ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ 2014-15
1η Ατομική Άσκηση: Προσομοίωση Ουράς Τράπεζας

Να επεκτείνετε το πρόγραμμα προσομοίωσης της ουράς πελατών μιας Τράπεζας που συνοδεύει την άσκηση με νέες δυνατότητες και λειτουργικότητα ως ακολούθως

1. Να υλοποιήσετε τον ΑΤΔ Ουρά χρησιμοποιώντας μετρητή (η $3^{η}$ υλοποίηση στις διαφάνειες). Ελέγξτε την νέα υλοποίηση με πρόγραμμα δοκιμής.
2. Να προσθέσετε στην υλοποίηση του ΑΤΔ Ουρά την πράξη
   ```
   int SizeOuras(TOuras oura);
   ```
   που επιστρέφει τον αριθμό στοιχείων σε μια ουρά. Ελέγξτε την υλοποίηση με πρόγραμμα δοκιμής. Επεκτείνετε το κυρίως πρόγραμμα, ώστε να εκτυπώνει τον αριθμό πελατών στην ουρά που δεν εξυπηρετήθηκαν.
3. Να επεκτείνετε τον TStoixeiouOuras, ώστε το struct να περιλαμβάνει όχι μόνο το λεπτό εισόδου (όπως στην υλοποίηση που σας δίδεται), αλλά επίσης τον χρόνο εξυπηρέτησης του πελάτη (int XronosEksipiretisis για την ώρα σταθερός). Επίσης να προσθέσετε δυο συναρτήσεις
   ```
   void PelatisSetXronoEksipiretisis(TPelatis *stoixeioPtr, int duration);
   int PelatisGetXronoEksipiretisis(TPelatis *stoixeioPtr);
   ```
   Η πρώτη ενημερώνει τον χρόνο Εξυπηρέτησης και η δεύτερη επιστρέφει τον χρόνο εξυπηρέτησης. Ελέγξτε την υλοποίησή σας με πρόγραμμα δοκιμής. Κατόπιν επεκτείνετε το κυρίως πρόγραμμα, ώστε αντί ο χρόνος εξυπηρέτησης ενός πελάτη να είναι σταθερός να είναι μεταξύ 1 και ενός Μέγιστου που να ζητείται ως είσοδος στο πρόγραμμα αντί του σταθερού χρόνου. Σημειώνουμε ότι η συνάρτηση rand( ) επιστρέφει τυχαίο ακέραιο μεταξύ 0 και MAX_RAND, rand( ) % X επιστρέφει τυχαίο μεταξύ 0 και X-1.
4. Να επεκτείνετε το κυρίως πρόγραμμα, ώστε να δέχεται τον χρόνο που κλείνει η τράπεζα (μετά τον οποίο δεν δέχεται άλλους πελάτες). Όμως οι υπάρχοντες πελάτες στην ουρά να εξυπηρετούνται όλοι. Να εκτυπώσετε τον πραγματικό τελικό χρόνο λειτουργίας και πόσα επιπλέον λεπτά ήταν απαραίτητα.
5. Να σχεδιάσετε και να αναπτύξετε μια νέα ενότητα για τον ταμία (tamias.h και tamias.c), που να ορίζει (struct TTamias) και να χειρίζεται α) τον χρόνο που ήταν απασχολημένος (TimeBusy), β) τον χρόνο που ήταν αδρανής (TimeInactive), γ) τον αριθμό πελατών που εξυπηρέτησε (ArithmoPelaton) και δ) τον εναπομείναντα χρόνο (enapomenonXronos) για να ολοκληρώσει την εξυπηρέτηση ενός πελάτη. Θα χρειαστείτε τις συναρτήσεις
   ```
   void TamiasDimiourgia(TTamias *tamias);                 //αρχικοποιεί τα μέλη του struct ταμία

   void TamiasNewCustomer(TTamias *tamias);
                                  // αυξάνει τον μετρητή πελατών και προσθέτει 1 λεπτό απασχόλησης
   void TamiasSetXrono(TTamias *tamias, int Duration);     // αρχικοποιεί εναπομείναντα χρόνο
   void TamiasNoWork(TTamias *tamias);                     // αυξάνει χρόνο αδράνειας
   void TamiasBusy(TTamias *tamias);                       // αυξάνει χρόνο απασχόλησης
   int TamiasFree(TTamias tamias);                         // ελέγχει αν είναι διαθέσιμος

   int TamiasGetArithmosPelatwn(TTamias *tamias);          // επιστρέφει αριθμό πελατών
   int TamiasGetEnapomenonXronos(TTamias *tamias);         // επιστρέφει εναπομείναντα χρόνο
   int TamiasGetInactiveXronos(TTamias *tamias);           // επιστρέφει χρόνο αδράνειας
   int TamiasGetBusyXronos(TTamias *tamias);               // επιστρέφει χρόνο απασχόλησης
   ```
   Αλλάξτε το κυρίως πρόγραμμα, ώστε να χρησιμοποιεί αποκλειστικά τον ανωτέρω τύπο του Ταμία (μέσω των συναρτήσεων). Δεν χρειάζεται αλλαγή δομής, αλλά αντί για την χρήση μεταβλητών προγράμματος να κάνετε χρήση των πράξεων της ενότητας ταμία.
6. Να επεκτείνετε το κυρίως πρόγραμμα προσομοίωσης ώστε να υποστηρίζει πολλούς (πίνακας) ταμίες (οι πελάτες όπως πριν σε μια κοινή ουρά). Στο τέλος της προσομοίωσης να εκτυπώνει για κάθε ταμία τον αριθμό πελατών που εξυπηρέτησε, τον χρόνο που ήταν απασχολημένος και τον χρόνο που είναι αδρανής.
7. Για επιπλέον Bonus 10%. Σχεδιάστε και αναπτύξτε μια δική σας πολιτική διαχείρισης πελατών/ταμιών, π.χ. να μην εργάζονται όλοι οι ταμίες αν δεν υπάρχουν αρκετοί πελάτες ή να βάζουμε επιπλέον ταμία όταν μεγαλώνει πολύ η ουρά και να βγάζουμε ταμία όταν μικραίνει, κλπ.

