Λειτουργικά Συστήματα

Τμήμα Πληροφορικής
Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

2η Εργασία
Χειμερινό Εξάμηνο 2022

Στόχος της εργασίας είναι η εξοικείωσή σας με τους αλγόριθμους δρομολόγησης διεργασιών.

Για το σκοπό αυτό, σας ζητείται να αναπτύξετε κώδικα σε γλώσσα *C*, ο οποίος θα υλοποιεί τους αλγόριθμους δρομολόγησης FCFS, SJF (χωρίς προεκχώρηση), SRTF (με προεκχώρηση) και Round Robin (για οποιαδήποτε τιμή κβάντου).

Στον κατάλογο *code* που συνοδεύει την εκφώνηση, θα βρείτε τα αρχεία γλώσσας προγραμματισμού C στα οποία θα πρέπει να βασίζετε τα παραδοτέα σας. Τα αρχεία αυτά είναι τα:

- **fcfs.c**: Αρχείο όπου θα πρέπει να υλοποιηθεί ο αλγόριθμος FCFS
- **sjf.c**: Αρχείο όπου θα πρέπει να υλοποιηθεί ο αλγόριθμος SJF
- **srtf.c**: Αρχείο όπου θα πρέπει να υλοποιηθεί ο αλγόριθμος SRTF
- **rr.c**: Αρχείο όπου θα πρέπει να υλοποιηθεί ο αλγόριθμος Round Robin
- **Makefile**: Αρχείο make για το compile του κώδικα και τη διενέργεια ελέγχων.

Σε κάθε ένα από τα αρχεία κώδικα C υλοποιείται ένα struct, στο οποίο αποθηκεύονται πληροφορίες για κάθε διεργασία. Μπορείτε να προσθέσετε επιπλέον ιδιότητες στο struct αυτό, αλλά δεν μπορείτε να αφαιρέσετε τις ιδιότητες που υπάρχουν ήδη και αφορούν στο ID της διεργασίας, τον χρόνο άφιξης (arrival) και το χρόνο CPU καταιγισμού (bust time).

Στη συνέχεια, στην αρχή της συνάρτησης `main` υπάρχει υλοποιημένο ένα κομμάτι κώδικα, το οποίο διαβάζει κάποια δεδομένα εισόδου από την τυπική είσοδο (stdin) και τα επεξεργάζεται, συμπληρώνοντας ένα πίνακα με στοιχεία τύπου struct process. Το κομμάτι αυτό του κώδικα δεν πρέπει να το αλλάξετε.

Μπορείτε να προσθέσετε επιπλέον συναρτήσεις, μεταβλητές και structs σύμφωνα με τις ανάγκες σας.

Η εργασία προϋποθέτει την εκτέλεση σε περιβάλλον Linux και τη χρήση των εργαλείων ανάπτυξης κώδικα σε γλώσσα C που υπάρχουν διαθέσιμα σε αυτό (gcc compiler και automake).

Το compile του κώδικα για κάθε αλγόριθμο μπορεί να γίνει με τη χρήση του εργαλείου `make`, δίνοντας ως παράμετρο το όνομα του αλγορίθμου (fcfs, sjf, srt, rr), π.χ.:

```
make fcfs
```

ή συνολικά για όλους τους αλγόριθμους, απλά τρέχοντας:

```
make
```

Το αποτέλεσμα είναι ότι θα δημιουργούνται τα αντίστοιχα εκτελέσιμα αρχεία, τα οποία μπορούν να εκτελεστούν δίνοντας ως τυπική είσοδό τους αρχεία όπως το *tests/INPUT01*, πχ:

```
cat tests/INPUT01 | ./fcfs
```

Τα αρχεία εισόδου, θα πρέπει να είναι απλά text files και να ακολουθούν το παρακάτω format:

- Η πρώτη γραμμή περιέχει έναν ακέραιο, ο οποίος αναφέρεται στον αριθμό των διεργασιών που περιγράφονται παρακάτω.
- Η δεύτερη γραμμή περιέχει ένα ακέραιο, ο οποίος αναφέρεται στο κβάντο που ενδεχομένως χρησιμοποιούν οι αλγόριθμοι. Το κβάντο ορίζεται πάντα, για όλους τους αλγόριθμους, αλλά προφανώς δεν θα χρησιμοποιείται από αλγόριθμους στους οποίους δεν παίζει κανένα ρόλο.
- Οι υπόλοιπες γραμμές (όσες και ο ακέραιος στην πρώτη γραμμή του αρχείου), περιέχουν τρεις ακεραίους, οι οποίοι αντιπροσωπεύουν το ID της διεργασίας, το χρόνο άφιξης της και το χρόνο CPU καταιγισμού της.

