Zadanie 3 – semafory

W czasie wyścigu każdy z N bolidów co pewien czas zjeżdża do alei serwisowej w celu zatankowania i zmiany opon. W alei serwisowej znajduje się K (K < N) stanowisk serwisowych (każdy bolid może skorzystać z dowolnego stanowiska). W celu dostania się do stanowiska serwisowego oraz powrotu na tor bolid korzysta z pasa serwisowego. W celu zwiększenia bezpieczeństwa zakładamy, że w danej chwili na pasie serwisowym może znajdować się co najwyżej jeden bolid. Aby zwiększyć przepustowość alei serwisowej bolid po dojechania na stanowisko serwisowe musi zwolnić pas serwisowy aby inne bolidy mogły w tym czasie zjeżdżać i wyjeżdżać. Gdy liczba bolidów korzystających ze stanowisk serwisowych jest mniejsza niż P (0 < P < K) priorytet w dostępie do pasa serwisowego mają bolidy wjeżdżające do alei serwisowej (bolidy, które chcą zatankować). W przeciwnym wypadku priorytet mają bolidy, które zakończyły tankowanie i oczekują na możliwość powrotu na tor. Należy zapisać algorytm bolidu, który funkcjonuje według schematu: jazda – zjazd – tankowanie – wyjazd – jazda ….. Program powinien trwać aż L tankowań zostanie wykonanych (w sumie lub przez każdy z bolidów, według uznania), następnie powinien zostać poprawnie zakończony.

Wskazówki do koncepcji:

- Należy uwzględnić, że każda z czynności bolidu trwa niezerowy czas (symulacja za pomocą sleep()).
- W koncepcji należy opisać:
    - Sposób wykorzystania semaforów do realizacji zadania (ile? Jakich?)
    - Zdefiniować potrzebne zmienne współdzielone
    - Algorytm bolidu w postaci pseudokodu ze szczególnym uwzględnieniem operacji na semaforach
    - Sposób zakończenia programu
    - Scenariusze testowe – używając różnych wartości czasów symulujących czynności bolidu można doprowadzić do różnych sytuacji: np. długi czas tankowania spowoduje zapełnienie się wszystkich stanowisk co pozwoli nam sprawdzić czy nie dochodzi do zakleszczeń.

Użyteczne funkcje:

Operacje na semaforach (zwrócić uwagę na zestawy semaforów) -
http://www.cs.cf.ac.uk/Dave/C/node26.html

-**semget**; wywołanie **tworzy nowy zestaw semaforów**, które będą wykorzystane przez procesy
-**semctl**; wywołanie wykorzystane do **odczytywania** wartości danego semafora
-**semop**; wywołanie wykorzystane do **wykonywania operacji** na semaforach – inkrementacji i dekrementacji

Pamięć dzielona - pamięć która może być dzielona między procesy/wątki programu, służy do komunikacji między procesami oraz zapobiega redundancji danych.

-**shmget**; wywołanie wykorzystane do **alokowania segmentu pamięci** współdzielonej
-**shmat**; wywołanie wykorzystane do **dołączania segmentu pamięci** współdzielonej do danego procesu
-**shmdt**; wywołanie wykorzystane do **odłączania segmentu pamięci** współdzielonej od danego procesu

-**fork**; wywołanie wykorzystane do **uruchamiania nowych procesów** z poziomu procesu-ojca