Lab10 – CUDA Reduction

ex.1 Design a CUDA program to find the maximum value in the array.

```
#include <stdio.h>
#include <unistd.h>
#include <pwd.h>

#define N 10
#define THREADS 10

__global__ void findmax(float *A, float *max)
{
    // code segments are given in the lecture
}

int main()
{
    float A[N], *A_d, max, *max_d;
    int i, dev;

    dim3 dimBlock(THREADS);
    dim3 dimGrid((N+dimBlock.x-1)/dimBlock.x);

    dev = (getpwuid(getuid())->pw_name[3]-'0')%2? 1: 0;
    cudaSetDevice(dev);

    srand(1);
    for (i=0; i<N; i++) {
        A[i] = rand() % 999;
        printf("%2.1f ", A[i]);
    }
    printf("₩n");

    cudaMalloc((void **) &A_d, sizeof(float)*N);
    cudaMemcpy(A_d, A, sizeof(float)*N, cudaMemcpyHostToDevice);

    cudaMalloc((void **) &max_d, sizeof(float));

    findmax<<<dimGrid, dimBlock>>>(A_d, max_d);

    cudaMemcpy(&max, max_d, sizeof(float), cudaMemcpyDeviceToHost);

    printf("%f₩n", max);

    cudaFree(A_d);
    cudaFree(max_d);
}
```

ex.2 Design a CUDA program to find the maximum value in the array. Compare the red-colored statements with the statements in the ex1 program

Submit the program(max.cu) when you are done.

```
#include <stdio.h>
#include <unistd.h>
#include <pwd.h>

#define N 40
#define THREADS 10

__global__ void findmax(float *A, float *max)
{
        // code segments are given in the lecture
}

int main()
{
    float A[N], *A_d, *max_arr, *max_arr_d, max;
    int i, dev;

    dim3 dimBlock(THREADS);
    dim3 dimGrid((N+dimBlock.x-1)/dimBlock.x);

    dev = (getpwuid(getuid())->pw_name[3]-'0')%2? 1: 0;
    cudaSetDevice(dev);

    srand(1);
    for (i=0; i<N; i++) {
        A[i] = rand() % 999;
        printf("%2.1f ", A[i]);
    }
    printf("₩n");

    cudaMalloc((void **) &A_d, sizeof(float)*N);
    cudaMemcpy(A_d, A, sizeof(float)*N, cudaMemcpyHostToDevice);

    cudaMalloc((void **) &max_arr_d, dimGrid.x*sizeof(float));

    findmax<<<dimGrid, dimBlock>>>(A_d, max_arr_d);

    max_arr = (float*)malloc(dimGrid.x*sizeof(float));
    cudaMemcpy(max_arr, max_arr_d, dimGrid.x*sizeof(float), cudaMemcpyDeviceToHost);

    // find the maximum
    max = max_arr[0];
    for (i=1; i<dimGrid.x; i++)
        if (max < max_arr[i]) max = max_arr[i];

    printf("%f₩n", max);

    cudaFree(A_d);
    cudaFree(max_arr_d);
    free(max_arr);
    exit(0);
}
```