

2024-2025

# Λειτουργικά Συστήματα 2024 – 2025

Απαντήσεις στην 1η  
Εργαστηριακή Άσκηση  
(Project)

Η ομάδα αποτελείται από τους εξής φοιτητές:

*Απόστολος Ζεκυριάς*

*Σπυρίδων Μανταδάκης*

*Παναγιώτης Παπανικολάου*

*Αλέξανδρος Γεώργιος Χαλαμπάκης*

## Αναλυτικότερες Πληροφορίες:

Απόστολος  
Ζεκυριάς  
1100554

[up1100554@ac.upatras.gr](mailto:up1100554@ac.upatras.gr)

Φοιτητής 3ου έτους

Σπυρίδων  
Μανταδάκης  
1100613

[up1100613@ac.upatras.gr](mailto:up1100613@ac.upatras.gr)

Φοιτητής 3ου έτους

Παναγιώτης  
Παπανικολάου  
1104804

[up1104804@ac.upatras.gr](mailto:up1104804@ac.upatras.gr)

Φοιτητής 3ου έτους

Αλέξανδρος  
Γεώργιος  
Χαλαμπάκης  
1100754

[up1100754@ac.upatras.gr](mailto:up1100754@ac.upatras.gr)

Φοιτητής 3ου έτους

## Γενικές Πληροφορίες

**Σχόλια:** Κατά την ανάπτυξη των προγραμμάτων μας, όλα τα ερωτήματα υλοποιήθηκαν επιτυχώς και οι κώδικες λειτούργησαν ορθά. Κατά τη διαδικασία, δεν προέκυψαν σοβαρά προβλήματα. Ωστόσο, αντιμετωπίσαμε δυσκολίες στο Ερώτημα 1, καθώς χρειάστηκε να τροποποιήσουμε τον κώδικα που είχαμε δημιουργήσει αφού βασιζόταν στη χρήση μενού. Επιπλέον, στο Ερώτημα 3, και συγκεκριμένα στον διαμοιρασμό της ελεύθερης μνήμης σε blocks, η υλοποίηση της συγκεκριμένης λειτουργικότητας μας απασχόλησε για μεγάλο χρονικό διάστημα, λόγω της χρήσης λανθασμένης προσέγγισης κατά την εκχώρηση των δεδομένων.

Στις επόμενες σελίδες παρουσιάζονται οι απαντήσεις της ομάδας μας στο 1<sup>ο</sup> Project του μαθήματος.

## Περιεχόμενα

1. Shell Scripting.....	4
<b>Εισαγωγή δεδομένων στην εφαρμογή.....</b>	<b>4</b>
Screenshot Εκτέλεσης .....	4
<b>Προβολή Στοιχείων Επιβαίνοντα.....</b>	<b>5</b>
Screenshot Εκτέλεσης .....	5
<b>Αλλαγή Στοιχείων Επιβαίνοντα.....</b>	<b>5</b>
Screenshot Εκτέλεσης .....	6
<b>Προβολή Αρχείου.....</b>	<b>6</b>
Screenshot Εκτέλεσης .....	7
<b>Δημιουργία αναφορών.....</b>	<b>8</b>
Screenshot Εκτέλεσης .....	8
2. Συγχρονισμός Διεργασιών και Σημαφόροι .....	11
Βήματα Υλοποίησης: .....	11
<b>Προβλήματα κατά την υλοποίηση.....</b>	<b>11</b>
Screenshot Εκτέλεσης .....	12
3. Χρονοπρογραμματισμός Διεργασιών και Διαχείριση Μνήμης .....	13
1. Δομές Δεδομένων .....	13
• Process:.....	13
• MemoryBlock.....	13
2. Αρχικοποίηση .....	13
3. Διαχείριση Μνήμης.....	13
• Κατανομή μνήμης:.....	13
• Αποδέσμευση μνήμης: .....	13
4. Προσομοίωση Round Robin .....	13
5. Εκτύπωση Καταστάσεων .....	14
6. Εκτέλεση Προγράμματος.....	14
ΠΑΡΑΔΕΙΓΜΑΤΑ ΛΕΙΤΟΥΡΓΙΑΣ .....	14
Process 1 (.....	14
Process 2.....	14
Παράδειγμα 2.....	15
Process 1:.....	16
Process 2:.....	16

Process 3:.....	16
Process 4:.....	16
Process 5:.....	16
4. Χρονοπρογραμματισμός Διεργασιών .....	18

# 1. Shell Scripting

## Εισαγωγή δεδομένων στην εφαρμογή

Συμβουλευτήκαμε διάφορες έγκυρες πηγές από το διαδίκτυο προκειμένου να εξοικειωθούμε με τη χρήση εντολών **Linux**, όπως οι **sed**, **awk**, **grep**, **less**, και άλλες. Η εξοικείωση με αυτές τις εντολές κρίθηκε απαραίτητη, καθώς αποτελούν βασικά εργαλεία για την υλοποίηση των ζητούμενων στα πλαίσια του ερωτήματος.

Το πρόγραμμα αναπτύχθηκε ώστε να διαχειρίζεται δεδομένα επιβατών από δύο πηγές:

**Αρχείο:** Αν δοθεί ένα αρχείο **passenger\_data.csv**, τα δεδομένα του εισάγονται στην εφαρμογή.

**Πληκτρολόγιο:** Αν δε δοθεί αρχείο, το πρόγραμμα επιτρέπει την εισαγωγή δεδομένων μέσω του πληκτρολογίου.

Αν υπάρχει το αρχείο που δόθηκε ως είσοδος, αντιγράφεται το περιεχόμενό του στο αρχείο **passengers.csv**, ενώ αν δεν υπάρχει, ζητείται η εισαγωγή δεδομένων από τον χρήστη στη μορφή:

```
[code];[fullname];[age];[country];[status (Passenger/Crew)];[rescued (Yes/No)]
```

Τα δεδομένα που εισάγει ο χρήστης αποθηκεύονται γραμμή-γραμμή στο αρχείο **passengers.csv** και η εισαγωγή δεδομένων από το πληκτρολόγιο ολοκληρώνεται μόλις ο χρήστης πληκτρολογήσει τη λέξη **exit**.

**Έλεγχος ύπαρξης αρχείου:** Χρησιμοποιήθηκε η εντολή **if [ -f "\$input\_file" ]** για να ελέγξουμε αν το αρχείο υπάρχει πριν την αντιγραφή του περιεχομένου.

**Εισαγωγή μέσω πληκτρολογίου:** Αντιμετωπίστηκε η πρόκληση να διακόπτεται η εισαγωγή δεδομένων μέσω της λέξης **exit**. Αυτό υλοποιήθηκε με βρόχο **while** και έλεγχο στη λέξη **exit**.

