

## 2Η ΕΡΓΑΣΙΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

C++

Ορφέας- Άγγελος Νικολάου

ΑΜ: 2792

E-MAIL: int02792@uoi.gr

Άρτα, 2024/01/21

## Table of contents

JobShopSchedulingProblem.....	3
Οδηγίες μεταγλώττισης:.....	3
Οδηγίες χρήσεως.....	4
Ερώτημα A1.....	4
Ερώτημα A2.....	5
Ερώτημα B.....	6
UML διάγραμμα.....	7
Catch2.....	7

# **JobShopSchedulingProblem**

Ένα ελεύθερο πρόγραμμα το οποίο διαβάζει ένα αρχείο το οποίο περιέχει ένα πρόβλημα διαχείρισης χρόνου με μηχανές και εργασίες.

[GITHUB MIRROR](#)

## **Οδηγίες μεταγλώττισης:**

Προτιμότερο με χρήση σύστημα GNU + Linux ή \*Nix-like λειτουργικά συστήματα.

- Εγκατάσταση g++ και GNUMakefile.
- Εγκατάσταση Catch2 library ώστε να υπάρχει το header <catch2/catch.hpp> αν ο χρήστης επιθυμεί unit tests.

Για να εκτελέσετε το πρόγραμμα, τρέξτε στο main directory (Erg\_2)

```
$ make (με επιλογή για -jX όπου X αριθμός νημάτων).
```

```
$ ./jssp
```

Για να τρέξετε τα Unit Tests: (από το ίδιο directory).

```
$ cd unit_testing && make -jX && ./tests
```

## Οδηγίες χρήσεως

Με το που τρέξετε το πρόγραμμα, θα εμφανιστεί ένα menu επιλογών σχετικά με ποιο αρχείο επιθυμείτε να διαβάσει το πρόγραμμα. Είναι σημαντικό τα αρχεία να είναι στον υποκατάλογο `./data/` με οποιοδήποτε όνομα από τις βασικές επιλογές που προσφέρονται.

```
-----  
Erg_2 $make -j12 && ./jssp  
g++ -Wall -pedantic -Werror -Wshadow -Wstrict-aliasing -Wstrict-overflow -std=c++11 source/* -o jssp  
Choose a file to use as a job stop scheduling problem  
All of them are inside the data/ folder  
Choose one of the following JSS problems to pick from:  
1. la01.txt  
2. la02.txt  
3. la03.txt  
4. la04.txt  
5. la05.txt  
6. mt06.txt  
7. mt10.txt  
8. mt20.txt
```

### Ερώτημα A1

Αφού επιλέξουμε αρχείο, επιλέγουμε μια εργασία για την οποία θέλουμε να δούμε τον επιθυμητό χρόνο εκτέλεσης, τους χρόνους σε κάθε μηχανή και την σειρά εκτέλεσης που απαιτεί.

```
-----  
|EXERCISE A1|  
-----
```

```
Select what job you want to see times and machine order:
```

```
1 2 3 4 5 6 7 8 9 10
```

```
1
```

```
Due time for job 1: 335
```

```
Time for machine 1 for this job: 53
```

```
Time for machine 2 for this job: 21
```

```
Time for machine 3 for this job: 34
```

```
Time for machine 4 for this job: 55
```

```
Time for machine 5 for this job: 95
```

```
Execution order: 2 1 5 4 3
```

## Ερώτημα A2

Για το ερώτημα A2 έχουμε 3 επιλογές:

switch (επιλογή){

case 1:

Εμφανίζει ταξινομημένους τους επιθυμητούς χρόνους τις κάθε εργασίας (με προκαθορισμένη παράμετρο 1.3).

break;

case 2:

Εμφανίζει ταξινομημένους τους συνολικούς χρόνους η οποία κάθε εργασία απαιτεί.

break;

case 3:

Εμφανίζει ταξινομημένες τις μηχανές κατά συνολικό χρόνο εργασίας τους.

}

```
Erg_2 : bash
Erg_2 $make -j12 && ./jssp | grep "EXERCISE A2" -A 16
g++ -Wall -pedantic -Werror -Wshadow -Wstrict-aliasing -Wstrict-overflow
-w -std=c++11 source/* -o jssp
1
1
[EXERCISE A2]
-----
Choose sorting order:
1. Sort by job due date.
2. Sort by job total work time.
3. Sort by machine totaling work time
1
Due time for job n.2: 241
Due time for job n.3: 288
Due time for job n.9: 302
Due time for job n.5: 308
Due time for job n.8: 319
Due time for job n.1: 335
Due time for job n.6: 429
Due time for job n.4: 460
Due time for job n.10: 481
Due time for job n.7: 536
-----
Erg_2 $

Erg_2 : bash
Erg_2 $./jssp | grep "EXERCISE A2" -A 16
1
1
[EXERCISE A2]
-----
Choose sorting order:
1. Sort by job due date.
2. Sort by job total work time.
3. Sort by machine totaling work time
2
Total work time for job n.2: 186
Total work time for job n.3: 222
Total work time for job n.9: 233
Total work time for job n.5: 237
Total work time for job n.8: 246
Total work time for job n.1: 258
Total work time for job n.6: 330
Total work time for job n.4: 354
Total work time for job n.10: 370
Total work time for job n.7: 413
-----
Erg_2 $

Erg_2 : bash
Erg_2 $./jssp | grep "EXERCISE A2" -A 11
1
1
[EXERCISE A2]
-----
Choose sorting order:
1. Sort by job due date.
2. Sort by job total work time.
3. Sort by machine totaling work time
3
Total work time for machine n.4: 508
Total work time for machine n.3: 530
Total work time for machine n.2: 536
Total work time for machine n.1: 609
Total work time for machine n.5: 666
-----
Erg_2 $
```