Μπορείτε να κάνετε τις παρακάτω παραδοχές:

- Τα αρχεία εισόδου θα έχουν πάντα το σωστό format.
- Οι διεργασίες θα δίνονται με τη σειρά αφιξής τους.
- Δεν θα υπάρχουν κενά στη δρομολόγηση των διεργασιών, όπως για παράδειγμα να τελειώνει η εκτέλεση όλων των υπόλοιπων διεργασιών σε κάποιο χρόνο, αλλά να μην έχει αφιχθεί ακόμα κάποια επόμενη διεργασία.

Ο κώδικας που θα συμπληρώσετε στα αρχεία κώδικα C θα πρέπει στο τέλος της εκτέλεσής του να τυπώνει στην τυπική έξοδο, σε κάθε γραμμή, το ID της διεργασίας που εκτελείται σε κάθε χρόνο (ξεκινώντας από τη χρονική στιγμή 0). Για τις διεργασίες που ορίζονται στο αρχείο *tests/INPUT01* η έξοδος που θα πρέπει να δημιουργείται θα είναι αυτή που υπάρχει στα αντίστοιχα αρχεία *tests/fcfs01*, *tests/sjf01*, *tests/srtf01* και *tests/rr01*, π.χ.:

```
$ cat tests/INPUT01 | ./fcfs
1
1
1
2
2
3
```

Μπορείτε να ελέγξετε αν η έξοδος που δημιουργείται από τα εκτελέσιμα αρχεία σας αντιστοιχεί σε αυτά, τρέχοντας για κάθε αλγόριθμο μια εντολή όπως η

```
make test-fcfs
```

ή για όλους τους αλγόριθμους συνολικά:

```
make test
```

Το ελάχιστο απαιτούμενο για την εργασία, είναι για το ένα test case που δίνεται (*tests/INPUT01*) τρέχοντας την εντολή `make test`, **να μη τυπώνεται τίποτα στο τερματικό** (άρα να υπάρχει αντιστοίχιση της εξόδου των προγραμμάτων με αυτή που βρίσκεται στα αρχεία *tests/fcfs01* κλπ.

Σε περίπτωση που υπάρχουν σφάλματα και τα προγράμματα δεν δημιουργούν την έξοδο που πρέπει, θα υπάρχουν μηνύματα όπως στο παρακάτω παράδειγμα:

```
$ make
$ make test
Files tests/fcfs01 and /dev/fd/63 differ
Files tests/sjf01 and /dev/fd/63 differ
Files tests/srtf01 and /dev/fd/63 differ
Files tests/rr01 and /dev/fd/63 differ
```

Μπορείτε να προσθέσετε νέα test cases δημιουργώντας αρχεία με ονόματα *INPUTXX*, *fcfsXX*, *sjfXX*, *srtfXX* και *rrXX* στο φάκελο *tests*, όπου XX κάποιος ακέραιος με δύο ψηφία (π.χ. INPUT02 κλπ), ώστε να ελέγξετε καλύτερα τη σωστή υλοποίηση των αλγορίθμων.

Δεν θα πρέπει να συνοδεύετε τον κώδικά σας από κάποιο εξωτερικό κείμενο, ο σχολιασμός θα πρέπει να γίνει αποκλειστικά και μόνο πάνω στον κώδικα.

Για την υποβολή της εργασίας σας, θα πρέπει να εκτελέσετε την εντολή:

```
make tar
```

η οποία θα δημιουργήσει ένα αρχείο *scheduling.tar.gz*, το οποίο θα υποβάλλετε μέσω της πλατφόρμας e-learning.

Υπάρχει επιπρόσθετα ο κανόνας `clean`, μέσω του οποίου διαγράφονται τα εκτελέσιμα αρχεία που έχουν δημιουργηθεί καθώς και το αρχείο scheduling.tar.gz, αν υπάρχει:

```
make clean
```

Το αρχείο *Makefile* δεν επιτρέπεται να το αλλάξετε με κανένα τρόπο.

Η εργασία είναι **ατομική**. Απαγορεύεται η οποιαδήποτε μορφή αντιγραφής.

Αν έχετε απορίες σχετικά με την εργασία, μπορείτε να τις υποβάλετε μέσω της πλατφόρμας elearning στο αντίστοιχο forum συζητήσεων.