

# Systemy operacyjne

## Lista zadań nr 5

Na zajęcia 9 i 10 listopada 2021

Należy przygotować się do zajęć czytając następujące materiały: [1, rozdziały 3.15, 4-14 – 4.18, 17.2], [2, rozdział 62], [3, rozdziały 4.1, 4.2, 10.6].

**UWAGA!** W trakcie prezentacji należy być gotowym do zdefiniowania pojęć oznaczonych **wytłuszczoną** czcionką.

**Zadanie 1.** Rura `pipe(7)` to **jednokierunkowe** narzędzie do komunikacji międzyprocesowej. Co robi operacja `read(2)` i `write(2)`, jeśli **bufor** rury jest odpowiednio pusty albo pełny? Jakie gwarancje daje nam operacja `write` na rurze, do której pisze wiele procesów – każdy z nich wiersze tekstu nie dłuższe niż «PIPE\_BUF»? Weźmy potok utworzony poleceniem «ps -ef | grep sh | wc -l». Czemu wszystkie procesy należące do potoku zakończą się bez interwencji powłoki, jeśli co najmniej jeden z nich umrze? Kiedy operacje `read` i `write` na rurze zwracają „*short count*”? Jak można połączyć rodzica i dziecko rurą, która została utworzona po uruchomieniu dziecka?

**Zadanie 2.** Zapoznaj się z krytyką interfejsu plików przedstawioną w podrozdziale „*ioctl and fcntl Are an Embarrassment*”<sup>1</sup>. Do czego służy wywołanie systemowe `ioctl(2)`? Zauważ, że stosowane jest głównie do plików **urządzeń znakowych** lub **blokowych**. Na podstawie pliku `ioccom.h`<sup>2</sup> wyjaśnij znaczenie drugiego i trzeciego parametru wywołania `ioctl`. Używając **przeglądarki kodu**<sup>3</sup> jądra NetBSD znajdź definicje operacji «DIOCEJECT», «KIOCTYPE» i «SIOCGIFCONF», a następnie wytłumacz co one robią.

**Komentarz:** Autor zadania zgadza się z autorem krytyki. Czy i Ty widzisz brzydotę tego interfejsu?

**Zadanie 3.** W systemach uniksowych katalog to ciąg bajtów reprezentujący listy rekordów `dirent(3)`. Na podstawie [3, rysunek 10-32] przedstaw reprezentację katalogu, a następnie wyjaśnij jak przebiegają operacje usuwania i dodawania pliku. W pierwszym przypadku rozważ scenariusz, w którym w reprezentacji katalogu za lub przed usuwanym wpisem istnieją **nieużytki**. W drugim, kiedy w pliku katalogu nie udaje się znaleźć wystarczająco dużo miejsca na przechowanie wpisu. Jądro leniwie wykonuje operację **kompaktowania** na katalogach – kiedy opłaca się ją zrobić?

**Zadanie 4.** Korzystając z poleceń «stat» i «ls -lia» zaprezentuj jak jądro systemu operacyjnego trawersuje **ścieżkę bezwzględną** «/usr/bin/cc». Od jakiego numeru **i-węzła** algorytm zaczyna działanie? Skąd sterownik uniksowego systemu plików wie gdzie na dysku znajduje się *i*-ty bajt pliku? Próba utworzenia dowiązania do pliku «/proc/version» kończy się błędem «EXDEV». Czemu nie możemy tworzyć dowiązań do plików znajdujących się w obrębie innych zamontowanych systemów plików?

**Uwaga!** Autor zadania zakłada, że studenci korzystają z systemu plików ext4.

<sup>1</sup><http://www.catb.org/~esr/writings/taoup/html/ch20s03.html>

<sup>2</sup><https://nxr.netbsd.org/xref/src/sys/sys/ioccom.h>

<sup>3</sup><https://nxr.netbsd.org>

Ściągnij ze strony przedmiotu archiwum «so21\_lista\_5.tar.gz», następnie rozpakuj i zapoznaj się z dostarczonymi plikami.

**UWAGA!** Można modyfikować tylko te fragmenty programów, które zostały oznaczone w komentarzu napisem «TODO».

**Zadanie 5.** Program «`listdir`» wypisuje zawartość katalogu w formacie przypominającym wyjście polecenia «`ls -l`». Poniżej można znaleźć przykładowy wydruk, na którym widnieją odpowiednio: plik zwykły, dowiązanie symboliczne, urządzenie znakowe, plik wykonywalny z bitem `set-uid`, jeden katalog z ustawionym bitem `set-gid` i drugi z bitem `sticky`.

```
1 -rw-r--r--  1  cahir  cahir   2964  Fri Nov 15 14:36:59 2019  listdir.c
2 lrwxrwxrwx  1  cahir  cahir    17  Mon Nov  4 11:14:49 2019  libcsapp -> ../csapp/libcsapp
3 crw--w----  1  cahir   tty    4,  2  Tue Nov 12 08:42:33 2019  tty2
4 -rwsr-xr-x  1  root   root   63736  Fri Jul 27 10:07:37 2018  passwd
5 drwxrwsr-x 10  root   staff  4096  Mon Jan  9 13:49:40 2017  local
6 drwxrwxrwt 23  root   root   12288  Fri Nov 15 16:01:16 2019  tmp
```

Uzupełnij kod programu według wskazówek zawartych w komentarzach w kodzie źródłowym. Należy użyć:

- `fstatat(2)` do przeczytania metadanych pliku,
- `major(3)` i `minor(3)` do zdekodowania numeru urządzenia,
- `readlinkat(2)` to przeczytania ścieżki zawartej w dowiązaniu symbolicznym.

