Etap	1	2	3	4	Suma
Punkty	3	7	6	5	21
Wynik					

L2: Uber

Twoim zadaniem jest stworzenie symulacji systemu do zawiania usług transportu samochodowego za pomocą kolejek wiadomości POSIX.
Program będzie przyjmować dwa argumenty N ($N \geqslant 1$) - będący liczbą kierowców biorących udział w symulacji, oraz T - czas trwania symulacji w sekundach ($T \geqslant 5$).

Etapy:

1. [3 pkt.] Proces główny tworzy N procesów kierowców i czeka na ich zakończenie. Kierowcy losują i wypisują swoją pozycję startową z kwadratu $[-1000, 1000]^2$ i kończą działanie. Przyjmujemy że wszystkie wektory pozycji mają współrzędne całkowitoliczbowe.

2. [7 pkt.] Wątek główny tworzy kolejkę na nowe zadania o nazwie `uber_tasks`, a pracownicy czekają na nowe zadanie poprzez czytanie z tej kolejki. Gdy otrzymają zadanie (cztery liczby całkowitoliczbowe z zakresu [-1000, 1000]), zmieniają swoją pozycję na koniec trasy, drukują zadanie na standardowym wyjściu, śpią d (odległość od obecnej pozycji do punktu startowego trasy + długość trasy w metryce miejskiej) ms i czekają na kolejne zadanie. Główny proces tworzy nowe zadanie co 500 - 2000 ms, jeżeli w kolejce znajduje się już 10 zadań, to zadanie jest odrzucane a na stderr wypisywany jest stosowny komunikat.

3. [6 pkt.] Kierowcy po wykonaniu kursu wysyłają informację o wykonanym zadaniu do głównego procesu za pomocą indywidualnych kolejek o nazwie `uber_results_[PID]`. Kierowcy wysyłają tą kolejką długość przebytego kursu w metryce miejskiej. Główny proces odbiera wyniki i wypisuje je na standardowym wyjściu: `The driver [PID] drove a distance of [distance]`.

4. [5 pkt.] Główny proces po stworzeniu dzieci i kolejek ustawia alarm na T sekund. Po otrzymaniu `SIGALRM` przestaje wysyłać nowe zadania, rozsyła do dzieci wiadomość o zwiększonym priorytecie i zamyka kolejkę. Kierowcy wykrywają, że główny proces zamknął kolejkę, poprzez odczytanie wiadomości o wyższym priorytecie i kończą swoją pracę. Wszystkie zasoby są poprawnie zwalniane, a kolejki zamykane i usuwane.