## Screenshot Εκτέλεσης

```
$ ./processes_ipc.sh
Δώσε το Path του αρχείου: passenger_data.csv
Τα δεδομένα από το αρχείο passenger_data.csv αποθηκεύτηκαν στο passengers.csv.
```

## Προβολή Στοιχείων Επιβαίνοντα

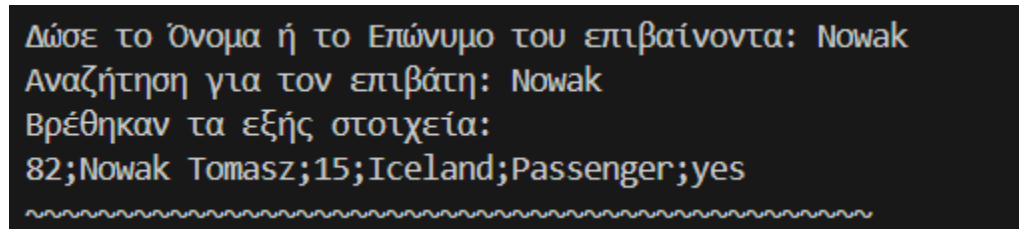
Το πρόγραμμα υλοποιεί τη δυνατότητα αναζήτησης και προβολής των στοιχείων ενός επιβαίνοντα με βάση το όνομα ή το επώνυμό του. Τα δεδομένα λαμβάνονται από το αρχείο `passengers.csv`. Για την υλοποίηση περιλαμβάνει ακολουθούνται τα εξής βήματα:

1. **Ανάγνωση αρχείου:** Το αρχείο `passengers.csv` φορτώνεται στη μνήμη.
2. **Αναζήτηση χρήστη:** Ζητείται από τον χρήστη να εισάγει το όνομα ή το επώνυμο του επιβαίνοντα.
3. **Φιλτράρισμα αποτελεσμάτων:** Ο αλγόριθμος αναζητά γραμμές του αρχείου που περιέχουν το όνομα ή το επώνυμο, ανεξάρτητα από πεζά/κεφαλαία γράμματα.
4. **Προβολή αποτελεσμάτων:** Εμφανίζονται στην οθόνη τα στοιχεία του επιβαίνοντα, αν βρεθεί, διαφορετικά εμφανίζεται μήνυμα ότι δεν εντοπίστηκε.

Για να μην επηρεάζεται η αναζήτηση από τη χρήση πεζών ή κεφαλαίων γραμμάτων, μετατρέπονται όλα τα γράμματα σε πεζά πριν τη σύγκριση.

Για περιπτώσεις όπου υπάρχουν πολλοί επιβάτες με το ίδιο όνομα/επώνυμο το πρόγραμμα εμφανίζει όλες τις σχετικές εγγραφές.

### Screenshot Εκτέλεσης



```
Δώσε το Όνομα ή το Επώνυμο του επιβαίνοντα: Nowak
Αναζήτηση για τον επιβάτη: Nowak
Βρέθηκαν τα εξής στοιχεία:
82;Nowak Tomasz;15;Iceland;Passenger;yes
~~~~~
```

## Αλλαγή Στοιχείων Επιβαίνοντα

Η λειτουργία επιτρέπει τη διόρθωση συγκεκριμένων πεδίων ή ολόκληρης της εγγραφής ενός επιβαίνοντα με βάση τον κωδικό, το όνομα ή το επώνυμό του. Το πρόγραμμα επεξεργάζεται το αρχείο `passengers.csv` και ανανεώνει τα δεδομένα όπως απαιτείται.

Η διαδικασία περιλαμβάνει:

1. **Αναζήτηση εγγραφής:** Το πρόγραμμα αναζητά τον επιβάτη βάσει του κριτηρίου που εισάγει ο χρήστης (κωδικός, όνομα ή επώνυμο).
2. **Επιλογή πεδίου ή ολόκληρης εγγραφής για αλλαγή:** Ο χρήστης καθορίζει μέσω ορίσματος το πεδίο προς αλλαγή ή επιλέγει να αντικαταστήσει ολόκληρη την εγγραφή.

3. **Ενημέρωση δεδομένων:** Η συγκεκριμένη αλλαγή πραγματοποιείται και τα παλιά/νέα δεδομένα προβάλλονται στην οθόνη.
4. **Αποθήκευση στο αρχείο:** Το αρχείο ενημερώνεται με τις αλλαγές

**Διαχείριση Εισόδων:** Έγινε έλεγχος ώστε να ανιχνεύεται η λαθος μορφή των δεδομένων που εισάγονται. Σε τέτοιες περιπτώσεις εμφανίζεται μήνυμα σφάλματος.

**Διατήρηση Μορφής Αρχείου:** Το αρχείο ανανεώνεται μόνο αφού επαληθευθεί ότι η νέα εγγραφή είναι σωστή, αποφεύγοντας αλλαγές στη δομή του.

## Screenshot Εκτέλεσης

```
$ ./processes_ipc.sh Nowak code:999999
Δώσε το Path του αρχείου: passenger_data.csv
Τα δεδομένα από το αρχείο passenger_data.csv αποθηκεύτηκαν στο passengers.csv.
~~~~~
Δώσε το Όνομα ή το Επώνυμο του επιβαίνοντα: Nowak
Αναζήτηση για τον επιβάτη: Nowak
Βρέθηκαν τα εξής στοιχεία:
82;Nowak Tomasz;15;Iceland;Passenger;yes
~~~~~
Δοσμένο όρισμα: Nowak
Αριθμός ορισμάτων: 2
Αναζήτηση για τον επιβάτη: Nowak
Βρέθηκαν τα εξής στοιχεία:
82;Nowak Tomasz;15;Iceland;Passenger;yes
Το πεδίο ενημερώθηκε:
~~~~~
Παλαιά: 82;Nowak Tomasz;15;Iceland;Passenger;yes
Νέα: 999999;Nowak Tomasz;15;Iceland;Passenger;yes
~~~~~
```

## Προβολή Αρχείου

Σε αυτό το μέρος της άσκησης δίνουμε στον χρήστη την ικανότητα προβολής του αρχείου `passengers.csv` και την παρουσίαση όλων των εγγραφών του στην οθόνη. Η διαδικασία προβλέπει τη διαχείριση των δεδομένων, ώστε να γεμίζει η οθόνη με συγκεκριμένο αριθμό εγγραφών κάθε φορά. Ο χρήστης μπορεί να συνεχίσει την προβολή πατώντας το πλήκτρο `<space>`.

### Βήματα Υλοποίησης:

1. **Άνοιγμα Αρχείου:** Το αρχείο `passengers.csv` διαβάζεται γραμμή-γραμμή.

2. **Εμφάνιση Εγγραφών:** Οι εγγραφές εμφανίζονται στην οθόνη, μέχρι να γεμίσει η διαθέσιμη χωρητικότητα.
3. **Εισαγωγή Από Χρήστη:** Το πρόγραμμα αναμένει από τον χρήστη να πατήσει `<space>` για να συνεχίσει ή οποιοδήποτε άλλο πλήκτρο για να διακόψει την προβολή.
4. **Ολοκλήρωση Προβολής:** Όταν όλες οι εγγραφές έχουν εμφανιστεί, το πρόγραμμα ενημερώνει τον χρήστη ότι η προβολή ολοκληρώθηκε.

**Έλεγχος Εισαγωγής Χρήστη:** Ενσωματώθηκε κατάλληλος έλεγχος για την ανίχνευση του πλήκτρου `<space>` και τη διαχείριση άλλων εισαγωγών.

**Διαχείριση Μεγάλων Αρχείων:** Ορίστηκε κατάλληλο μέγεθος σελίδας (π.χ., 10 γραμμές) για να εξασφαλιστεί η ομαλή εμφάνιση των δεδομένων.

## Screenshot Εκτέλεσης

Εμφάνιση περιεχομένου του αρχείου; NAI=1|OXI=0

