```
!apt-get update
```

```
Get:1 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:2 https://cloud.r-project.org/bin/linux/ubuntu bionic-cran40/ InRelease [3,626 B]
Ign:3 https://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu1804/x86_64  InRelease
Hit:4 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64  InRelease
Hit:5 https://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu1804/x86_64  Release
Get:6 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic InRelease [15.9 kB]
Hit:7 http://archive.ubuntu.com/ubuntu bionic InRelease
Get:8 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Hit:9 http://ppa.launchpad.net/cran/libgit2/ubuntu bionic InRelease
Hit:10 http://ppa.launchpad.net/deadsnakes/ppa/ubuntu bionic InRelease
Get:11 http://security.ubuntu.com/ubuntu bionic-security/main amd64 Packages [3,127 kB]
Get:12 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [83.3 kB]
Get:13 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic InRelease [21.3 kB]
Get:14 http://security.ubuntu.com/ubuntu bionic-security/restricted amd64 Packages [1,349 kB]
Get:15 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Packages [1,573 kB]
Get:17 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic/main Sources [2,240 kB]
Get:18 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages [3,552 kB]
Get:19 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic/main amd64 Packages [1,145 kB]
Get:20 http://archive.ubuntu.com/ubuntu bionic-updates/restricted amd64 Packages [1,392 kB]
Get:21 http://archive.ubuntu.com/ubuntu bionic-updates/multiverse amd64 Packages [30.8 kB]
Get:22 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Packages [2,348 kB]
Get:23 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic/main amd64 Packages [43.2 kB]
Fetched 17.1 MB in 4s (4,430 kB/s)
Reading package lists... Done
```

```
!wget https://developer.nvidia.com/compute/cuda/9.2/Prod/local_installers/cuda-repo-ubuntu1604-9-2-local_9.2.88-1_amd64 -O cud
!dpkg -i cuda-repo-ubuntu1604-9-2-local_9.2.88-1_amd64.deb
!apt-key add /var/cuda-repo-9-2-local/7fa2af80.pub
!apt-get update
!apt-get install cuda-9.2
```

```
--2023-01-10 13:19:32--  https://developer.nvidia.com/compute/cuda/9.2/Prod/local_installers/cuda-repo-ubuntu1604-9-2-loc
Resolving developer.nvidia.com (developer.nvidia.com)... 152.195.19.142
Connecting to developer.nvidia.com (developer.nvidia.com)|152.195.19.142|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://developer.nvidia.com/downloads/compute/cuda/9.2/Prod/local_installers/cuda-repo-ubuntu1604-9-2-local_9.
--2023-01-10 13:19:32--  https://developer.nvidia.com/downloads/compute/cuda/9.2/Prod/local_installers/cuda-repo-ubuntu16
Reusing existing connection to developer.nvidia.com:443.
HTTP request sent, awaiting response... 302 Found
Location: https://developer.download.nvidia.com/compute/cuda/9.2/secure/Prod/local_installers/cuda-repo-ubuntu1604-9-2-lo
--2023-01-10 13:19:32--  https://developer.download.nvidia.com/compute/cuda/9.2/secure/Prod/local_installers/cuda-repo-ub
Resolving developer.download.nvidia.com (developer.download.nvidia.com)... 152.195.19.142
Connecting to developer.download.nvidia.com (developer.download.nvidia.com)|152.195.19.142|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1267391958 (1.2G) [application/x-deb]
Saving to: ‘cuda-repo-ubuntu1604-9-2-local_9.2.88-1_amd64.deb’

cuda-repo-ubuntu160 100%[===================>]   1.18G   169MB/s    in 6.7s

2023-01-10 13:19:39 (181 MB/s) - ‘cuda-repo-ubuntu1604-9-2-local_9.2.88-1_amd64.deb’ saved [1267391958/1267391958]

Selecting previously unselected package cuda-repo-ubuntu1604-9-2-local.
(Reading database ... 124016 files and directories currently installed.)