Implementacja iterowania zawartości katalogu będzie wymagała zapoznania się ze strukturą «`linux_dirent`» opisaną w podręczniku `getdents(2)`. Wywołanie systemowe «`getdents`» nie jest eksportowane przez bibliotekę standardową, zatem należało je wywołać pośrednio – zobacz plik «`libcsapp/Getdents.c`».

**Zadanie 6.** (Pomysłodawcą zadania jest Tomasz Wierzbicki.)

Program «`primes`» używa [Sita Eratostenesa](#)<sup>4</sup> do obliczania liczb pierwszych z przedziału od 2 do 10000. Proces główny tworzy dwóch potomków wykonujących procedurę «`generator`» i «`filter_chain`», spiętych rurą «`gen_pipe`». Pierwszy podproces wpisuje do rury kolejne liczby z zadanego przedziału. Drugi podproces tworzy łańcuch procesów filtrów, z których każdy jest spięty rurą ze swoim poprzednikiem. Procesy w łańcuchu powstają w wyniku obliczania kolejnych liczb pierwszych. Każdy nowy filtr najpierw wczytuje liczbę pierwszą  $p$  od poprzednika, po czym drukuje ją, a następnie kopiuje kolejne liczby z poprzednika do następnika za wyjątkiem liczb podzielnych przez  $p$ . Program musi poprawnie działać dla argumentu 10000 – w tym przypadku powinno zostać utworzonych  $1229 + 2$  podprocesów.

**Uwaga!** Rozwiązania, które nie zapewniają pochówku umarłym dzieciom lub nie dbają o zamykanie nieużywanych końców rur, są uważane za błędne. Będziemy to sprawdzać poleceniem «`ps`» i «`lsof`».

**Zadanie 7 (2).** (Pomysłodawcą zadania jest Tomasz Wierzbicki.)

Program «`mergesort`» odczytuje ze standardowego wejścia liczbę naturalną  $n$ , po czym czyta  $n$  liczb całkowitych. Program realizuje algorytm sortowania przez scalanie. Proces główny zajmuje się wczytywaniem danych wejściowych i drukowaniem posortowanego ciągu. Żeby posortować liczby, program uruchamia podproces, który wykonuje procedurę «`Sort`». Rozmawia z nim przy pomocy gniazda domeny uniksowej `unix(7)`, które tworzy z użyciem `socketpair(2)`, czyli lokalnej dwukierunkowej metody komunikacji międzyprocesowej. Jeśli proces sortujący otrzyma od rodzica pojedynczą liczbę, to natychmiast odsyła ją swojemu rodzicowi i kończy działanie. Jeśli dostanie więcej liczb, to startuje odpowiednio lewe i prawe dziecko, po czym za pomocą procedury «`SendElem`» przesyła im liczby do posortowania. Następnie wywołuje procedurę «`Merge`», która odbiera od potomków posortowane ciągi, scala je i wysyła do procesu nadrzędnego.

Twoim zadaniem jest uzupełnienie procedury «`Sort`» tak by wystartowała procesy potomne i uruchomiła procedury «`SendElem`» i «`Merge`». Należy odpowiednio połączyć procesy z użyciem gniazd oraz zamknąć niepotrzebne gniazda w poszczególnych procesach. Posługując się rysunkiem wyjaśnij strukturę programu. Kiedy tworzysz podprocesy i gniazda? Kiedy zamykasz niepotrzebne gniazda? Jak wygląda przepływ danych?

Skrypt «`gen-nums.py`» przyjmuje w linii poleceń  $n$ , czyli liczbę elementów do wygenerowania. Po uruchomieniu drukuje  $n$  na standardowe wyjście, po czym drukuje  $n$  losowych liczb całkowitych. Produkowane dane są w odpowiednim formacie do wprowadzenia do programu «`mergesort`».

**Uwaga!** Wszystkie procesy **muszą** działać w stałej pamięci. Rozwiązania nie spełniające tego warunku są niepoprawne!

<sup>4</sup>[https://en.wikipedia.org/wiki/Sieve\\_of\\_Eratosthenes](https://en.wikipedia.org/wiki/Sieve_of_Eratosthenes)

## Literatura

- [1] „*Advanced Programming in the UNIX Environment*”  
W. Richard Stevens, Stephen A. Rago  
Addison-Wesley Professional; 3rd edition; 2013
- [2] „*The Linux Programming Interface: A Linux and UNIX System Programming Handbook*”  
Michael Kerrisk  
No Starch Press; 1st edition; 2010
- [3] „*Systemy operacyjne*”  
Andrew S. Tanenbaum, Herbert Bos  
Helion; wydanie czwarte; 2015
- [4] „*Operating Systems: Three Easy Pieces*”  
Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau  
<https://pages.cs.wisc.edu/~remzi/OSTEP/>