Lab 10: CUDA Threads

Name: Waris Damkham ID: 6388014 Sec: 1

Save your file to “lab10_63xxxxxx.pdf” and upload it to MyCourses website.

Q1. Output from “vecAdd4.cu”. Also, pick n and T

n = 1024

T = 240

Output:

```
[u6388014@cluster ~]$ ./vecadd4
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
```

Q2. Source code for filling the array and its output.

Source code:

```
#include <stdio.h>
#define N 256
#define T 4

__global__ void vecAdd(int *A){
  int i = blockIdx.x * blockDim.x + threadIdx.x;

  if(i < N){
    A[i] = i;
  }
}

int main(int argc, char *argv[]){
  int i;
  int blocks = N/T;
  int size = N*sizeof(int);
  int a[N], *devA;

  cudaMalloc( (void**) &devA, size);

  cudaMemcpy( devA, a, size, cudaMemcpyHostToDevice);

  vecAdd<<<blocks, T>>>(devA);

  cudaMemcpy( a, devA, size, cudaMemcpyDeviceToHost);
  cudaFree(devA);

  for(i=0; i<N; i++){
    if(i != 0 && i%20 == 0) printf("\n");
    printf("%d ", a[i]);
  }
  printf("\n");
}
```

Output:

```
[u6388014@cluster ~]$ ./vecfill
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39
40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59
60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79
80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119
120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139
140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159
160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179
180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199
200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219
220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239
240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255
```

Q3. Output from "matmul2.cu"

```
[u6388014@cluster ~]$ ./matmul2
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64
.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00 64.00
```

Q4. Source code and output from “matmul3.cu”

Source code:

```
#include <stdio.h>
#define Width 32

#define TITE_WIDTH 5

__global__ void MatrixMulKernel (float* Md, float* Nd, float* Pd, int ncols) {
        int row = blockIdx.y * blockDim.y + threadIdx.y;
        int col = blockIdx.x * blockDim.x + threadIdx.x;
        float Pvalue = 0;
        if(row < Width || col < Width){
        for(int k=0;k<ncols;k++){
                float Melement = Md[row*ncols+k];
                float Nelement = Nd[k*ncols+col];
                Pvalue += Melement * Nelement;
        }
}

        Pd[row*ncols+col] = Pvalue;
}

int main (int argc, char *argv[]){
        int i,j;
        int size = Width * Width * sizeof(float);
        float M[Width][Width], N[Width][Width], P[Width][Width];
        float* Md, *Nd, *Pd;

        for(i=0;i<Width;i++){
                for(j=0;j<Width;j++){
                        M[i][j] = 1;
                        N[i][j] = 2;
                }
        }
        cudaMalloc( (void**)&Md, size);
        cudaMalloc( (void**)&Nd, size);
        cudaMalloc( (void**)&Pd, size);

        cudaMemcpy( Md, M, size, cudaMemcpyHostToDevice);
        cudaMemcpy( Nd, N, size, cudaMemcpyHostToDevice);

        dim3 dimBlock(TITE_WIDTH, TITE_WIDTH);
        dim3 dimGrid((Width+TITE_WIDTH-1)/TITE_WIDTH,(Width+TITE_WIDTH-1)/TITE_WIDTH);

        MatrixMulKernel<<<dimGrid, dimBlock>>>(Md, Nd, Pd, Width);

        cudaMemcpy(P, Pd, size, cudaMemcpyDeviceToHost);

        cudaFree(Md);
        cudaFree(Nd);
        cudaFree(Pd);
        printf("\n================================\n");
        for(i=0;i<Width;i++){
                for(j=0;j<Width;j++){
                        printf("%.2f ", P[i][j]);
                }printf("\n");
        }
}
```

Output:

```
[u6388014@cluster ~]$ ./matmul3
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00 32.00
```