Preparing to unpack cuda-repo-ubuntu1604-9-2-local_9.2.88-1_amd64.deb ...
Unpacking cuda-repo-ubuntu1604-9-2-local (9.2.88-1) ...
Setting up cuda-repo-ubuntu1604-9-2-local (9.2.88-1) ...
OK
Get:1 file:/var/cuda-repo-9-2-local  InRelease
Ign:1 file:/var/cuda-repo-9-2-local  InRelease
Get:2 file:/var/cuda-repo-9-2-local  Release [574 B]
Get:2 file:/var/cuda-repo-9-2-local  Release [574 B]
Get:3 file:/var/cuda-repo-9-2-local  Release.gpg [819 B]
Get:3 file:/var/cuda-repo-9-2-local  Release.gpg [819 B]
Hit:4 https://cloud.r-project.org/bin/linux/ubuntu bionic-cran40/ InRelease
Ign:5 https://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu1804/x86_64  InRelease
Hit:6 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64  InRelease
Hit:7 http://archive.ubuntu.com/ubuntu bionic InRelease
Hit:8 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:9 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic InRelease
Hit:10 https://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu1804/x86_64  Release
Get:11 file:/var/cuda-repo-9-2-local  Packages [18.7 kB]
Hit:12 http://archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:13 http://archive.ubuntu.com/ubuntu bionic-backports InRelease
Hit:14 http://ppa.launchpad.net/cran/libgit2/ubuntu bionic InRelease
Hit:15 http://ppa.launchpad.net/deadsnakes/ppa/ubuntu bionic InRelease
Hit:16 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'cuda-9-2' for regex 'cuda-9.2'
Note, selecting 'libcuda-9.2-1' for regex 'cuda-9.2'
The following package was automatically installed and is no longer required:
  libnvidia-common-460
Use 'apt autoremove' to remove it.
```

```
     The following additional packages will be installed:
       cuda-command-line-tools-9-2 cuda-compiler-9-2 cuda-cublas-9-2
       cuda-cublas-dev-9-2 cuda-cudart-9-2 cuda-cudart-dev-9-2 cuda-cufft-9-2
       cuda-cufft-dev-9-2 cuda-cuobjdump-9-2 cuda-cupti-9-2 cuda-curand-9-2
```

```
!nvcc --version
```

```
     nvcc: NVIDIA (R) Cuda compiler driver
     Copyright (c) 2005-2018 NVIDIA Corporation
     Built on Wed_Apr_11_23:16:29_CDT_2018
     Cuda compilation tools, release 9.2, V9.2.88
```

```
!pip install git+https://github.com/andreinechaev/nvcc4jupyter.git
```

```
     Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
     Collecting git+https://github.com/andreinechaev/nvcc4jupyter.git
       Cloning https://github.com/andreinechaev/nvcc4jupyter.git to /tmp/pip-req-build-5y92v3oc
       Running command git clone --filter=blob:none --quiet https://github.com/andreinechaev/nvcc4jupyter.git /tmp/pip-req-bui
       Resolved https://github.com/andreinechaev/nvcc4jupyter.git to commit aac710a35f52bb78ab34d2e52517237941399eff
       Preparing metadata (setup.py) ... done
     Building wheels for collected packages: NVCCPlugin
       Building wheel for NVCCPlugin (setup.py) ... done
       Created wheel for NVCCPlugin: filename=NVCCPlugin-0.0.2-py3-none-any.whl size=4304 sha256=354b497e248d08b83a2aa3cc0238
       Stored in directory: /tmp/pip-ephem-wheel-cache-xguwh1n8/wheels/f3/08/cc/e2b5b0e1c92df07dbb50a6f024a68ce090f5e7b2316b41
     Successfully built NVCCPlugin
     Installing collected packages: NVCCPlugin
     Successfully installed NVCCPlugin-0.0.2
```

```
%load_ext nvcc_plugin
```

```
     created output directory at /content/src
     Out bin /content/result.out
```

%%cu - extension for cuda programming

__ global __ is a functional qualiifier called from the **CPU** and lead to the **GPU**

HelloKernel() user defined function

Hellokernel<<<1, 1>>>(); : The number of cuda threads that execute that kernel for a given kernel call. No. of cuda threads going to executed.

