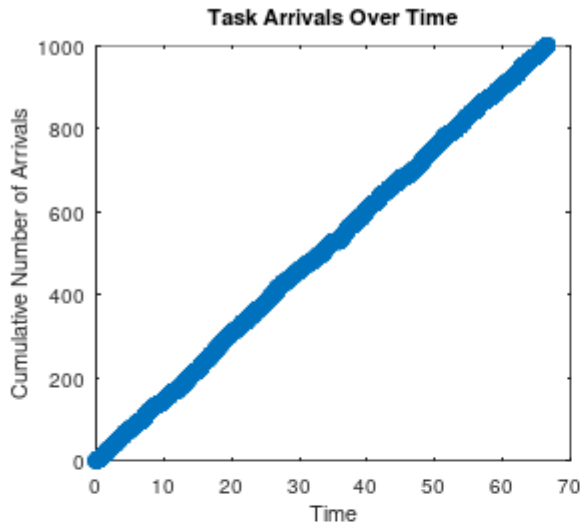


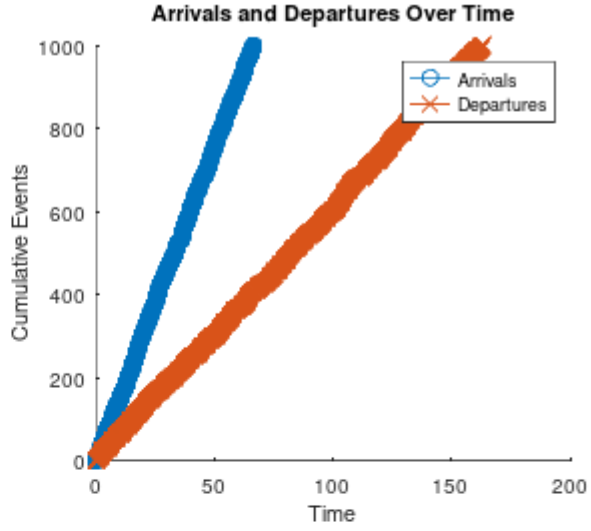


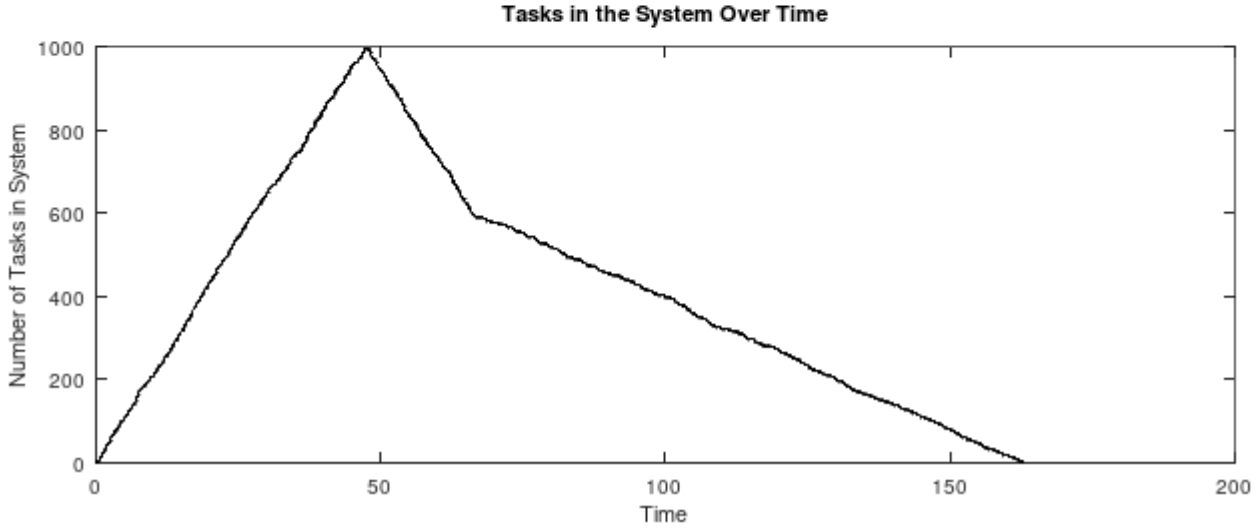

```
Simulation Time 1000, λ = 8.0, μ = 6.0
  Average Waiting Time: 18.32
  Total Arrivals: 1000
  Total Departures: 1000
  Processor Utilization: 0.17
>>
```

