Oskar Sobczyk

Systemy operacyjne – problem palaczy

Spis treści

[2 Opis zadania 3](#_Toc472507583)

[2.1 Problem palaczy tytoniu (ang. The Cigarette-Smokers Problem) 3](#_Toc472507584)

[2.2 Rozwiązanie problemu 3](#_Toc472507585)

[3 Omówienie kodu źródłowego 4](#_Toc472507586)

[3.1 main.c 4](#_Toc472507587)

[3.2 valet.c 4](#_Toc472507588)

[3.3 smoker.c 5](#_Toc472507589)

[4 Testowanie 5](#_Toc472507590)

# Opis zadania

## Problem palaczy tytoniu (ang. The Cigarette-Smokers Problem)

Rozważmy problem, w którym występuje lokaj oraz trzech palaczy.Każdy palacz nieustannie skręca i wypala papierosy. Aby zapalić papierosa, palacz potrzebuje trzech rzeczy: tytoniu, papieru i zapałek. Lokaj posiada nieskończony zapas papieru, tytoniu i zapałek, a palacze: pierwszy ma tytoń, drugi papier, trzeci zapałki. Lokaj losowo wybiera dwa z posiadanych przez siebie zasobów, a palacz posiadający trzeci może wypalić papierosa. Lokaj ,,nie wie", który palacz posiada jaki zasób. Agent czeka aż palacz skończy palić i ponownie wystawia dwa różne, losowe składniki. Cykl się powtarza.

## Rozwiązanie problemu

#kropki po numerach

W celu rozwiązania zadania potrzebne będzie utworzenie czterech procesów (trzech procesów palaczy oraz jednego procesu lokaja). Procesy będą synchronizowane za pomocą czterech semaforów:

***sem\_valet*** informuje, czy lokaj może udostępnić składniki#przecinki kropki

***sem\_tobacco*** informuje, czy palacz z tytoniem może skręcić papierosa

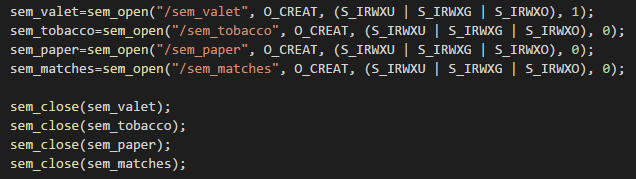
***sem\_paper*** informuje, czy palacz z papierem może skręcić papierosa

***sem\_matches*** informuje, czy palacz z zapałkami może skręcić papierosa

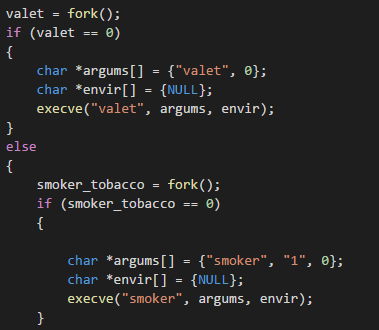
Proces główny na początku swojego działania utworzy i zamknie semafory. Następnie za pomocą polecenia ***fork()*** utworzy cztery procesy. W trzech z nich zostaną wywołane programy palaczy a w jednym lokaja. Proces lokaja czeka, aż ***sem\_valet*** zostanie zwolniony, a następnie losuje liczbę z przedziału [0,2] odpowiadającą kombinacji brakujących składników i zwalnia semafor odpowiedniego palacza. Program palacza czeka, aż odpowiedni semafor zostanie zwolniony i jeśli jest dostępny, to wypisuje komunikat o skręceniu papierosa, a następnie zwalnia ***sem\_valet***.

# Omówienie kodu źródłowego

## Proces main.c

W początkowej części main.c następuje utworzenie oraz zamknięcie semaforów.

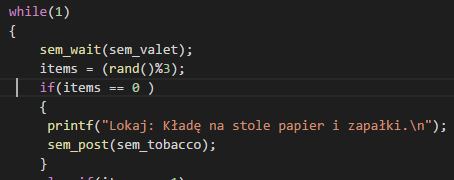
W dalszej części zajmujemy się klonowaniem procesu za pomocą funkcji ***fork().***



Jeśli znajdujemy się w procesie macierzystym(***fork()*** == 1) wykonujemy program tworząc kolejne procesy palaczy. Jeśli jesteśmy w procesie potomnym, wywołujemy program valet.c. W sposób identyczny działa tworzenie palaczy z tą różnicą, że wywołujemy program smoker.c.