```
code;fullname;age;country;status;rescued
1;Nunez Jorge;20;Russia;Passenger;Yes
2;Hartmann Wolfgang;21;Germany;Passenger;Yes
3;Haarhoff Lily Tembo;22;United Kingdom;Passenger;yes
4;Zhang Wei;23;France;Passenger;yes
5;Zhang Yang;24;Italy;Passenger;yes
6;Nguyen Cam;25;Spain;Passenger;yes
7;Znaimer Moses;26;Poland;Passenger;yes
8;Phan Don;27;Ukraine;Passenger;no
9;Takahashi Koji;28;Romania;Passenger;yes
10;Chen Ben;29;Netherlands;Passenger;yes
11;Ngoche Alex Obanda;30;Belgium;Crew;no
12;Kobayashi Ken;31;Czech Republic (Czechia);Passenger;Yes
13;Ben Dhifallah Karim;32;Sweden;Passenger;Yes
14;Santos Carlos;33;Portugal;Passenger;yes
15;Korner Karl;34;Greece;Passenger;yes
16;Fayed Paul;35;Hungary;Passenger;yes
17;Charoenpura Somchai;36;Austria;Passenger;yes
18;Abe Kenji;37;Belarus;Passenger;yes
19;Li Lei;38;Switzerland;Passenger;no
20;Khan Babar;39;Bulgaria;Passenger;yes
21;Williams Jack;40;Serbia;Passenger;yes
22;Chen Hsin;41;Denmark;Crew;no
23;Wagner Richard;42;Finland;Passenger;Yes
24;Basov Irina;43;Norway;Passenger;Yes
passengers.csv
```



## Δημιουργία αναφορών

Σε αυτό το σημείο το πρόγραμμα μας πρέπει να μπορεί να δημιουργήσει αναφορές βασισμένες σε ξεχωριστά χαρακτηριστικά. Επομένως αναλύει τα δεδομένα του αρχείου `passengers.csv` και παράγει τέσσερις ξεχωριστές αναφορές. Η επεξεργασία πραγματοποιείται σε επιμέρους βήματα, καθένα από τα οποία εξάγει δεδομένα σε ξεχωριστό αρχείο.

**Οι αναφορές που δημιουργούνται είναι:**

**1. Ηλικιακές Ομάδες (ages.txt):**

- Ταξινόμηση των επιβατών σε ηλικιακές ομάδες: 0-18, 19-35, 36-50, 51+.
- Καταμέτρηση επιβατών ανά ομάδα.

**2. Ποσοστά Διάσωσης (percentages.txt):**

- Υπολογισμός του ποσοστού των διασωθέντων για κάθε ηλικιακή ομάδα.

**3. Μέση Ηλικία (avg.txt):**

- Υπολογισμός της μέσης ηλικίας επιβατών ανά κατηγορία (Passenger ή Crew).

**4. Διασωθέντες (rescued.txt):**

- Δημιουργία νέου αρχείου με δεδομένα μόνο για τους διασωθέντες.

## Screenshot Εκτέλεσης

```
$ ./processes_ipc.sh reports
Δώσε το Path του αρχείου: passenger_data.csv
Τα δεδομένα από το αρχείο passenger_data.csv αποθηκεύτηκαν στο passengers.csv.
~~~~~
Δώσε το Όνομα ή το Επώνυμο του επιβαίνοντα: Nowak
Αναζήτηση για τον επιβάτη: Nowak
Βρέθηκαν τα εξής στοιχεία:
82;Nowak Tomasz;15;Iceland;Passenger;yes
~~~~~
1-18 (1)
19-35 (2)
36-50 (3)
51+ (4)
Δώσε το γκρουπ ηλικιακής ομάδας: 1
Εμφάνιση περιεχομένου του αρχείου; NAI=1|OXI=0 0
```

```

ages.txt
96 842;Smith Michael;15;Slovakia;Passenger;yes
97 843;Silva Rodrigo;16;Ireland;Passenger;yes
98 844;Zhang Yang;17;Croatia;Passenger;no
99 845;Kerndlova Jan;18;Bosnia and Herzegovina;Passenger;yes
100 913;Bach Paul;10;Netherlands;Crew;no
101 914;Anwar Muhammad;11;Belgium;Passenger;Yes
102 915;Angello Dimitrios;12;Czech Republic (Czechia);Passenger;Yes
103 916;Liu Jun;13;Sweden;Passenger;yes
104 917;Hoffmann Walter;14;Portugal;Passenger;yes
105 918;Tavares Artur;15;Greece;Passenger;yes
106 919;Antonio Armando;16;Hungary;Passenger;yes
107 920;Cardinale Ammar;17;Austria;Passenger;yes
108 921;Schmidt Wolfgang;18;Belarus;Passenger;no
109 989;Kaczmarek Kazimierz;10;San Marino;Passenger;yes
110 990;Bernal Pablo;11;Russia;Crew;yes
111 991;Ha Cesar;12;Germany;Passenger;Yes
112 992;Carter Robert;13;United Kingdom;Passenger;Yes
113 993;Perera Nihal;14;France;Passenger;yes
114 994;Davis Mike;15;Italy;Passenger;yes
115 995;Yang Alice;16;Spain;Passenger;yes
116 996;Parra Pedro;17;Poland;Passenger;yes
117 997;Ho Chau;18;Ukraine;Passenger;yes
118 1065;Negga Aisso;10;Latvia;Passenger;yes
119 1066;Ito Jun;11;North Macedonia;Passenger;yes
120 1067;Zhang Li;12;Estonia;Crew;yes
121 1068;Garcia Manuel;13;Luxembourg;Passenger;Yes
122 1069;He Chen;14;Montenegro;Passenger;Yes
123 1070;Kolessov Marat;15;Malta;Passenger;yes
124 1071;Wei Baoping;16;Iceland;Passenger;yes
125 1072;Choi Bob;17;Andorra;Passenger;yes
126 1073;Lima Ricardo;18;Liechtenstein;Passenger;yes
127 1141;Zhang Zhi;10;Finland;Passenger;no
128 1142;Muller Carl;11;Norway;Passenger;yes
129 1143;Zhang Yang;12;Slovakia;Passenger;yes
130 1144;Fulton James;13;Ireland;Crew;yes
131 1145;Araa Abdul Rahim;14;Croatia;Passenger;Yes
132 1146;Alves Rafael;15;Bosnia and Herzegovina;Passenger;no
133 1147;Cheng Aska;16;Moldova;Passenger;yes
134 1148;Bacs Jozef;17;Lithuania;Passenger;yes
135 1149;Johnson Jack;18;Albania;Passenger;yes
136 1217;Russo Giuseppe;10;Sweden;Passenger;yes
137 1218;Thapa Ruby;11;Portugal;Passenger;no
138 1219;Herzigov Josef;12;Greece;Passenger;yes
139 1220;Ahmed Javed;13;Hungary;Passenger;yes
140 1221;Kamkar Ali;14;Austria;Crew;yes
141 1222;Brown John;15;Belarus;Passenger;Yes
142 1223;Herath Lewis;16;Switzerland;Passenger;no
143 1224;Lopez Enrique;17;Bulgaria;Passenger;yes
144 1225;Chen Peng;18;Serbia;Passenger;yes
145
146

```

