***Задание 24. Сложение векторов в CUDA.***

Напишите CUDA-программу, которая вычисляет сумму двух векторов размера n. Используйте следующие формулы:

$$A = (a_1, a_2, \dots, a_n)$$
$$B = (b_1, b_2, \dots, b_n)$$
$$C = (a_1 + b_1, a_2 + b_2, \dots, a_n + b_n)$$

**Входные данные:** целое число $n \geq 1, n$ целых элементов вектора $A$ и $n$ целых элементов вектора $B$.

**Выходные данные:** $n$ целых элементов вектора $C$.

**Пример входных и выходных данных**

| Входные данные | Выходные данные |
|---|---|
| 4<br>1 3 4 8<br>5 4 3 0 | 6 7 7 8 |

***Указания к заданию 24. Сложение векторов в CUDA.***

1. Создайте проект с поддержкой CUDA (см. задание 8).
2. Ознакомьтесь со встроенными переменными:

   **threadIdx.x**, **threadIdx.y**, **threadIdx.z** - индексы нити в блоке

   **blockIdx.x**, **blockIdx.y**, **blockIdx.z** – индексы блока в гриде

   **blockDim.x**, **blockDim.y**, **blockDim.z** – размеры блоков в нитях

   **gridDim.x**, **gridDim.y**, **gridDim.z** – размеры грида в блоках
3. Напишите функцию-ядро для вычисления элемента результирующего вектора на основе элементов входных векторов. Для вычисления уникального идентификатора нити используйте следующую формулу:

   $$threadID = blockDim.x * blockIdx.x + threadIdx.x$$
4. Ознакомьтесь с функциями выделения и копирования памяти CUDA:

   `cudaError_t` **`cudaMalloc`**`(void** devPtr, size_t size)`

- Выделяет size байтов линейной памяти на устройстве и возвращает указатель на выделенную память в devPtr. Память не обнуляется
- Адрес памяти выровнен по 512 байт

```
cudaError_t cudaFree(void** devPtr)
```

- Освобождает память устройства на которую указывает devPtr

```
cudaError_t cudaMemcpy(void* dst, const void* src, size_t count, cudaMemcpyKind kind )
```

- Копирует count байтов из памяти, на которую указывает src в память на которую указывает dst, kind указывает направление передачи
  - cudaMemcpyHostToHost – копирование между двумя областями памяти на хосте
  - cudaMemcpyHostToDevice – копирование с хоста на устройство
  - cudaMemcpyDeviceToHost – копирование с устройства на хост
  - cudaMemcpyDeviceToDevice – между двумя областями памяти на устройстве
- Вызов cudaMemcpy() с kind, не соответствующим dst и src, приводит к непредсказуемому поведению

5. Создайте входные и выходные массивы, выполните копирование входных массивов на устройство.
6. Для вычисления размеров грида используйте формулу:

$$gridDim = dim3\left(\frac{n-1}{blockDim} + 1\right)$$

7. Запустите ядро, выполните копирование результирующего массива на хост и выведите его на экран.

***8.*** Скомпилируйте и запустите ваше приложение. Убедитесь, что выводится верный результат.