## Ερώτημα B

Εκτυπώνει ένα έγκυρο (όχι) πρόγραμμα εργασιών (δηλαδή αν μια εργασία εκτελείται σε ένα μηχάνημα αυτή δεν μπορεί να εκτελείται σε ένα άλλο).

```
-----
|EXERCISE B|
-----
-----
Each column on the terminal are 10 time units.
Each number is a job, in the case it takes more than 10 units,
it will be followed by a dot. Time will be rounded up,
which will be rounded to the upper limit.
(i.e. a job which takes 31 time units will be displayed as 40).
-----
Machine 1: 2..5.....1.....4.....6.....8.....3..7.....9.....10.....
Machine 2: 1..4.....8....9.....6.....2.....5..10.....3....7.....
Machine 3: 3...7.....8....5.....10....1...2...4.....6....9.....
Machine 4: 6.....9..5...10.....2.....3...4.....7.....1.....8...
Machine 5: 3.....7.....10.....2..1.....9...4.....6.....5....8.....
```

Αυτό σχετικά με τις τελείες αλλάζει μέσα σε ένα for loop (αντί για  $k+=10$  να βάλω κάτι άλλο). Όμως το κάθε μηχάνημα θα έπιανε πάνω από μία γραμμή στο CLI.

Το πρόβλημα με αυτό το ερώτημα είναι δεν κατάφερα να μην έχει επικάλυψη και το παράτησα (7 ώρες προσπαθώντας μόνος μου και με LLMs).

Λογικά κάπως με μία bool `is_taken()` όπου κοιτάει αν μια εργασία είναι μέσα σε ένα μηχάνημα.

Πέρα από αυτό, αρκετές φορές δεν σέβονται την σειρά εκτέλεσης μιας εργασίας, όπως φαίνεται με την εργασία '1' στα μηχανήματα 3 και 4 (κανονικά πρέπει πρώτα στο 4 και μετά στο 3).

## UML διάγραμμα

```
31 -----|-----
30 |                                     jssp                                     |
29 |-----|-----
28 | - job_number : u64                                                         |
27 | - machine_number : u64                                                    |
26 | - execution_time : u64[job_number][machine_number]                      |
25 | - execution_order : u64[job_number][machine_number]                     |
24 | - desired_time : u64[job_number]                                         |
23 |-----|-----
22 | + <<constructor>> jssp(filename : char const* const)                      |
21 | + <<constructor>> jssp(jn : <<reference>>u64 const,                          |
20 |                               mn : <<reference>>u64 const,                    |
19 |                               et : <<reference>>u64[job_number][machine_number] const |
18 |                               eo : <<reference>>u64[job_number][machine_number] const) |
17 | + <<destructor>> ~jssp()                                                    |
16 | + print(void)                                                             : void |
15 | + askhsh1(void)                                                           : void |
14 | + askhsh2(void)                                                           : void |
13 | + askhsh3(void)                                                           : void |
12 | - read_file(filename : char const* const)                               : void |
11 | - allocate(void)                                                         : void |
10 | - calc_desired_time(param = 1.3 : double const) : void                  |
 9 | - print_times(job : u64 const) const                                     : void |
 8 | - print_machine_order(job : u64 const) const                             : void |
 7 | - select(void) const                                                     : u64 |
 6 | - print_sorted_due(void)                                                 : void |
 5 | - print_sorted_total(void)                                               : void |
 4 | - print_sorted_machine_total_work_time(void) : void                     |
 3 |-----|-----
```

Μόνο έφταιξα μια κλάση, άρα μόνο υπάρχει ένα <<κουτί>>. Όπως φαίνεται, ο χρήστης μόνο έχει επαφή με την κλάση με τις συναρτήσεις print(), askhsh1(), askhsh2() και askhsh3().

## Catch2

```
Erg_2 $cd unit_testing/ && make -j12 && ./tests
g++ -Wall -pedantic -Werror -Wshadow -Wstrict-aliasing -Wstrict-overflow -std=c++11 ../source/error_checkers.cpp ./unique_jssp.cpp ../source/helpers.cpp ./unit_tests.cpp -o tests
All tests passed (3 assertions in 1 test case)
unit_testing $
```