```

avg.txt
1 Μέση ηλικία πληρώματος: 13.2857
2 Μέση ηλικία επιβατών: 14.0769
3

```

```
percentages.txt
1 percentage for choice 1: 67%
2
3
```

```
rescued.txt
1 code;fullname;age;country;status;rescued
2 77;Li Lin;10;North Macedonia;Crew;yes
3 80;Kitoko Jean-Pierre;13;Montenegro;Passenger;yes
4 81;Smith Tom;14;Malta;Passenger;yes
5 82;Nowak Tomasz;15;Iceland;Passenger;yes
6 83;Yves Douard;16;Andorra;Passenger;yes
7 84;Allen Richard;17;Liechtenstein;Passenger;yes
8 85;Black George;18;Monaco;Passenger;yes
9 153;Niculi Serghei;10;Norway;Passenger;yes
10 154;Nilsson Johan;11;Slovakia;Crew;yes
11 157;Xu Changpeng;14;Bosnia and Herzegovina;Passenger;yes
12 158;Kobayashi Prince;15;Moldova;Passenger;yes
13 159;Wei Yang;16;Lithuania;Passenger;yes
14 160;Ayar Mamoon Eshaq 0;17;Albania;Passenger;yes
15 161;Yang Yang;18;Slovenia;Passenger;yes
16 229;Reyes Chito;10;Portugal;Passenger;yes
17 230;Wei Ning;11;Greece;Passenger;yes
18 231;Al-Mukhaini Ali;12;Hungary;Crew;yes
19 234;Li Ge;15;Switzerland;Passenger;yes
20 235;Budjana Zaenal;16;Bulgaria;Passenger;yes
21 236;Li Chen;17;Serbia;Passenger;yes
22 237;Zhang Benedict;18;Denmark;Passenger;yes
23 306;Zhang Wei;11;Italy;Passenger;yes
24 307;Wong Mohamed;12;Spain;Passenger;yes
25 311;Li Yang;16;Netherlands;Passenger;yes
26 312;Anwar A.;17;Belgium;Passenger;yes
27 313;Perera M;18;Czech Republic (Czechia);Passenger;yes
28 381;Liu Li;10;Montenegro;Passenger;yes
29 382;Alvarez Alfredo;11;Malta;Passenger;yes
30 383;Koirala Kamal;12;Iceland;Passenger;yes
31 384;Anwar Amrozi Bin;13;Andorra;Passenger;yes
32 388;Dijkstra Albert;17;Russia;Passenger;yes
33 389;Acuadhi Aharon;18;Germany;Passenger;yes
34 457;Alonso Antonio;10;Croatia;Passenger;yes
35 458;Wang Jun;11;Bosnia and Herzegovina;Passenger;yes
36 460;Santos Rafael;13;Lithuania;Passenger;yes
37 461;Butt Imran;14;Albania;Passenger;yes
38 465;Li Zheng;18;Estonia;Passenger;yes
39 533;Smith Peter;10;Austria;Passenger;yes
40 534;Bandara Duleep;11;Belarus;Passenger;yes
41 535;Loan Batong;12;Switzerland;Passenger;yes
42 537;Becker Paul;14;Serbia;Passenger;yes
43 538;Colombo Paolo;15;Denmark;Passenger;yes
44 609;Dia Djibril Diop;10;Poland;Passenger;yes
45 610;Zhao Ding;11;Ukraine;Passenger;yes
46 611;Samir Abo El Seoud;12;Romania;Passenger;yes
47 612;Malik Amer Haider Khan;13;Netherlands;Passenger;yes
48 614;Wang Ning;15;Czech Republic (Czechia);Passenger;yes
49 615;Hu Yang;16;Sweden;Passenger;yes
50 685;Wang Jie;10;Andorra;Passenger;yes
51 686;Malik Javed;11;Liechtenstein;Passenger;yes
52 688;Alvarez Manuel;13;San Marino;Passenger;yes
53 689;Khan Ahsan;14;Russia;Passenger;yes
54 690;Silva Rafael;15;Germany;Passenger;yes
55 691;Nguyen Jacques Van Mau;16;United Kingdom;Passenger;yes
56 692;Niang Andre;17;France;Passenger;yes
57 693;Bianchi Stefano;18;Italy;Crew;yes
58 762;Dang Batong;11;Albania;Passenger;yes
59 764;Liu Jie;13;Latvia;Passenger;yes
60 765;Rodriguez Gabriela;14;North Macedonia;Passenger;yes
61 766;Tekin Mehmet;15;Estonia;Passenger;yes
62 768;Zhang Diana;17;Montenegro;Passenger;yes
63 769;Yang Lin;18;Malta;Passenger;yes
64 840;Zhang Lin;13;Finland;Passenger;yes
65 841;Adhia Edward;14;Norway;Passenger;yes
66 842;Smith Michael;15;Slovakia;Passenger;yes
67 843;Silva Rodrigo;16;Ireland;Passenger;yes
```

## 2. Συγχρονισμός Διεργασιών και Σημαφόροι

Η άσκηση αφορά την προσομοίωση μιας διαδικασίας εγκατάλειψης πλοίου, όπου οι επιβαίνοντες προσπαθούν να επιβιβαστούν σε λέμβους διάσωσης υπό περιορισμούς χωρητικότητας. Η προσέγγιση μας βασίζεται στη χρήση διεργασιών (processes) και σημαφόρων για τον συγχρονισμό των επιβαινόντων και των διαθέσιμων θέσεων.

Κάθε λέμβος έχει περιορισμένο αριθμό θέσεων, ο οποίος ελέγχεται μέσω των σημαφόρων. Οι επιβάτες περιμένουν μέχρι να απελευθερωθεί θέση και στη συνέχεια επιβιβάζονται. Ο συγχρονισμός αυτός εξασφαλίζει ότι δεν θα επιβιβαστούν περισσότεροι επιβάτες από τις διαθέσιμες θέσεις των λέμβων.

### Βήματα Υλοποίησης:

1. Ο κύριος κώδικας δημιουργεί τον σημαφόρο και εκκινεί τις διεργασίες των επιβατών.
2. Κάθε διεργασία ελέγχει τη διαθεσιμότητα θέσης μέσω του σημαφόρου.
3. Οι επιβάτες επιβιβάζονται αν υπάρχει θέση. Αν όχι, περιμένουν.
4. Όταν ένας επιβάτης αποβιβάζεται, απελευθερώνει τη θέση.

Πρέπει να σημειωθεί ότι η εκτέλεση του προγράμματος πραγματοποιείται μέσω του αρχείου [launch.c](#), ενώ τα αρχεία [ipc\\_utils.h](#) και [passenger.c](#) λειτουργούν ως βοηθητικά αρχεία. Ωστόσο, για την ορθή λειτουργία του [launch.c](#), είναι απαραίτητο να έχει προηγηθεί η δημιουργία του εκτελέσιμου αρχείου που αντιστοιχεί στο [passenger.c](#).

### Προβλήματα κατά την υλοποίηση

**1° Πρόβλημα:** Κατά την ανάπτυξη της εφαρμογής αυτής, αντιμετωπίσαμε ένα αρκετά μεγάλο πρόβλημα στη διαχείριση των σημαφόρων στο [Windows API](#). Συγκεκριμένα, διαπιστώσαμε ότι το header [<semaphores.h>](#) δεν είναι διαθέσιμο σε συστήματα [Windows](#), καθώς αποτελεί αποκλειστικό στοιχείο των [Unix](#). Αυτό είχε ως αποτέλεσμα την αδυναμία χρήσης threads ή ακόμη και απλών διεργασιών.

Παράλληλα, η δεύτερη προσέγγιση μας που βασίστηκε στη χρήση του header [<windows.h>](#), όπως προτάθηκε από πολλές διαδικτυακές πηγές, δεν απέδωσε τα επιθυμητά αποτελέσματα. Τελικά, καταλήξαμε στη χρήση του header [<winbase.h>](#), το οποίο μας επέτρεψε να υλοποιήσουμε τις απαραίτητες διεργασίες. Ωστόσο, αυτή η επιλογή καθιστά το πρόγραμμα μη εκτελέσιμο σε [Unix](#) συστήματα, καθώς βασίζεται εξ ολοκλήρου στο [Windows API](#).

Επιπλέον, παρατηρήθηκε ότι ενώ το πρόγραμμα εκτελέστηκε επιτυχώς στους τρεις από τους τέσσερις υπολογιστές της ομάδας, σε έναν από αυτούς παρουσίασε προβλήματα, η αιτία των οποίων παραμένει αδιευκρίνιστη.

**2° Πρόβλημα:** Κατά τη διάρκεια της εκτέλεσης, υπήρξαν περιπτώσεις όπου ένας από τους σημαφόρος δεν μπορούσε να ανοίξει. Για να επιλύσουμε το πρόβλημα αυτό προσθέσαμε μηνύματα σφάλματος (π.χ. `Error: ΑΔΥΝΑΜΙΑ ΔΗΜΙΟΥΡΓΙΑΣ ΣΗΜΑΦΟΡΟΥ. Error code: %lu\n"`) και σιγουρευτήκαμε ότι όλοι οι σημαφόροι κλείνουν σωστά στο τέλος.

## Screenshot Εκτέλεσης