5) Για μέσο χρόνο άφιξης λ = 15 και αναχώρησης μ = 6

ΑΠΑΝΤΗΣΗ:








```
Simulation Time 1000, λ = 15.0, μ = 6.0
  Average Waiting Time: 48.52
  Total Arrivals: 1000
  Total Departures: 1000
  Processor Utilization: 0.16
>>
```

6) Να συγκρίνετε τα αποτελέσματα των προσομοιώσεων για διαφορετικούς ρυθμούς άφιξης λ και να σχολιάσετε τι παρατηρείτε ως προς το
   a. Το μέσο χρόνο αναμονής στην ουρά των tasks
   b. Τον συνολικό-αθροιστικό αριθμό αφίξεων νέων tasks στο σύστημα
   c. Το συνολικό-αθροιστικό αριθμό αναχωρήσεων-εξυπηρετήσεων των tasks στο σύστημα.
   d. Το μέσο συντελεστή χρησιμοποίησης του επεξεργαστή - ποσοστό utilization.

ΑΠΑΝΤΗΣΗ:

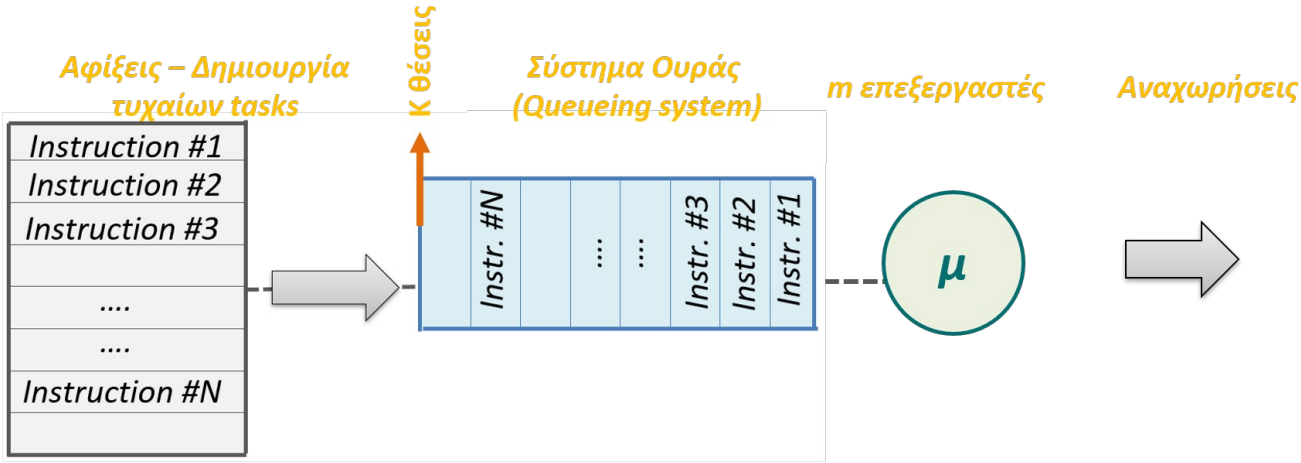
a. Ο μέσος χρόνος αναμονής αυξάνεται όσο ο ρυθμός άφιξης πλησιάζει (και ξεπερνάει) τον ρυθμό εξυπηρέτησης όπως και θα περιμέναμε

b, c, d. Ο αριθμός αφίξεων/αναχωρήσεων καθώς και το utilization του επεξεργαστή παραμένουν σχεδόν, αν όχι εντελώς, ίδια ανεξάρτητα από τον ρυθμό άφιξης κάτι που πιθανώς οφείλεται σε λανθασμένη υλοποίηση.

3η σειρά από ερωτήματα προσομοίωσης επεξεργαστή πεπερασμένης ουράς:

Η παρούσα σειρά ερωτημάτων αποτελεί επανάληψη της σειράς ερωτημάτων 2, αλλά αυτή τη φορά για ουρά MM1 με πεπερασμένο αριθμό K = 10 θέσεων μόνο.

Αυτό σημαίνει ότι για κάθε νέα άφιξη στο σύστημα, ο προσομοιωτής (του συστήματος διαχείρισης των tasks) **θα ελέγχει πρώτα πόσα tasks είναι αποθηκευμένα στην ουρά** και δεν έχουν εξυπηρετηθεί. Στην περίπτωση που υπάρχουν ήδη K = 10 αποθηκευμένες διεργασίες, και η ουρά είναι πεπερασμένη και γεμάτη, **η άφιξη του νέου task-γεγονότος θα απορρίπτεται (θα γίνεται dropped)** και δεν θα μπαίνει στην ουρά προς εξυπηρέτηση. Αυτό φυσικά γίνεται γιατί στην ρεαλιστική ζωή, όλα τα συστήματα έχουν περιορισμένη μνήμη, δεν μπορούν να αποθηκεύουν επ' άπειρον, όπως για παράδειγμα και σε μία τράπεζα υπάρχει συγκεκριμένος αριθμός καρεκλών για την αναμονή των πελατών.