Οδηγίες Σχεδίασης και Ανάπτυξης Προγράμματος
Το πρόγραμμά σας πρέπει να είναι οργανωμένο σε ενότητες (modules) και σε πρόγραμμα-πελάτη. Το πρόγραμμα πελάτης να μην χρησιμοποιεί άμεσα την δομή των ενοτήτων στα .h παρά μόνο μέσω των συναρτήσεων που ορίζονται.
Σας δίδεται αρχικό πρόγραμμα που θα επεκτείνετε, καθώς επίσης και πρόγραμμα δοκιμής ουράς σε ξεχωριστό κατάλογο.
Συνιστάται να κρατάτε αντίγραφα των προγραμμάτων σε κάθε στάδιο ανάπτυξης, π.χ. μετά την ολοκλήρωση κάθε ερωτήματος, ώστε να έχετε μια προηγούμενη έκδοση του προγράμματος σας. Ένα αντίγραφο θα σας φανεί πολύ χρήσιμο αν θέλετε να επιστρέψετε σε προηγούμενη έκδοση.

Παραδοτέα
1. Πηγαίος κώδικας (όλα τα .c, .h αρχεία σας). Επίσης το Makefile για gcc-linux ή το project file του DevC++. Σε κάθε αρχείο να αναφέρετε το όνομά σας στο εισαγωγικό σχόλιο.
2. Τεκμηρίωση (μέγιστο 1 σελίδα) Σύντομο κείμενο (pdf) με την εξής δομή
   - Τα στοιχεία σας: (Όνομα-Επώνυμο-ΑΜ)
   - Λειτουργικότητα: Να περιγράψετε τι κάνει το πρόγραμμά σας (μπορεί να κάνει περισσότερα ή και λιγότερα από τα ζητούμενα της άσκησης). Ειδικά αν έχετε απαντήσει το ερώτημα 7, να αναφέρετε την πολιτική που σχεδιάσατε.
   - Οδηγίες Χρήσης του προγράμματος σας: π.χ. Διάταξη δεδομένων εισόδου.
   - **Περιβάλλον Υλοποίησης** και Δοκιμών: πχ. Αναπτύχθηκε σε Dev C++ σε περιβάλλον Windows XP, δοκιμάστηκε επίσης σε gcc σε Linux

Οδηγίες Παράδοσης
Το αρχείο τεκμηρίωσης μαζί με τα αρχεία του προγράμματος να τα βάλετε σε έναν φάκελο, τον οποίο θα συμπιέσετε (zip, rar) και θα ανεβάσετε στο eclass. Προσοχή ανεβάστε το στην κατάλληλη κατηγορία υλοποίησης (Dev-C++ ή gcc).

Τρόπος Αξιολόγησης
Οι ασκήσεις είναι **ατομικές** και θα ελεγχθούν για ομοιότητες χρησιμοποιώντας ειδικό σύστημα εντοπισμού ομοιοτήτων/αντιγραφών. Σε περίπτωση μεγάλης «ομοιότητας» όλες οι «παρόμοιες» ασκήσεις θα μηδενιστούν.

Θα αξιολογηθούν η λειτουργικότητα, η δομή και η τεκμηρίωση του προγράμματος. Αναλυτικά:

Λειτουργικότητα (70/100)

1.	Υλοποίηση ΑΤΔ Ουρά με μετρητή	(05)
2.	Πράξη SizeOuras και αλλαγή main	(05)
3.	Επέκταση τύπου στοιχείου Pelati και τυχαίου χρόνου εξυπηρέτησης	(15)
4.	Επέκταση προσομοίωσης με εξάντληση ουράς	(10)
5.	Τύπος στοιχείου Ταμίας και αλλαγές στο main	(25)
6.	Πολλοί ταμίες και νέες εκτυπώσεις	(10)
7.	Πολιτική Διαχείρισης Πελατών-Ταμιών	(10) Bonus

Δομή (25/100)

Οργάνωση σε Ενότητες (.h, .c) και πρόγραμμα πελάτη	(10)
Απόκρυψη Υλοποίησης, σωστή χρήση στο πρόγραμμα-πελάτης	(10)
Δομημένο Πρόγραμμα-πελάτη (μορφοποίηση, σχόλια, κλπ)	(05)

Τεκμηρίωση και Παρουσίαση (5/100)

ΠΡΟΣΟΧΗ:
Για να αξιολογηθεί το πρόγραμμα σας (έστω για την δομή του) πρέπει τουλάχιστον να μεταγλωττίζεται. Αν δεν μεταγλωττίζεται δεν παίρνει βαθμό. Πριν παραδώσετε το πρόγραμμά σας δοκιμάστε το μια τελευταία φορά και βεβαιωθείτε ότι παραδίδετε τα σωστά αρχεία.