```
Enter number of passengers: 10
Enter number of boats: 2
Enter seats per boat: 2
Passenger: Attempting to board the boat...
Passenger: Successfully boarded!
Passenger: Attempting to board the boat...
Passenger: Successfully boarded!
Passenger: Attempting to board the boat...
Passenger: Attempting to board the boat...
Passenger: Attempting to board the boat...
Passenger: Attempting to board the boat...
Passenger: Attempting to board the boat...
Passenger: Attempting to board the boat...
All passengers are attempting to board. Please wait...
Passenger: Attempting to board the boat...
Passenger: Attempting to board the boat...
Passenger: Disembarking and freeing up a seat.
Passenger: Disembarking and freeing up a seat.
Passenger: Successfully boarded!
Passenger: Successfully boarded!
Passenger: Disembarking and freeing up a seat.
Passenger: Disembarking and freeing up a seat.
Passenger: Successfully boarded!
Passenger: Successfully boarded!
Passenger: Disembarking and freeing up a seat.
Passenger: Disembarking and freeing up a seat.
Passenger: Successfully boarded!
Passenger: Successfully boarded!
Passenger: Disembarking and freeing up a seat.
Passenger: Disembarking and freeing up a seat.
Passenger: Successfully boarded!
Passenger: Successfully boarded!
Passenger: Disembarking and freeing up a seat.
Passenger: Disembarking and freeing up a seat.
Simulation complete. All passengers have been processed.

