

ORSI 1. beadandó feladat - dokumentáció

Kovács Bálint - FANW4Z

November 6, 2016

1 Feladat

A bemenet – `input.txt` – első sorában tartalmazza a szóközzel elválasztott `N` és `M` pozitív egészeket – ahol következő `N` sorában pedig egy-egy neptunkód, játéknév és percben eltöltött idő (egész szám) szóközzel elválasztott hármasát:

```
N M
NEPTUN_1 gameazon perc    - az 1. válasz adatai
NEPTUN_2 gameazon perc    - az 2. válasz adatai
...
...
...
NEPTUN_N gameazon perc    - az N. válasz adatai
```

```
//Egy lehetséges konkrét sor pl:
//BATMAN WoW 1500
```

A főfolyamat olvassa be az adatokat, indítson `M` gyerekfolyamatot, majd minden gyerekfolyamathoz társítson egy-egy játékhoz tartozó adathalmazt. (A játékok száma pontosan `M`, így minden játék adatát párhuzamosan kell kiszámolni.) A gyerekfolyamatok dolga megállapítani, hogy az adott játékkal mennyi időt töltöttek összesen (percben) a válaszadók, valamint az átlagos játékidő kiszámítása az adott játékhoz (egész percre lefelé kerekítve - `floor(..)`). Ezt a két adatot küldje vissza a szülőfolyamatnak.

A főfolyamat ezek után a játékokhoz tartozó, összesen és átlagosan eltöltött időt szóközzel elválasztva a JÁTÉKOK azonosítója ALAPJÁN BETŰRENDBEN írja az `output.txt` kimeneti fájlba.

```
//egy példa sor az output fájlból:
//WoW 6000 573
```

2 Felhasználói dokumentáció

2.1 Környezet

A program több platformon futtatható, nincsen dinamikus függősége. Telepítésre nincs szükség, elegendő a futtatható állományt elhelyezni a számítógépen.

2.2 Használat

A program elindítása egyszerű, mivel nem vár parancssori paramétereket, így parancssoron kívül is lehet futtatni. A fájl mellett kell elhelyezni az `input.txt` fájlt, melyet feldolgoz és az eredményt az `output.txt` nevű fájlba írja, a betűrend alapján. Egy lehetséges bemenetet tartalmaz a mellékelt `input.txt` tesztfájl. Saját bemeneti fájlok esetén fontos, hogy a feladatban megadott szempontok alapján írjuk az adatokat az inputfájlba, mivel a feladat megoldása és a program futása során feltesszük, hogy az adatok a fájlban helyesek, így erre külön ellenőrzés nincsen a kódban.

3 Fejlesztői dokumentáció

3.1 A megoldás módja

A kódot logikailag két részre bonthatjuk, egy fő és több alfolyamatra. A főfolyamatot a `start()` függvény fogja megvalósítani, ez fogja beolvasni az input fájl tartalmát, azt csoportosítja és nemszabályos mátrixba rendezi. Minden alfolyamathoz a mátrix egy sorát társítjuk majd, ezekkel számolnak. Az eredményt a főfolyamat fogja beleírni a kimeneti fájlba.

3.2 Implementáció

Az Erlang nyelvi elemeit kihasználva egy lista (melyben $\{String, \{Int, Int\}\}$ párokat tárolunk az első tag szerint rendezve) fogja a különböző szálakon futó függvények visszatérési értékét tárolni. A szükséges N folyamatot a beépített `spawn()` függvény segítségével fogjuk párhuzamosan indítani a `start_process()` függvényben, majd minden szál az `addAndAvg()` függvényt fogja a mátrix hozzá tartozó sorára végrehajtani. Az imént említett függvény a számtani közép alapján számol átlagot, azaz

$x_{avg} = \frac{1}{n} \sum_{i=1}^n x_i$ -t számolja ki, és a ezzel az elemek összegét, majd ezekkel az értékkel tér vissza. A teljes implementáció egyetlen forrásfájlba szervezve, a `orsi_bead1.erl` fájlban található.

3.3 Fordítás

A program forráskódja az `orsi_bead1.erl` fájlban található. A program fordításához követelmény Erlang fordítóprogram megléte a rendszeren. Fordítás után az `orsi_bead1` modul `start()` folyamatának meghívásával futtathatjuk a programot.