Лабораторная работа 4.2

Разработка приложений для вычислений на GPU

*Задание 2 на лабораторную работу*

1. В приложении из лабораторной работы 4.1 заменить тестовый пример на решение задачи из приведенных вариантов.
2. Предусмотреть возможность проведения расчетов с использованием одного из типов памяти: разделяемой, текстурной, константной. Сравнить скорость расчетов при использовании только глобальной памяти.

*Средства выполнения задания*

Среда программирования MS VISUAL STUDIO (язык C#, альтернатива: C++), GPU NVidia с поддержкой CUDA (альтернативы: GPU AMD с поддержкой AMD FireStream, OpenCL).

*Литература*

*Tumakov\_\_\_Tekhnologiya\_programmirovaniya\_CUDA.pdf*

*Варианты заданий*

1. Расчет интеграла ниже от заданной функции на заданном отрезке от *a* до *b* методом прямоугольников. Значения *a* и *b* задаются на форме.

.

1. Расчет интеграла ниже от заданной функции на заданном отрезке от *a* до *b* по формуле Симпсона. Значения *a* и *b* задаются на форме.



1. Расчет числа Пи по формуле ниже. Шаг разбиения задается на форме.

.

1. Расчет произведения двух квадратных матриц A x B = C. Размерность матриц N задается на форме. Элементы матриц заполняются произвольными вещественными числами и отображаются на форме.
2. Расчет произведения двух прямоугольных матриц A x B = C. Размерности матриц M x N задаются на форме. Элементы матриц заполняются произвольными натуральными числами и отображаются на форме.
3. Поиск всех простых чисел от 1 до N. Число N задается на форме.
4. Сумма всех четных и всех нечетных чисел от 1 до N. Число N задается на форме.
5. Поиск минимального элемента в массиве чисел размерности N. Число N задается на форме.
6. Поиск максимального элемента в массиве чисел размерности N. Число N задается на форме.
7. Поиск минимума функции от одной переменной на заданном отрезке. Отрезок разделить на одинаковые части между процессами. Поиск минимума выполнять методом перебора, задав на всех процессах одинаковое число узлов, в которых вычисляются значения функции. Переслать локальные минимумы на нулевой процесс, при пересылке найти глобальный минимум.
8. Поиск максимума функции от одной переменной на заданном отрезке. Отрезок разделить на одинаковые части между процессами. Поиск максимума выполнять методом перебора, задав на всех процессах одинаковое число узлов, в которых вычисляются значения функции. Переслать локальные максимумы на нулевой процесс, при пересылке найти глобальный максимум.
9. Поиск минимального элемента квадратной матрицы. Матрицу разделить на полосы по числу процессов программы. Каждый процесс отыскивает локально минимальный элемент в своей полосе, номер строки и номер столбца. Результаты поиска минимума собираются на нулевом процессе, определяется глобально минимальный элемент, его номер строки и столбца.
10. Поиск максимального элемента квадратной матрицы. Матрицу разделить на полосы по числу процессов программы. Каждый процесс отыскивает локально минимальный элемент в своей полосе, номер строки и номер столбца. Результаты поиска минимума собираются на нулевом процессе, определяется глобально минимальный элемент, его номер строки и столбца.
11. Поиск обратной матрицы B = A-1. Размерность матриц N задается на форме. Элементы исходной матрицы заполняются произвольными целыми числами и отображаются на форме.
12. Расчет евклидовой нормы вектора (путем скалярного умножения его на себя и вычисления корня из произведения). Исходный вектор задается на форме.
13. Расчет скалярного произведения двух векторов.
14. Проверка двух заданных матриц A и B на коммутируемость (их произведение не зависит от порядка умножения: AB = BA).
15. Расчет числа Пи по формуле ниже. Шаг разбиения задается на форме. Предусмотреть возможность остановки/продолжения расчетов.

.

1. Проверка, является ли данная квадратная матрица ортогональной, путем умножения на транспонированную (не выделяя под нее дополнительной памяти) и сравнения результата с единичной матрицей).
2. Расчет дзета-функции Римана (сумма ряда из обратных степеней).
3. Сложение двух матриц M x N, заданных как вектор векторов (float a[M][N], b[M][N]). Матрицы задаются на хосте, после копируются на девайс, и там производится сложение.