```
#include <stdio.h>

int main(void)
{
    // function declarations
    void PrintCUDADeviceProperties(void);

    // code
    PrintCUDADeviceProperties();
}

void PrintCUDADeviceProperties(void)
{
    // code
    printf("CUDA INFORMATION :\n");
    printf
      ("================================================================================
      ====\n");

    cudaError_t ret_cuda_rt;
    int dev_count;
    ret_cuda_rt = cudaGetDeviceCount(&dev_count);
    if (ret_cuda_rt != cudaSuccess)
    {
        printf("CUDA Runtime API Error - cudaGetDeviceCount() Failed Due To %s.
          \n", cudaGetErrorString(ret_cuda_rt));
    }
    else if (dev_count == 0)
    {
        printf("There Is No CUDA Supprted Device On This System.\n");
        return;
    }
    else
    {
        printf("Total Number Of CUDA Supporting GPU Device/Devices On This
          System : %d\n", dev_count);
        for (int i = 0; i<dev_count; i++)
        {
            cudaDeviceProp dev_prop;
            int driverVersion = 0, runtimeVersion = 0;

            ret_cuda_rt = cudaGetDeviceProperties(&dev_prop, i);
            if (ret_cuda_rt != cudaSuccess)
            {
                printf("%s in %s at line %d\n", cudaGetErrorString
                  (ret_cuda_rt), __FILE__, __LINE__);
                return;
            }
            printf("\n");
            cudaDriverGetVersion(&driverVersion);
            cudaRuntimeGetVersion(&runtimeVersion);
            printf("******** CUDA DRIVER AND RUNTIME INFORMATION ********\n");
            printf("=================================================\n");
            printf("CUDA Driver Version                                : %d.%
              d\n", driverVersion / 1000, (driverVersion % 100) / 10);
            printf("CUDA Runtime Version                               : %d.%
```

```
    d\n", runtimeVersion / 1000, (runtimeVersion % 100) / 10);
51          printf("\n");
52          printf("====================================================\n");
53          printf("********** GPU DEVICE GENERAL INFORMATION **********\n");
54          printf("====================================================\n");
55          printf("GPU Device Number                                  : %d
              \n", i);
56          printf("GPU Device Name                                    : %s
              \n", dev_prop.name);
57          printf("GPU Device Compute Capability                      : %d.%
              d\n", dev_prop.major, dev_prop.minor);
58          printf("GPU Device Clock Rate                              : %d
              \n", dev_prop.clockRate);
59          printf("GPU Device Type                                    : ");
60          if (dev_prop.integrated)
61              printf("Integrated ( On-Board )\n");
62          else
63              printf("Discrete ( Card )\n");
64          printf("\n");
65          printf("********** GPU DEVICE MEMORY INFORMATION ***********\n");
66          printf("====================================================\n");
67          printf("GPU Device Total Memory                            : %.0f
               GB = %.0f MB = %llu Bytes\n", ((float)dev_prop.totalGlobalMem /
              1048576.0f) / 1024.0f, (float)dev_prop.totalGlobalMem /
              1048576.0f, (unsigned long long) dev_prop.totalGlobalMem);
68          printf("GPU Device Constant Memory                         : %lu
              Bytes\n", (unsigned long)dev_prop.totalConstMem);
69          printf("GPU Device Shared Memory Per SMProcessor           : %lu
              \n", (unsigned long)dev_prop.sharedMemPerBlock);
70          printf("\n");
71          printf("****** GPU DEVICE MULTIPROCESSOR INFORMATION ********\n");
72          printf("====================================================\n");
73          printf("GPU Device Number Of SMProcessors                  : %d
              \n", dev_prop.multiProcessorCount);
74          printf("GPU Device Number Of Registers Per SMProcessor     : %d
              \n", dev_prop.regsPerBlock);
75          printf("\n");
76          printf("*********** GPU DEVICE THREAD INFORMATION ***********\n");
77          printf("====================================================\n");
78          printf("GPU Device Maximum Number Of Threads Per SMProcessor : %d
              \n", dev_prop.maxThreadsPerMultiProcessor);
79          printf("GPU Device Maximum Number Of Threads Per Block     : %d
              \n", dev_prop.maxThreadsPerBlock);
80          printf("GPU Device Threads In Warp                         : %d
              \n", dev_prop.warpSize);
81          printf("GPU Device Maximum Thread Dimensions               : ( %
              d, %d, %d )\n", dev_prop.maxThreadsDim[0], dev_prop.maxThreadsDim
              [1], dev_prop.maxThreadsDim[2]);
82          printf("GPU Device Maximum Grid Dimensions                 : ( %
              d, %d, %d )\n", dev_prop.maxGridSize[0], dev_prop.maxGridSize[1],
               dev_prop.maxGridSize[2]);
83          printf("\n");
84          printf("*********** GPU DEVICE DRIVER INFORMATION ***********\n");
85          printf("====================================================\n");
86          printf("GPU Device has ECC support                         : %s
              \n", dev_prop.ECCEnabled ? "Enabled" : "Disabled");
```

```
87  #if defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
88              printf("GPU Device CUDA Driver Mode ( TCC Or WDDM )          : %s
                  \n", dev_prop.tccDriver ? "TCC ( Tesla Compute Cluster
                  Driver )" : "WDDM ( Windows Display Driver Model )");
89  #endif
90              printf
                  ("*************************************************************
                  ************\n");
91          }
92      }
93  }
94
```