Process returned 0 (0x0)   execution time : 15.133 s
Press any key to continue.
```

### 3. Χρονοπρογραμματισμός Διεργασιών και Διαχείριση Μνήμης

Το πρόγραμμα που υλοποιήσαμε προσομοιώνει **προγραμματισμό διαδικασιών (process scheduling)** με χρήση της **πολιτικής Round Robin (RR)** και διαχείριση μνήμης με στατική κατανομή. Παρακάτω παρατίθεται μια συνοπτική περιγραφή του:

#### 1. Δομές Δεδομένων

- **Process:** Αναπαριστά μια διεργασία και περιλαμβάνει:
  - ID της διεργασίας (pid), χρόνο άφιξης, διάρκεια, εναπομένουσα διάρκεια, απαιτήσεις μνήμης και αν είναι φορτωμένη στη μνήμη.
  -
- **MemoryBlock:** Αναπαριστά ένα τμήμα μνήμης και περιλαμβάνει:
  - Διεύθυνση εκκίνησης, μέγεθος, αν είναι ελεύθερο και το ID της διεργασίας που το κατέχει.

#### 2. Αρχικοποίηση

- Η μνήμη αρχικοποιείται ως ένα μεγάλο ελεύθερο μπλοκ 512 KB.
- Οι διεργασίες εισάγονται από τον χρήστη, παρέχοντας:
  - Χρόνο άφιξης, συνολική διάρκεια εκτέλεσης και απαιτούμενη μνήμη.

#### 3. Διαχείριση Μνήμης

- **Κατανομή μνήμης:** Ελέγχεται αν υπάρχει ελεύθερο μπλοκ με αρκετή μνήμη. Αν ναι, καταλαμβάνεται, και αν περισσεύει μνήμη, το μπλοκ "σπάει".
- **Αποδέσμευση μνήμης:** Απελευθερώνεται η μνήμη όταν ολοκληρωθεί μια διεργασία.

#### 4. Προσομοίωση Round Robin

- Προσομοιώνει την εκτέλεση των διεργασιών σε κβάντα χρόνου (**time slices**) των 3ms:
  - Αν μια διεργασία δεν έχει φορτωθεί στη μνήμη, επιχειρείται κατανομή μνήμης.
  - Η διεργασία εκτελείται για το διαθέσιμο **time slice** (ή λιγότερο αν τελειώνει νωρίτερα).
  - Μετά από κάθε **time slice**, ενημερώνονται οι χρόνοι και η μνήμη.
- Αν όλες οι διεργασίες ολοκληρωθούν, η προσομοίωση σταματά.

## 5. Εκτύπωση Καταστάσεων

- Κατά την εκτέλεση:
  - Εμφανίζονται μηνύματα όταν μια διεργασία φορτώνεται/περιμένει μνήμη.
  - ΕΜΦΑΝΙΖΟΝΤΑΙ ΤΑ ΤΜΗΜΑΤΑ ΜΝΗΜΗΣ ΜΕ ΤΗΝ ΚΑΤΑΣΤΑΣΗ ΚΑΘΕ ΜΠΛΟΚ (ΕΛΕΥΘΕΡΟ Η κατειλημμένο).

## 6. Εκτέλεση Προγράμματος

- Ο χρήστης εισάγει:
  - Αριθμό διεργασιών (μέγιστο 5).
  - Τα χαρακτηριστικά κάθε διεργασίας.
- Ξεκινά η προσομοίωση με διαχείριση μνήμης και εκτέλεση διεργασιών.

## ΠΑΡΑΔΕΙΓΜΑΤΑ ΛΕΙΤΟΥΡΓΙΑΣ

Παράδειγμα 1.

Εισάγω 2 διεργασίες, η πρώτη με χρόνο άφιξης 0, χρόνο εκτέλεσης 10 και απαιτούμενη μνήμη 10 KB και η δεύτερη με χρόνο άφιξης 2, χρόνο εκτέλεσης 8 και απαιτούμενη μνήμη 10 KB.

**Process 1** (άφιξη 0 ms): Φορτώνεται στη μνήμη, εκτελείται σε 4 time slices (3 + 3 + 3 + 1 ms) και ολοκληρώνεται στα 10 ms.

**Process 2** (άφιξη 2 ms): Φορτώνεται στη μνήμη μόλις η Process 1 ολοκληρωθεί, εκτελείται σε 3 time slices (3 + 3 + 2 ms) και ολοκληρώνεται στα 15 ms.

Η μνήμη αποδεσμεύεται πλήρως μετά την ολοκλήρωση και των δύο διεργασιών.



```

Enter the number of processes (max 5): 2

Enter arrival time, total duration, and memory needed (in KB) for Process 1: 0 10 10

Enter arrival time, total duration, and memory needed (in KB) for Process 2: 2 8 10
Time 0: Process 1 loaded into memory.

Memory Status:
Block 0: Start=0, Size=10, Free=No, PID=1
Block 1: Start=10, Size=502, Free=Yes, PID=-1
~~~~~

Time 0: Process 1 is running for 3 msec.
Time 3: Process 2 loaded into memory.

Memory Status:
Block 0: Start=0, Size=10, Free=No, PID=1
Block 1: Start=10, Size=10, Free=No, PID=2
Block 2: Start=20, Size=492, Free=Yes, PID=-1
~~~~~

Time 3: Process 2 is running for 3 msec.
Time 6: Process 1 is running for 3 msec.
Time 9: Process 2 is running for 3 msec.
Time 12: Process 1 is running for 3 msec.
Time 15: Process 2 is running for 2 msec.
Time 17: Process 2 completed.

Memory Status:
Block 0: Start=0, Size=10, Free=No, PID=1
Block 1: Start=10, Size=10, Free=Yes, PID=-1
Block 2: Start=20, Size=492, Free=Yes, PID=-1
~~~~~

Time 17: Process 1 is running for 1 msec.
Time 18: Process 1 completed.

Memory Status:
Block 0: Start=0, Size=10, Free=Yes, PID=-1
Block 1: Start=10, Size=10, Free=Yes, PID=-1
Block 2: Start=20, Size=492, Free=Yes, PID=-1
~~~~~

```

## Παράδειγμα 2.

Εισάγω 5 διεργασίες,

1. με χρόνο άφιξης 0, χρόνο εκτέλεσης 10 και απαιτούμενη μνήμη 10 KB.
2. με χρόνο άφιξης 2, χρόνο εκτέλεσης 8 και απαιτούμενη μνήμη 10 KB.
3. με χρόνο άφιξης 4, χρόνο εκτέλεσης 5 και απαιτούμενη μνήμη 10 KB.
4. με χρόνο άφιξης 6, χρόνο εκτέλεσης 15 και απαιτούμενη μνήμη 10 KB.
5. με χρόνο άφιξης 8, χρόνο εκτέλεσης 2 και απαιτούμενη μνήμη 10 KB.



Process 1: Ξεκινά από 0 ms, ολοκληρώνεται στα **21 ms**.

Process 2: Ξεκινά από 2 ms, ολοκληρώνεται στα **23 ms**.

Process 3: Ξεκινά από 4 ms, ολοκληρώνεται στα **16 ms**.

Process 4: Ξεκινά από 6 ms, ολοκληρώνεται στα **30 ms**.

Process 5: Ξεκινά από 8 ms, ολοκληρώνεται στα **10 ms**.

Η εκτέλεση ολοκληρώνεται στα **30 ms**, με όλες τις διεργασίες και τη μνήμη αποδεδουλευμένες.

```
Enter the number of processes (max 5): 5
Enter arrival time, total duration, and memory needed (in KB) for Process 1: 0 10 10
Enter arrival time, total duration, and memory needed (in KB) for Process 2: 2 8 10
Enter arrival time, total duration, and memory needed (in KB) for Process 3: 4 5 10
Enter arrival time, total duration, and memory needed (in KB) for Process 4: 6 15 10
Enter arrival time, total duration, and memory needed (in KB) for Process 5: 8 2 10
Time 0: Process 1 loaded into memory.

