

# Tworzenie procesów. Środowisko procesu, sterowanie procesami.

## Zadanie 1 (20%)

Napisz program, który przyjmuje jeden argument: `argv[1]`. Program ma utworzyć `argv[1]` procesów potomnych. Każdy proces potomny ma wypisać na standardowym wyjściu w jednym wierszu dwa identyfikatory: identyfikator procesu macierzystego i swój własny. Na końcu standardowego wyjścia proces macierzysty ma wypisać `argv[1]`. Wskazówka: aby program na pewno wypisywał `argv[1]` jako ostatni wiersz standardowego wyjścia, należy użyć funkcji systemowej `wait`.

## Zadanie 2 (20%)

Napisz program, który przyjmuje jeden argument: `argv[1]` — ścieżkę katalogu. Program ma wypisać na standardowym wyjściu swoją nazwę (bez znaku nowego wiersza), korzystając z funkcji `printf`, a następnie wykonać program `/bin/ls` z argumentem `argv[1]`, korzystając z funkcji `execl`.

Ważne: funkcje biblioteki standardowej języka C buforują dane zapisywane na wyjściu. Zawartość bufora jest wypisywana na wyjściu po każdym wierszu lub po każdym `BUFSIZ` bajtach. Ostatni bufor jest wypisywany automatycznie podczas kończenia procesu. W tym programie proces nie kończy się przed wywołaniem funkcji `execl`, a bufor, który należy do segmentu danych użytkownika procesu, jest zamazywany, zanim jego zawartość mogła zostać wypisana. Aby wymusić brak buforowania danych wyjściowych, można użyć funkcji `setbuf(stdout, NULL)`. Można też wypisać zawartość bufora tuż przed wywołaniem funkcji `execl`, używając funkcji `fflush(stdout)`.

Proszę przetestować program, podając jako `argv[1]` katalogi zawierające 0 plików, kilka plików i wiele plików.

## Zadanie 3 (60%)

3. Napisz program, który przyjmuje dwa argumenty: `argv[1]` — ścieżkę katalogu i `argv[2]` — łańcuch znaków nie dłuższy niż 255 bajtów. Program ma rekurencyjnie przeglądać katalog `argv[1]` i drzewo jego podkatalogów, tworząc drzewo procesów tego samego kształtu, co drzewo katalogów, po jednym procesie dla każdego katalogu. Każdy proces powinien przeglądać wszystkie pliki w katalogu.

- Jeśli przeglądany plik jest katalogiem, to proces powinien utworzyć proces

potomny dla tego katalogu poprzez rekurencyjne wywołanie funkcji przeglądającej katalog.

- Jeśli przeglądany plik nie jest katalogiem, można go otworzyć do odczytu i jego treść pliku zaczyna się od łańcucha znaków `argv[2]`, to proces ma na standardowym wyjściu wypisać w jednym wierszu: ścieżkę pliku (nie nazwę pliku) i swój identyfikator.

Wskazówki:

- Żeby sprawdzić, czy plik jest katalogiem, proszę użyć funkcji systemowej `fstat` lub `stat`.

- Proszę napisać program tak, żeby sprawdzał przy wywołaniu każdej funkcji systemowej, czy zwróciła ona błąd. Jeśli błąd uniemożliwia dalsze działanie programu, powinien on wypisać rodzaj błędu, korzystając z funkcji `perror`, i zakończyć działanie.

- Przy wywołaniach rekurencyjnych proszę pomijać katalogi . i .., ale nie pomijać pozostałych katalogów ani plików zwykłych, których nazwa zaczyna się od kropki.

- Procesy nie muszą czekać na zakończenie swoich procesów potomnych.

- Długość ścieżki do dowolnego pliku, wraz z końcowym znakiem `\0`, nie przekracza stałej `PATH_MAX`, zdefiniowanej w pliku nagłówkowym `limits.h`

- Proszę przetestować program, podając jako `argv[1]` między innymi ścieżki ..  
../. ~ /root, a jako `argv[2]` między innymi łańcuchy znaków `\#include asdf`  
zadanie

Miłego programowania :)