| Systemy operacyjne 2 | | |
|---|---|---|
| Informatyka S1 | Semestr 4 | 2018/2019 |
| *Laboratorium 3* | | |

Wstęp teoretyczny:

- Tworzenie procesu (fork), kończenie procesu (exit), synchronizacja procesów (wait, waitpid), pobieranie identyfikatorów procesów (getpid, getppid).
- Zależności między procesami macierzystymi i potomnymi, różne przypadki zakończenia procesów (np. przedwczesne zakończenie rodzica, zakończenie potomka bez wywołania wait ze strony rodzica).
- Zastąpienie procesu obrazem nowego procesu (rodzina funkcji exec).
- Analiza informacji zwracanych przez wait (identyfikator i status zakończenia).

Zadanie:

- Napisać program uruchamiany z co najmniej jednym argumentem. Podstawowe wywołanie programu będzie wymagało podania jako argumentu dowolnego łańcucha znaków (np. **abcd**). Program będzie tworzył 2 procesy potomne. Każdy nowo utworzony proces potomny za pomocą jednej z funkcji **exec** jeszcze raz wywołuje ten sam program przekazując jako argument połowę otrzymanego łańcucha (czyli np. pierwszy proces **ab**, drugi **cd**). Proces, który otrzymane łańcuch o długości ***1***, nie tworzy kolejnych procesów. Wszystkie procesy (macierzysty i potomne na wszystkich poziomach) czekają na zakończenie wszystkich swoich bezpośrednich potomków, a następnie wyświetlają swój identyfikator oraz wszystkie łańcuchy (argumenty) na ścieżce od pierwszego uruchomionego procesu aż do procesu aktualnego. Należy rozważyć możliwość przekazywanie dodatkowych argumentów wywołania **exec**.
- Uruchomienie programu z argumentem **abcd** może dać przykładowo następujący wynik:

```
$./program  abcd
26062 abcd ab a
26063 abcd ab b
26060 abcd ab
26065 abcd cd d
26064 abcd cd c
26061 abcd cd
26059 abcd
```

- Pierwszy łańcuch powinien mieć długość będącą potęgą 2, jeżeli długość będzie inna należy go albo uzupełnić albo skrócić (do wyboru).
- Do pobierania argumentów nie używamy funkcji **getopt**, należy wykorzystać bezpośrednio zmienne przekazywane do f-cji main (zwyczajowe nazwy argc i argv).

- Zwrócić uwagę na to, aby:
    - procesy nie były wykonywane sekwencyjnie (czyli nie na zasadzie naprzemiennego wywoływania funkcji fork i wait),
    - sprawdzać poprawne zakończenie funkcji wait.

Uwaga! Kod źródłowy programu (1 plik) po oddaniu prowadzącemu zajęcia laboratoryjne musi zostać przesłany na adres so2@zut.edu.pl :

- plik z kodem źródłowym musi mieć nazwę:

  ***numer_indeksu*.so2.lab03.main.c**

  (np. 666.so2.lab03.main.c),
- plik musi zostać wysłany z poczty uczelnianej (**zut.edu.pl**),
- nagłówek maila musi mieć postać:

  **SO2 IS1 XXXY LAB03**

  gdzie XXXY to numer grupy laboratoryjnej (np. SO2 IS1 210C LAB03),
- **w pierwszych pięciu liniach** pliku z kodem źródłowym w komentarzach (każda linia komentowana osobno) musi znaleźć się:
    - informacja identyczna z zamieszczoną w nagłówku maila,
    - imię i nazwisko wysyłającego oraz
    - adres email, z którego wysłał wiadomość,
    - komenda jaką należy skompilować program z linii komend,
    - komenda jaką należy uruchomić skompilowany program z obydwoma przełącznikami

  czyli np.:

```
// SO2 IS1 210C LAB03
// Jan Nowak
// jn666@zut.edu.pl
// gcc  666.so2.lab03.main.c -o lab03
// ./lab03 abcde
```

- email **nie powinien** zawierać żadnej treści (tylko załącznik).

Dostarczone kody źródłowe będą analizowane pod kątem wykrywania plagiatów. Niewysłanie wiadomości, wysłanie jej w **formie niezgodnej** z powyższymi wymaganiami lub wysłanie pliku, który nie będzie się kompilował i poprawnie uruchamiał będzie traktowane jako brak programu i skutkowało otrzymaniem za niego oceny niedostatecznej.