Memory Status:
Block 0: Start=0, Size=10, Free=No, PID=1
Block 1: Start=10, Size=502, Free=Yes, PID=-1
~~~~~

Time 0: Process 1 is running for 3 msec.
Time 3: Process 2 loaded into memory.

Memory Status:
Block 0: Start=0, Size=10, Free=No, PID=1
Block 1: Start=10, Size=10, Free=No, PID=2
Block 2: Start=20, Size=492, Free=Yes, PID=-1
~~~~~

Time 3: Process 2 is running for 3 msec.
Time 6: Process 3 loaded into memory.

Memory Status:
Block 0: Start=0, Size=10, Free=No, PID=1
Block 1: Start=10, Size=10, Free=No, PID=2
Block 2: Start=20, Size=10, Free=No, PID=3
Block 3: Start=30, Size=482, Free=Yes, PID=-1
~~~~~

Time 6: Process 3 is running for 3 msec.
Time 9: Process 4 loaded into memory.

Memory Status:
Block 0: Start=0, Size=10, Free=No, PID=1
Block 1: Start=10, Size=10, Free=No, PID=2
Block 2: Start=20, Size=10, Free=No, PID=3
Block 3: Start=30, Size=10, Free=No, PID=4
Block 4: Start=40, Size=472, Free=Yes, PID=-1
~~~~~

Time 9: Process 4 is running for 3 msec.
Time 12: Process 5 loaded into memory.
```

```
Memory Status:
Block 0: Start=0, Size=10, Free=No, PID=1
Block 1: Start=10, Size=10, Free=No, PID=2
Block 2: Start=20, Size=10, Free=No, PID=3
Block 3: Start=30, Size=10, Free=No, PID=4
Block 4: Start=40, Size=10, Free=No, PID=5
Block 5: Start=50, Size=462, Free=Yes, PID=-1
~~~~~
```

```
Time 12: Process 5 is running for 2 msec.
Time 14: Process 5 completed.
```

```
Memory Status:
Block 0: Start=0, Size=10, Free=No, PID=1
Block 1: Start=10, Size=10, Free=No, PID=2
Block 2: Start=20, Size=10, Free=No, PID=3
Block 3: Start=30, Size=10, Free=No, PID=4
Block 4: Start=40, Size=10, Free=Yes, PID=-1
Block 5: Start=50, Size=462, Free=Yes, PID=-1
~~~~~
```

```
Time 14: Process 1 is running for 3 msec.
Time 17: Process 2 is running for 3 msec.
Time 20: Process 3 is running for 2 msec.
Time 22: Process 3 completed.
```

```
Memory Status:
Block 0: Start=0, Size=10, Free=No, PID=1
Block 1: Start=10, Size=10, Free=No, PID=2
Block 2: Start=20, Size=10, Free=Yes, PID=-1
Block 3: Start=30, Size=10, Free=No, PID=4
Block 4: Start=40, Size=10, Free=Yes, PID=-1
Block 5: Start=50, Size=462, Free=Yes, PID=-1
~~~~~
```

```
Time 22: Process 4 is running for 3 msec.
Time 25: Process 1 is running for 3 msec.
Time 28: Process 2 is running for 2 msec.
Time 30: Process 2 completed.
```

```
Memory Status:
Block 0: Start=0, Size=10, Free=No, PID=1
Block 1: Start=10, Size=10, Free=Yes, PID=-1
Block 2: Start=20, Size=10, Free=Yes, PID=-1
Block 3: Start=30, Size=10, Free=No, PID=4
Block 4: Start=40, Size=10, Free=Yes, PID=-1
Block 5: Start=50, Size=462, Free=Yes, PID=-1
~~~~~
```

```
Time 30: Process 4 is running for 3 msec.
```

```

Time 33: Process 1 is running for 1 msec.
Time 34: Process 1 completed.