Εικόνα 5 Σύστημα ουράς με έναν εξυπηρετητή / διακομιστή και πεπερασμένη ουρά K-θέσεων μνήμης.

Εφόσον έχετε αναπτύξει την παραπάνω προσομοίωση ενός συστήματος άπειρης ουράς, να την **τροποποιήσετε κατάλληλα, βάζοντας το όριο των K-θέσεων στην ουρά και ένα επιπλέον μετρικό των tasks** που έχουν απορριφθεί και δεν εξυπηρετήθηκαν από το σύστημα.

Να κρατήσετε το χρόνο της προσομοίωσης σταθερό, δηλαδή 1000 (όσο το είχατε στη 2η σειρά ερωτημάτων) και να εκτελέσετε τις παρακάτω προσομοιώσεις με τις μέσες τιμές άφιξης - αναχώρησης που ζητώνται (και είναι ίδιες με τη 2η σειρά ερωτημάτων). Θα πρέπει ουσιαστικά να παράξετε τα ίδια αποτελέσματα με την προηγούμενη προσομοίωση της 2ης σειράς, για να μπορέσουν να συγκριθούν στο τέλος. Να αποθηκεύετε κάθε φορά τα αποτελέσματα-απαντήσεις των τεσσάρων προηγούμενων υπο ερωτημάτων, μαζί πλέον και την πέμπτη παράμετρο των dropped tasks, δηλαδή να εκτιμάτε με βάση την προσομοίωσή σας κάθε φορά:

a. Το μέσο χρόνο αναμονής στην ουρά των tasks
b. Τον συνολικό-αθροιστικό αριθμό αφίξεων νέων tasks στο σύστημα

c. Το συνολικό-αθροιστικό αριθμό αναχωρήσεων-εξυπηρετήσεων των tasks στο σύστημα.
d. Το μέσο συντελεστή χρησιμοποίησης του επεξεργαστή που δείχνει τι ποσοστό του χρόνου ο επεξεργαστής ήταν κατειλημμένος.
e. Το πλήθος των tasks που έχουν γίνει dropped από την ουρά και δεν έχουν εξυπηρετηθεί από το σύστημα.

Για το νέο σύστημα με K=10 θέσεις μνήμης να καταγράψετε τα αποτελέσματα για τις παρακάτω περιπτώσεις:

1) Για μέσο χρόνο άφιξης λ = 2 και αναχώρησης μ = 6

ΑΠΑΝΤΗΣΗ:

```
queue_capacity = 10;
queue_length = 0;
dropped_tasks = 0;

departure_times = [];
for i = 1:length(arrival_times)
   if queue_length < queue_capacity
      % Accept the task
      if i == 1
         departure_times(i) = arrival_times(i) + service_times(i);
      else
         departure_times(i) = max(arrival_times(i), departure_times(i-1)) + service_times(i);
      end
      queue_length = queue_length + 1;
   else
      % Drop the task
      dropped_tasks = dropped_tasks + 1;
   end
   queue_length = max(0, queue_length - 1); % Update queue after departure
end

fprintf('Dropped Tasks: %d\n', dropped_tasks);
```

```
Simulation Time 1000, λ = 2.0, μ = 6.0
  Average Waiting Time: 0.08
  Total Arrivals: 1000
  Total Departures: 1000
  Processor Utilization: 0.16
>> ask2_3

Dropped Tasks: 0
>>
```

2) Για μέσο χρόνο άφιξης λ = 4 και αναχώρησης μ = 6

ΑΠΑΝΤΗΣΗ:

```
Simulation Time 1000, λ = 4.0, μ = 6.0
  Average Waiting Time: 0.28
  Total Arrivals: 1000
  Total Departures: 1000
  Processor Utilization: 0.16
>> ask2_3

Dropped Tasks: 0
>>
```

3) Για μέσο χρόνο άφιξης λ = 5.9 και αναχώρησης μ = 6

ΑΠΑΝΤΗΣΗ:

```
Simulation Time 1000, λ = 5.9, μ = 6.0
  Average Waiting Time: 6.35
  Total Arrivals: 1000
  Total Departures: 1000
  Processor Utilization: 0.17
>> ask2_3

Dropped Tasks: 0
```

