Zaawansowane zastosowania kart graficznych

Laboratorium 4

Zadania demonstrujące użycie biblioteki cuRand i zarządzanie pamięcią przez thrust.

1. Napisz program, który wykonuje rzuty kostką d6 dla zadanej liczby kostek (np. poprzez zmienną) i wypisuje wynik na konsoli. Rzuty kostką powinny być generowane równolegle. W ćwiczeniu wykorzystaj wektory thrust do zarządzania pamięcią.
2. Napisz program, który testuje generator liczb pseudolosowych zgodnych z ustandaryzowanym rozkładem normalnym. W ćwiczeniu wykorzystaj wektory thrust do zarządzania pamięcią. Program powinien
    a. Wygenerować dużo liczb pseudolosowych (czytaj: setki tysięcy)
    b. Obliczyć ich średnią (średnia powinna wyjść bliska 0)
        i. oblicz sumę za pomocą reduce
        ii. podziel przez liczbę wygenerowanych wartości
    c. Obliczyć odchylenie standardowe (powinno wyjść bliskie 1):
        i. Za pomocą własnego kernela odjąć od każdej wartości z a) średnią obliczoną w b).
        ii. Różnicę podnieść do kwadratu (w tym samym kernelu)
        iii. Obliczyć średnią z tych kwadratów analogicznie jak w b)
        iv. Obliczyć pierwiastek z tej średniej (na CPU)
3. Napisz program szukający "po omacku" minimum funkcji. W ćwiczeniu wykorzystaj wektory thrust do zarządzania pamięcią.
    a. Zdefiniuj jakąkolwiek jednoargumentową funkcję np. $f(x)=x^2+2x+1$. Funkcja powinna być zadeklarowana jako `__device__`, żeby mogła być wykonywana z poziomu kernela.
    b. Zdefiniuj zakres testowania, np: [-10;10] (będziemy szukać minimum w przedziale od -10 do 10). W programie chodzi po prostu o przypisanie początku i końca przedziału do jakichś zmiennych.
    c. Zaalokuj dwie tablice *A* i *B* (wektory thrust) o rozmiarze *n*, gdzie *n* to jakaś duża liczba (czyt: setki tysięcy).
    d. Uruchom *n* wątków. Każdy wątek w pętli losuje 10 (parametr - ustalany w jakiejś zmiennej, czyli niekoniecznie 10) wartości *x* z przedziału określonego w punkcie b) z rozkładem jednorodnym i dla każdej wartości oblicza wartość funkcji. Wątek powinien zapamiętać najniższą obliczoną wartość funkcji i odpowiadającą wartość *x*.
    e. Każdy wątek zapisuje odnalezione minimum i odpowiadające *x* odpowiednio w tablicach *A* i *B*.
    f. Wykorzystując *reduce* znajdź najmniejszą spośród wartości w tablicy *A*.
    g. Wyszukaj iterator wskazujący na tą wartość za pomocą funkcji *find*:
        http://docs.thrust.googlecode.com/hg/group__searching.html
        Niech odnaleziony iterator będzie zapisany w zmiennej *it*. Wówczas *it-A.begin()* to indeks tej wartości w tablicy, a *B[it-A.begin()]* to wartość *x*, dla której funkcja *f* ma minimum.