## Proces valet.c

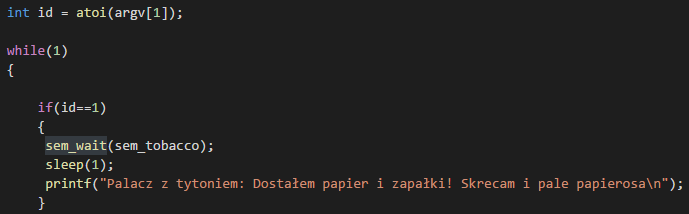
Odpowiada za proces lokaja. Jego działanie jest bardzo proste.



Program w nieskończonej pętli czeka, aż semafor ***sem\_valet*** zostanie zwolniony. Następnie losuje liczbę z przedziału [0,2] i zwalnia semafor odpowiadający wylosowanemu palaczowi.

## Proces smoker.c

Odpowiada za procesy palaczy.



Zmienna***id*** odpowiada za numer identyfikacyjny palacza. W zależności od wartości ***id*** program będzie czekał aż lokaj położy na stole odpowiednie składniki. Czyli czeka aż zwolni się odpowiedni semafor i wypisze odpowiedni komunikat. Pomiędzy wypisywaniem komunikatów znajduje się funkcja ***sleep()***, która ma za zadanie wydłużyć czas pomiędzy kolejnymi palaczami.

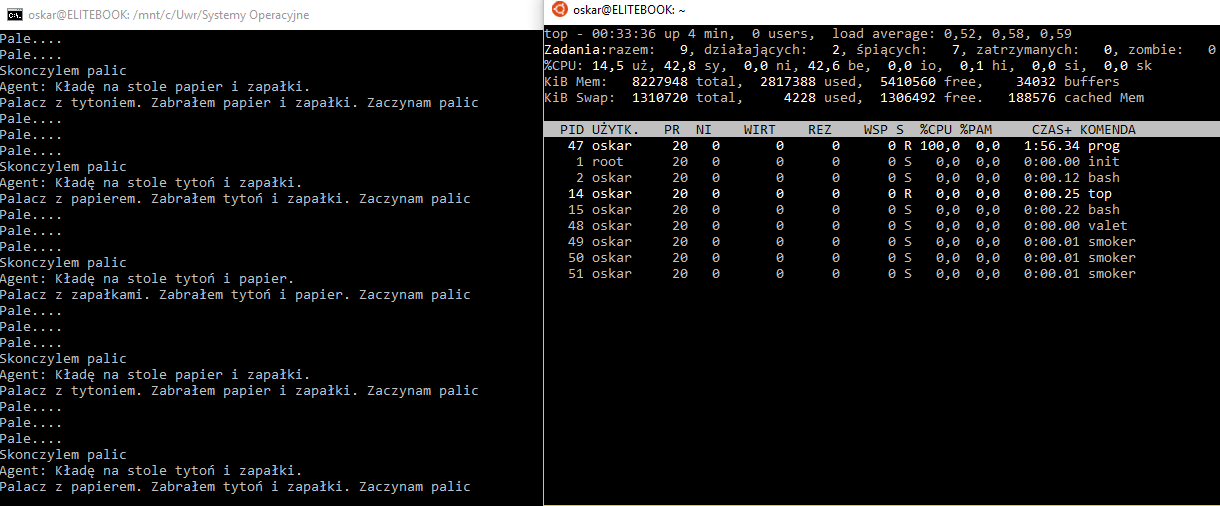


Na końcu zwalniamy semafor lokaja umożliwiając mu dalsze wykładanie składników.

# Testowanie

Program powinien być testowany w systemie Linux.

Przed uruchomieniem należy użyć polecenia ***make***, a następnie uruchomić program za pomocą polecenia ***./main.*** Za pomocą programu top sprawdzimy, jakie procesy działają w tle.



Program działa poprawnie. Tworzy 4 procesy a komunikaty wyświetlają się poprawnie. Nie zachodzi sytuacja, w której lokaj wykłada składniki przed zakończeniem palenia albo palacz zaczyna palić bez wszystkich składników.