Memory Status:
Block 0: Start=0, Size=10, Free=Yes, PID=-1
Block 1: Start=10, Size=10, Free=Yes, PID=-1
Block 2: Start=20, Size=10, Free=Yes, PID=-1
Block 3: Start=30, Size=10, Free=No, PID=4
Block 4: Start=40, Size=10, Free=Yes, PID=-1
Block 5: Start=50, Size=462, Free=Yes, PID=-1
~~~~~

Time 34: Process 4 is running for 3 msec.
Time 37: Process 4 is running for 3 msec.
Time 40: Process 4 completed.

Memory Status:
Block 0: Start=0, Size=10, Free=Yes, PID=-1
Block 1: Start=10, Size=10, Free=Yes, PID=-1
Block 2: Start=20, Size=10, Free=Yes, PID=-1
Block 3: Start=30, Size=10, Free=Yes, PID=-1
Block 4: Start=40, Size=10, Free=Yes, PID=-1
Block 5: Start=50, Size=462, Free=Yes, PID=-1
~~~~~

```

## 4. Χρονοπρογραμματισμός Διεργασιών

ΔΙΕΡΓΑΣΙΑ	ΧΡΟΝΟΣ ΑΦΙΞΗΣ	ΔΙΑΡΚΕΙΑ ΕΚΤΕΛΕΣΗΣ	PID
A	0	6	3
B	2	4	1
Γ	3	1	2
Δ	4	3	5
E	5	4	4
Z	6	7	1

Παρακάτω φαίνονται τα διαγράμματα Gant καθε αλγόριθμου χρονοδρομολόγησης, μαζί με τους πίνακες υπολογισμού των μέσων τιμών των χρόνων.

α)

FCFS:

A	A	A	A	A	A	B	B	B	B	Γ	Δ	Δ	Δ	E	E	E	E	Z	Z	Z	Z	Z	Z	Z
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

ΔΙΕΡΓΑΣΙΑ	ΧΡΟΝΟΣ ΑΝΑΜΟΝΗΣ	ΧΡΟΝΟΣ ΑΠΟΚΡΙΣΗΣ	ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ	ΘΕΜΑΤΙΚΕΣ ΑΛΛΑΓΕΣ
A	0	0	6	1
B	4	4	8	1
Γ	7	7	8	1
Δ	7	7	10	1
E	9	9	13	1
Z	12	12	19	1
ΜΕΣΟΣ ΟΡΟΣ	39/6=6.5	39/6=6.5	64/6=10.67	1

SJF:

A: 6		B: 4 A: 4	Γ: 1 B: 4 A: 3	Δ: 3 Γ: 1 B: 4 A: 2 A: 1	E: 4 Δ: 3 Γ: 1 B: 4 A: 1	Z:7 E:4 Δ: 3 Γ:1 B: 4	Z:7 E:4 Δ: 3 B: 4		Z:7 E:4 B: 4															
A	A	A	A	A	A	Γ	Δ	Δ	Δ	B	B	B	B	E	E	E	E	Z	Z	Z	Z	Z	Z	Z

ΔΙΕΡΓΑΣΙΑ	ΧΡΟΝΟΣ ΑΝΑΜΟΝΗΣ	ΧΡΟΝΟΣ ΑΠΟΚΡΙΣΗΣ	ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ	ΘΕΜΑΤΙΚΕΣ ΑΛΛΑΓΕΣ
A	0	0	6	1
B	10	10	12	1
Γ	3	3	4	1
Δ	3	3	6	1

E	9	9	13	1
Z	12	12	19	1
ΜΕΣΟΣ ΟΡΟΣ	37/6=6.17	37/6=6.17	60/6=10	1

SRTF:

A: 6		B: 4 A: 4	Γ: 1 B: 4 A: 3	Δ: 3 B: 4 A: 3	E: 4 Δ: 3 B: 4 A: 2	Z:7 E:4 Δ: 3 B: 4 A: 1	Z:7 E:4 Δ: 3 B: 4		Z:7 E:4 B: 4					Z:7 E:4				Z: 7							
A	A	A	Γ	A	A	A	Δ	Δ	Δ	B	B	B	B	E	E	E	E	Z	Z	Z	Z	Z	Z	Z	Z

ΔΙΕΡΓΑΣΙΑ	ΧΡΟΝΟΣ ΑΝΑΜΟΝΗΣ	ΧΡΟΝΟΣ ΑΠΟΚΡΙΣΗΣ	ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ	ΘΕΜΑΤΙΚΕΣ ΑΛΛΑΓΕΣ
A	1	0	7	2
B	8	8	12	1
Γ	0	0	1	1
Δ	3	3	6	1
E	9	9	13	1
Z	12	12	19	1
ΜΕΣΟΣ ΟΡΟΣ	33/6=5.5	32/6= 5.34	58/6=9.67	7/6=1.17

RR:

A: 6		A: 4 B: 4	Γ: 1 A: 4 B: 3	B: 2 Δ: 3 Γ: 1 A: 4	E: 4 B: 2 Δ: 3 Γ: 1 A: 3	A: 2 Z:7 E:4 B: 2 Δ: 3 Γ:1	A: 2 Z:7 E:4 B: 2 Δ: 3		Δ: 1 A: 2 Z:7 E:4 B: 2		Δ: 1 A: 2 Z:7 E:4		E:2 Δ: 1 A: 2 Z:7		Z:5 E:2 Δ: 1 A: 2		Z:5 E:2 Δ: 1	Z:5 E:2								
A	A	B	B	A	A	Γ	Δ	Δ	B	B	E	E	Z	Z	A	A	Δ	E	E	Z	Z	Z	Z	Z	Z	

ΔΙΕΡΓΑΣΙΑ	ΧΡΟΝΟΣ ΑΝΑΜΟΝΗΣ	ΧΡΟΝΟΣ ΑΠΟΚΡΙΣΗΣ	ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ	ΘΕΜΑΤΙΚΕΣ ΑΛΛΑΓΕΣ
A	11	0	17	3
B	5	0	9	2
Γ	3	3	4	1
Δ	11	3	14	2
E	11	6	15	2
Z	12	7	19	2
ΜΕΣΟΣ ΟΡΟΣ	53/6=8.84	19/6=3.17	78/6=13	12/6=2

Y)

Σύντομος ψευδοκώδικας για την λειτουργία του αλγόριθμου LRTFP:

Όσο υπάρχουν διεργασίες προς εκτέλεση:

Ενημέρωσε τη λίστα των έτοιμων διεργασιών (Ready Queue).

Αν υπάρχει τρέχουσα διεργασία:

Ενημέρωσε τον χρόνο που απομένει (Remaining Time) της τρέχουσας διεργασίας.

Αν η Remaining Time της τρέχουσας διεργασίας γίνει 0:

Ολοκλήρωσε την τρέχουσα διεργασία και αφαίρεσέ την από τη Ready Queue.

Επέλεξε τη διεργασία με τον μεγαλύτερο Remaining Time από τη Ready Queue:

Αν υπάρχει ισοδυναμία, επέλεξε τη διεργασία με το μικρότερο PID.

Αν η νέα διεργασία δεν είναι η ίδια με την τρέχουσα:

Κάνε προεκχώρηση και ξεκίνησε την εκτέλεση της νέας διεργασίας.

Προχώρησε την εκτέλεση κατά 1 μονάδα χρόνου.

LRTFP:

A: 6		B: 4 A: 4	Γ: 1 B: 3 A: 4	Δ:3 Γ:1 B:3 A:3	E:4 Δ:3 Γ:1 B:2 A:3	Z: 7 E: 3 Δ: 3 Γ: 1 B: 2 A: 3				Z: 3 E: 3 Δ: 3 Γ:1 B: 2 A: 3	Z: 2 E: 3 Δ: 3 Γ:1 B: 2 A: 3	Z: 2 E: 3 Δ: 3 Γ:1 B: 2 A: 2	Z: 2 E: 2 Δ: 3 Γ:1 B: 2 A: 2	Z: 2 E: 2 Δ: 2 Γ:1 B: 1 A: 2	Z: 1 E: 2 Δ: 2 Γ:1 B: 1 A: 2	Z: 1 E: 2 Δ: 2 Γ:1 B: 1 A: 1	Z: 1 E: 1 Δ: 2 Γ:1 B: 1 A: 1	Z: 1 E: 1 Δ: 1 Γ:1 B: 1 A: 1	Z: 1 E: 1 Δ: 1 Γ:1 B: 1 A: 1	E: 1 Δ: 1 Γ:1 A: 1	E: 1 Δ: 1 A: 1	E: 1 Δ: 1	Δ: 1	
A	A	B	A	B	E	Z	Z	Z	Z	Z	A	E	Δ	B	Z	A	E	Δ	B	Z	Γ	A	E	Δ

ΔΙΕΡΓΑΣΙΑ	ΧΡΟΝΟΣ ΑΝΑΜΟΝΗΣ	ΧΡΟΝΟΣ ΑΠΟΚΡΙΣΗΣ	ΧΡΟΝΟΣ ΟΛΟΚΛΗΡΩΣΗΣ	ΘΕΜΑΤΙΚΕΣ ΑΛΛΑΓΕΣ
A	17	0	23	5
B	14	0	18	4
Γ	18	18	19	1
Δ	18	9	21	3
E	14	0	19	4
Z	8	0	15	3
ΜΕΣΟΣ ΟΡΟΣ	89/6=14.8	27/6=4.5	115/6=19.1	20/6=3.33