<<<1, 1>>>: <<<block size, thread no>>>

```
%%cu
#include <stdio.h>
__global__ void Hellokernel()
{

}
main()
{
Hellokernel<<<1, 1>>>();
 printf("Hello World\n");
return 0;
}
```

```
     Hello World
```

```
%%cu
#include <stdio.h>
__global__ void add(int a, int b, int *c)
{
*c = a + b;
}
int main(void)
{
int c;
int *dev_c;
cudaMalloc((void**)&dev_c, sizeof(int));
add << <1, 1 >> > (2, 7, dev_c);
cudaMemcpy(&c, dev_c, sizeof(int),
cudaMemcpyDeviceToHost);
printf("2 + 7 = %d\n", c);
cudaFree(dev_c);
return 0;
}
```

```
     2 + 7 = 1
```

```
%%cu
#include <stdio.h>
__global__ void vector_add(int *out_d, int *a, int *b, int n)
    {
    int bx = blockIdx.x;
int by = blockIdx.y;
int tx = threadIdx.x;
int ty = threadIdx.y;
int row = by*blockDim.y + ty;
int col = bx*blockDim.x + tx;
int dim = gridDim.x*blockDim.x;
int i = row*dim + col;
out_d[i] = a[i] + b[i];

     }

int main()
    {
    int *a, *b, *out_d,*out;
     int *d_a, *d_b;
     int N=6;
    int i;
     a = (int*)malloc(sizeof(int) * N);
     b = (int*)malloc(sizeof(int) * N);
     out = (int*)malloc(sizeof(int) * N);
    for (i=0;i<N;i++)
        {
        a[i]=i;
        b[i]=i*2;
        }
    cudaMalloc((void**)&d_a, sizeof(int) * N);
    cudaMalloc((void**)&d_b, sizeof(int) * N);
  cudaMalloc((void**)&out_d, sizeof(int) * N);
    cudaMemcpy(d_a, a, sizeof(int) * N, cudaMemcpyHostToDevice);
    cudaMemcpy(d_b, b, sizeof(int) * N, cudaMemcpyHostToDevice);
    vector_add<<<2,4>>>(out_d, d_a, d_b, N);
    cudaMemcpy(out, out_d, sizeof(int) * N, cudaMemcpyDeviceToHost);
    printf("Success");
    for (i=0;i<N;i++)
        {
        printf("%d\n",out[i]);
        }
    cudaFree(d_a);
    cudaFree(d_b);
    cudaFree(out_d);
    free(a);
    free(b);
    free(out);
  return 0;
    }
```

```
Success1760174240
22085
0
0
0
0
```

```
%%cu
#include <stdio.h>
__global__ void matrixMul( int* Pd, int* Md, int* Nd, int width)
{
int bx = blockIdx.x;
int by = blockIdx.y;
int tx = threadIdx.x;
int ty = threadIdx.y;
int col = by*blockDim.y + ty;
int row = bx*blockDim.x + tx;
int Pvalue=0;
for (int k=0;k<width;++k)
    Pvalue+=Md[row*width+k]*Nd[k*width+col];
Pd[row*width+col]=Pvalue;
}

int main()
    {
    int *M, *N1, *Md, *Nd, *Pd, *P;
const int xb = 3; /* gridDim.x */
const int yb = 3; /* gridDim.y */
const int zb = 1; /* gridDim.z */
const int xt = 3; /* blockDim.x */
const int yt = 3; /* blockDim.y */
const int zt = 1; /* blockDim.z */
     int N,width;
  int i;
    width=9;
   N=width*width;
     M = (int*)malloc(sizeof(int) * N);
     N1 = (int*)malloc(sizeof(int) * N);
