Технологично училище "Електронни системи"
Технически университет – София

Операционни системи

Задача 2: Реализация на `wc`

Любомир Чорбаджиев
lchorbadjiev@elsys-bg.org

25 март 2012 г.

1 Условие на задачата

1.1 Основна функционалност (20 точки)

Целта на задачата е е да се реализира стандартната `UNIX` комадна `wc`.

Командата `wc` се използва за преброяване на символите, думите и редовете във файлове. От командния ред програмата получава списък от файлове, а на стандартния изход извежда броя на символите, думите и редовете във всеки един от файловете, както и общия брой символи, думи и редове.

Например, ако са дадени два файла със следното съдържание `a.txt`:

```
aa bb cc
dd ee ff
```

и `b.txt`:

```
1234 1234
567
890
```

то

```
$ wc a.txt
 2  6 18 a.txt
$ wc b.txt
 3  4 18 b.txt
$ wc a.txt b.txt
 2  6 18 a.txt
 3  4 18 b.txt
 5 10 36 total
```

Ако командата не получи аргументи от командния ред, тя започва да чете от стандартния вход (`stdin`).

```
$ wc < a.txt
 2  6 18
```

1.2 Обработка на грешки (20 точки)

Ако някой от аргументите на `wc` не е файл или файлът не може да се отвори, то програмата трябва да изведе съобщение на стандартната грешка (`stderr`). Например, ако бъдат предадени аргументи, които не са файлове

```
wc aa
```

съобщението трябва да бъде оформено по следния начин:

```
$ wc aa
wc: aa: No such file or directory
```

Ако на програмата се предаде име на файл, който потребителят няма права да отвори

```
wc /etc/shadow-
```

съобщението трябва да бъде оформено по следния начин:

```
$ wc /etc/shadow-
wc: /etc/shadow-: Permission denied
```

1.3 Поддръжка на `stdin` (10 точки)

Когато командата получи от командния ред аргумент `-`, тя трябва да интерпретира този аргумент като `stdin`. Например, командата

```
wc a.txt - b.txt
```

трябва да изпълни следната последователност от действия:

1. Да изведе на стандартния изход броя на символите, думите и редовете във файла `a.txt`.
2. Да чете от стандартния вход `stdin` докато не стигне до края на файла и да изведе на стандартния изход броя на символите, думите и редовете прочетени от стандартния вход.
3. Да изведе на стандартния изход броя на символите, думите и редовете във файла `a.txt`.

Съобщението в този случай трябва да бъде оформено по следния начин:

```
$ wc a.txt - b.txt
      2       6      18 a.txt
tt hh ww 44
sd sdf sdf
      2       7      23 -
      3       4      18 b.txt
      7      17      59 total
```

1.4 Поддръжка на опции (20 точки)

Командата `wc` трябва да поддържа следният набор от незадължителни аргументи (опции):

- `-c` — извежда на стандартния изход само броя на символите;
- `-w` — извежда на стандартния изход само броя на думите;
- `-l` — извежда на стандартния изход само броя на редовете.

Програмата трябва да поддържа произволна комбинация от опции. Например:

```
$ wc -c a.txt b.txt
18 a.txt
18 b.txt
36 total
```

```
$ wc -cw a.txt b.txt
 6 18 a.txt
 4 18 b.txt
10 36 total
```

```
$ wc -l a.txt b.txt -c
 2 18 a.txt
 3 18 b.txt
 5 36 total
```

2 Изисквания към решението и оценяване

1. Програмата трябва да бъде написана на езика `C` съгласно ISO/IEC 9899:1999.

2. Правилата за оценяване са следните. Приемаме, че напълно коректна и написана спрямо изискванията програма получава максималния брой точки — 100% или 70 точки. Ако в решението има пропуски, максималният брой точки ще бъде намален съгласно правилата описани по-долу.

3. За работа с файлове трябва да се използва семейството от функции за работа с файлови потоци `FILE` – `fopen()`, `fclose()`, `fread()`, `fwrite()`, `fgetc()`, `fputc()`, `getline()`, `fgets()`, `fputs()` и т.н. Неспазването на това изискване води до намаляване на оценката с 30%.

4. Задължително към файловете с решението трябва да е приложен и `Makefile`. Изпълнимият файл, който се създава по време на компилация на решението, трябва да се казва `wc`.

5. При проверка на решението програмата ви ще бъде компилирана и тествана по следния начин:

   ```
   make
   ./wc a.txt b.txt
   ```

   Предходната процедура ще бъде изпълнена няколко пъти с различни входни данни за да се провери дали вашата програма работи коректно.

6. Реализацията на програмата трябва да спазва точно изискванията. Всяко отклонение от изискванията ще доведе до получаване на 0 точки за съответната част от условието.

7. Работи, които са предадени по-късно от обявеното (или не са предадени), ще бъдат оценени с 0 точки.

8. Програмата ви трябва да съдържа достатъчно коментари. Оценката на решения без коментари или с недостатъчно и/или мъгляви коментари ще бъде намалена с 30%.

9. Всеки файл от решението трябва да започва със следният коментар:

   ```
   //-----------------------------------------
   // NAME: Ivan Ivanov
   // CLASS: XIa
   // NUMBER: 13
   // PROBLEM: #1
   // FILE NAME: xxxxxx.yyy.zzz (unix file name)
   ```

```
// FILE PURPOSE:
//  няколко реда, които описват накратко
//  предназначението на файла
//  ...
//---------------------------------------------
```

Всяка функция във вашата програма трябва да включва кратко описание в следния формат:

```
//---------------------------------------------
// FUNCTION: xxyyzz (име на функцията)
//  предназначение на функцията
// PARAMETERS:
//  списък с параметрите на функцията
//  и тяхното значение
//---------------------------------------------
```

10. Лош стил на програмиране и липсващи заглавни коментари ще ви костват 30%.

11. Програми, които не се компилират получават 0 точки. Под „не се компилират“ се има предвид произволна причина, която може да причини неуспешна компилация, включително липсващи файлове, неправилни имена на файлове, синтактични грешки, неправилен или липсващ `Makefile`, и т.н. Обърнете внимание, че в `UNIX` имената на файловете са “case sensitive”.

12. Програми, които се компилират, но не работят, не могат да получат повече от 50%. Под „компилира се, но не работи“ се има предвид, че вие сте се опитали да решите проблема до известна степен, но не сте успели да направите пълно решение. Често срещан проблем, който спада към този случай, е че вашият Makefile генерира изпълним файл, но той е именуван с име, различно от очакваното (т.е. `wc` в разглеждания случай).

13. Безсмислени или мъгляви програми ще бъдат оценявани с 0 точки, независимо че се компилират.

14. Програми, които дават неправилни или непълни резултати, или програми, в които изходът и/или форматирането се различава от изискванията ще получат не повече от 70%.

15. Всички наказателни точки се сумират. Например, ако вашата програма няма задължителните коментари в началото на файлове и функциите се отнемат 30%, ако няма достатъчно коментари се отнемат още 30%, компилира се, но не работи правилно — още 30%, то тогава резултатът ще бъде: $70 * (100 - 30 - 30 - 30)\% = 70 * 10\% = 7$ точки

16. Работете самостоятелно. Групи от работи, които имат твърде много прилики една с друга, ще бъдат оценявани с 0 точки.