IPC - pamieć wspólna, semafory

Przydatne funkcje:

System V:

```
<sys/shm.h> <sys/ipc.h> - shmget, shmclt, shmat, shmdt
```

POSIX:

```
<sys/mman.h> - shm_open, shm_close, shm_unlink, mmap, munmap
```

Zadanie

Wykorzystując semafory i pamięć wspólną z IPC Systemu V napisz program symulujący działanie pizzerii.

W pizzerii znajduje się piec, który może pomieścić jednocześnie 5 pizz. Aby włożyć lub wyjąć pizze należy skorzystać z małego okienka, które jednocześnie może obsługiwać tylko jedna osoba. Znajduje się tam również stół do wysyłki na którym mieści się maksymalnie 5 pizz.

W pizzeri pracuje N kucharzy, którzy w pętli wykonują:

1) Losuje typ pizzy (n) w przedziale 0-9 i następnie ją przygotowuje (1-2s).
Wypisanie komunikatu: (pid timestamp) Przygotowuje pizze: n.
gdzie pid to PID procesu pracownika, timestamp to aktualny czas (z dokładnością do milisekund)

2) Umiesza pizze w piecu (wpisuje n).
Wypisanie komunikatu: (pid timestamp) Dodałem pizze: n. Liczba pizz w piecu: m.

3) Czeka (4-5s).

4) Wyjmuje pizze i umieszcza ją na stole do wysyłki (wpisuje n).
Wypisanie komunikatu: (pid timestamp) Wyjmuję pizze: n. Liczba pizz w piecu: m. Liczba pizz na stole: k.

Dostawcy, których jest M, następnie je rozwożą:

1) Dostawca pobiera pizze ze stołu.
Wypisanie komunikatu: (pid timestamp) Pobieram pizze: n Liczba pizz na stole: k.

2) Dojeżdża do klienta (4-5s).

3) Dostarcza pizze.
Wypisanie komunikatu: (pid timestamp) Dostarczam pizze: n.

4) Wraca (4-5s).

Rozmiary tablic pieca oraz stołu do wysyłki (w pamięci wspólnej) są ograniczone i ustalone na etapie kompilacji. Tablice te są indeksowane w sposób cykliczny - po dodaniu pizzy na końcu tablicy, kolejna pizza dodawana jest od indeksu 0. Korzystając w odpowiedni sposób z semaforów należy zagwarantować, że liczba pizz nie przekroczy rozmiaru tablicy oraz że każda tablica nie będzie modyfikowana przez kilka procesów równocześnie. W pamięci wspólnej oprócz tablic można przechowywać także inne dane dzielone pomiędzy procesami. Kolejni pracownicy są uruchamiani w pętli przez jeden proces macierzysty (za pomocą funkcji fork oraz exec).

Sprawdź dla 2 różnych wartości *N,M* (dla $N,M < 5$ i $N,M > 5$) i zapisz w *wnioski.txt*

Zrealizuj powyższy problem synchronizacyjny, wykorzystując mechanizmy synchronizacji procesów oraz pamięć współdzieloną ze standardu:

1. IPC - System V (50%)
2. IPC - Posix (50%)