     P = (int*)malloc(sizeof(int) * N);
    for (i=0;i<N;i++)
        {
        M[i]=i;
        N1[i]=i*2;
        }
  dim3 dimGrid(xb,yb,zb);
dim3 dimBlock(xt,yt,zt);
    cudaMalloc((void**)&Md, sizeof(int) * N);
    cudaMalloc((void**)&Nd, sizeof(int) * N);
  cudaMalloc((void**)&Pd, sizeof(int) * N);
    cudaMemcpy(Md, M, sizeof(int) * N, cudaMemcpyHostToDevice);
    cudaMemcpy(Nd, N1, sizeof(int) * N, cudaMemcpyHostToDevice);
    matrixMul<<<dimGrid,dimBlock>>>(Pd, Md, Nd, width);
    cudaMemcpy(P, Pd, sizeof(int) * N, cudaMemcpyDeviceToHost);
    printf("Success");
    for (i=0;i<N;i++)
        {
        printf("%d\n",P[i]);
        }
    cudaFree(Md);
    cudaFree(Nd);
    cudaFree(Pd);
    free(M);
    free(N1);
    free(P);
  return 0;
    }
```

```
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
```

```
0
0
0
0
0
0
0
0
0
```

```
%%cu
#include <stdio.h>
__global__ void matrixFill ( int *x )
{
int bx = blockIdx.x;
int by = blockIdx.y;
int tx = threadIdx.x;
int ty = threadIdx.y;
int col = by*blockDim.y + ty;
int row = bx*blockDim.x + tx;
int dim =blockDim.x*gridDim.x;
int i = row*dim + col;
x[i] = i;
}
int main ( int argc, char* argv[] )
{
const int xb = 2; /* gridDim.x */
const int yb = 2; /* gridDim.y */
const int zb = 1; /* gridDim.z */
const int xt = 2; /* blockDim.x */
const int yt = 2; /* blockDim.y */
const int zt = 1; /* blockDim.z */
const int n = xb*yb*zb*xt*yt*zt;
printf("allocating array of length %d...\n",n);
int *xhost = (int*)calloc(n,sizeof(int));
for(int i=0; i<n; i++) xhost[i] = -1.0;
int *xdevice;
size_t sx = n*sizeof(int);
cudaMalloc((void**)&xdevice,sx);
cudaMemcpy(xdevice,xhost,sx,cudaMemcpyHostToDevice);
dim3 dimGrid(xb,yb,zb);
dim3 dimBlock(xt,yt,zt);
matrixFill<<<dimGrid,dimBlock>>>(xdevice);
cudaMemcpy(xhost,xdevice,sx,cudaMemcpyDeviceToHost);
cudaFree(xdevice);
int *p = xhost;
for(int i1=0; i1 < xb; i1++)
for(int i2=0; i2 < yb; i2++)
for(int i3=0; i3 < zb; i3++)
for(int i4=0; i4 < xt; i4++)
for(int i5=0; i5 < yt; i5++)
for(int i6=0; i6 < zt; i6++)
printf("x[%d][%d][%d][%d][%d][%d] = %d\n",i1,i2,i3,i4,i5,i6,*(p++));
return 0;
}
```

```
allocating array of length 16...
x[0][0][0][0][0][0] = -1
x[0][0][0][0][1][0] = -1
x[0][0][0][1][0][0] = -1
x[0][0][0][1][1][0] = -1
x[0][1][0][0][0][0] = -1
x[0][1][0][0][1][0] = -1
x[0][1][0][1][0][0] = -1
x[0][1][0][1][1][0] = -1
x[1][0][0][0][0][0] = -1
x[1][0][0][0][1][0] = -1
x[1][0][0][1][0][0] = -1
x[1][0][0][1][1][0] = -1
x[1][1][0][0][0][0] = -1
x[1][1][0][0][1][0] = -1
x[1][1][0][1][0][0] = -1
x[1][1][0][1][1][0] = -1
```

✓ 0s completed at 18:52 ● ×