4) Για μέσο χρόνο άφιξης λ = 8 και αναχώρησης μ = 6

ΑΠΑΝΤΗΣΗ:

```
Simulation Time 1000, λ = 8.0, μ = 6.0
  Average Waiting Time: 18.74
  Total Arrivals: 1000
  Total Departures: 1000
  Processor Utilization: 0.16
>> ask2_3

Dropped Tasks: 0
```

5) Για μέσο χρόνο άφιξης λ = 15 και αναχώρησης μ = 6

ΑΠΑΝΤΗΣΗ:

```
Simulation Time 1000, λ = 15.0, μ = 6.0
  Average Waiting Time: 53.47
  Total Arrivals: 1000
  Total Departures: 1000
  Processor Utilization: 0.17
>> ask2_3

Dropped Tasks: 0
```

6) Να συγκρίνετε τα αποτελέσματα των προσομοιώσεων της σειράς 2 και της σειράς 3 για διαφορετικούς ρυθμούς άφιξης λ και να σχολιάσετε τι παρατηρείτε ως προς το
   a. Το μέσο χρόνο αναμονής στην ουρά των tasks.
   b. Τον συνολικό-αθροιστικό αριθμό αφίξεων νέων tasks στο σύστημα.
   c. Το συνολικό-αθροιστικό αριθμό αναχωρήσεων-εξυπηρετήσεων των tasks στο σύστημα.
   d. Το μέσο συντελεστή χρησιμοποίησης του επεξεργαστή - ποσοστό utilization.

   Και
   e. Το πλήθος των tasks που έχουν γίνει dropped

ΤΕΛΟΣ: Να απαντήσετε με βάση τα αποτελέσματά σας για το χρόνο αναμονής/εξυπηρέτησης και το πλήθος των tasks που γίνονται dropped ως προς την τελευταία περίπτωση εκτέλεσης (λ=15, μ=6), τι θεωρείτε προτιμότερο: Αξίζει να υπάρχει ουρά πεπερασμένων θέσεων K=10 ή όχι ;

ΑΠΑΝΤΗΣΗ:

a. Παρατηρούμε ότι σε σχέση με τα αποτελέσματα της 2ης σειράς ερωτημάτων, ο χρόνος αναμονής είναι αυξημένος αλλά αυτό οφείλεται στην τυχαιότητα της δημιουργίας των δεδομένων.

Τα b, c, d παραμένουν ίδια με τα ερωτήματα της 2ης σειράς.

e. Σε όλες τις περιπτώσεις γίνονται 0 tasks drop και αυτό, όπως και στην 2η σειρά ερωτημάτων, οφείλεται σε λανθασμένη υλοποίηση.

Παρόλο που τα αποτελέσματα δεν επαρκούν για να βγει συμπέρασμα με σιγουριά, θεωρώ ότι η ουρά 10 θέσεων δεν θα είναι αποτελεσματική για την περίπτωση (λ=15, μ=6) (όπως και για τις περισσότερες περιπτώσεις) καθώς γεμίζει πολύ γρήγορα με αποτέλεσμα να γίνονται drop πολλά από τα tasks.

ΠΑΡΑΔΟΣΗ ΕΡΓΑΣΙΑΣ

Παράδοση εργαστηριακής άσκησης πάντα μόνο μέσω eclass και με χρήση του παρόντος απαντητικού φύλλου

Πηγές:

[1] Διαλέξεις των προπτυχιακών και μεταπτυχιακών μαθημάτων *«Απόδοση Παράλληλων Συστημάτων»* και *«Μοντελοποίηση και Προσομοίωση*, ακαδημαϊκά έτη 2010-2012, του Τμ. Πληροφορικής Α.Π.Θ. της καθηγ. Ελένης Καρατζά.

[2] *"Ανάλυση Επίδοσης Υπολογιστικών Συστημάτων: Αναλυτικά μποντέλα, προσομοίωση, μετρήσεις*", Α.Γ. Σταφυλοπάτης, Γ. Σιόλας, Ελληνικά Ακαδημαϊκά Ηλεκτρονικά Συγγραματα και Βοηθήματα, 2015, της Δράσης των Ανοικτών Ακαδημαϊκών Ηλεκτρονικών Συγγραμμάτων - Αποθετήριο Κάλλιπος, Υπερ-σύνδεσμος https://repository.kallipos.gr/handle/11419/6055