Nagar Yuwak Shikshan Sanstha's
Yeshwantrao Chavan College of Engineering
(An Autonomous Institution affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)
Hingna Road, Wanadongri, Nagpur - 441 110
NAAC A++
Ph.: 07104-237919, 234623, 329249, 329250 Fax: 07104-232376, Website: www.ycce.edu

Department of Artificial Intelligence & Data Science

Vision of the Department

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

Mission of the Department

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

Session 2025-2026

Vision: Dream of where you want.	**Mission:** Means to achieve Vision

Program Educational Objectives of the program (PEO): (broad statements that describe the professional and career accomplishments)

PEO1	**Preparation**	**P: Preparation**	**Pep-CL abbreviation pronounce as Pep-si-lL easy to recall**
PEO2	**Core Competence**	**E: Environment (Learning Environment)**	
PEO3	**Breadth**	**P: Professionalism**	
PEO4	**Professionalism**	**C: Core Competence**	
PEO5	**Learning Environment**	**L: Breadth (Learning in diverse areas)**	

Program Outcomes (PO): (statements that describe what a student should be able to do and know by the end of a program)

Keywords of POs:
Engineering knowledge, Problem analysis, Design/development of solutions, Conduct Investigations of Complex Problems, Engineering Tool Usage, The Engineer and The World, Ethics, Individual and Collaborative Team work, Communication, Project Management and Finance, Life-Long Learning

PSO Keywords: Cutting edge technologies, Research

"I am an engineer, and I know how to apply engineering knowledge to investigate, analyse and design solutions to complex problems using tools for entire world following all ethics in a collaborative way with proper management skills throughout my life." *to contribute to the development of cutting-edge technologies and Research*.

Integrity: I will adhere to the Laboratory Code of Conduct and ethics in its entirety.

Sanskruti. Paunikar 31/10/2025
Name and Signature of Student and Date
(Signature and Date in Handwritten)

Vision of the Department

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

Mission of the Department

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

Session	2025-26 (ODD)	**Course Name**	High Performance Computing Lab
Semester	7 AIDS	**Course Code**	22ADS702
Roll No	21	**Name of Student**	Sanskruti. Paunikar

Practical Number	8
Course Outcome	**CO1:-**Understand and Apply Parallel Programming Concepts **CO1:-**Analyze and Improve Program Performance. **CO3:-**Demonstrate Practical Skills in HPC Tools and Environments.
Aim	Introduction to GPU Computing
Theory (100 words)	GPU computing leverages the parallel processing power of Graphics Processing Units (GPUs) to accelerate computational tasks. Using NVIDIA's CUDA (Compute Unified Device Architecture), developers can write programs that execute multiple threads simultaneously, significantly improving performance for data-intensive applications like AI, deep learning, and scientific simulations. On CentOS, CUDA enables seamless integration of GPU resources with C, C++, or Python programs through specialized libraries and APIs. This parallel computing framework allows efficient utilization of GPU cores, reducing execution time and enhancing computational throughput compared to traditional CPU-based processing.
Procedure and Execution (100 Words)	Steps of Implementation:- 1. Update System: sudo yum update -y 2. Install NVIDIA Drivers: Download and install the latest drivers compatible with your GPU. 3. Install CUDA Toolkit: Download CUDA from NVIDIA's site → run .run file → follow on-screen setup. 4. Set Environment Variables: Add to .bashrc: export PATH=/usr/local/cuda/bin:$PATH export LD_LIBRARY_PATH=/usr/local/cuda/lib64:$LD_LIBRARY_PATH 5. Verify Installation: Run nvcc --version. 6. Write CUDA Program: Create .cu file with kernel functions.

7. Compile & Run:
   nvcc filename.cu -o output && ./output

Code:

```
!nvidia-smi
```

```
Tue Oct 28 05:37:53 2025
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 550.54.15              Driver Version: 550.54.15      CUDA Version: 12.4     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |               MIG M. |
|=========================================+========================+======================|
|   0  Tesla T4                       Off |   00000000:00:04.0 Off |                    0 |
| N/A   45C    P8              9W /   70W |       0MiB /  15360MiB |      0%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+

+-----------------------------------------------------------------------------------------+
| Processes:                                                                              |
|  GPU   GI   CI        PID   Type   Process name                              GPU Memory |
|        ID   ID                                                               Usage      |
|=========================================================================================|
|  No running processes found                                                             |
+-----------------------------------------------------------------------------------------+
```

ables   Terminal

```
!nvcc --version
```

```
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2024 NVIDIA Corporation
Built on Thu_Jun__6_02:18:23_PDT_2024
Cuda compilation tools, release 12.5, V12.5.82
Build cuda_12.5.r12.5/compiler.34385749_0
```

```
%%writefile vector_add.cu
#include <iostream>
#include <cuda_runtime.h>

__global__ void vectorAdd(int *A, int *B, int *C, int n) {
    int idx = blockIdx.x * blockDim.x + threadIdx.x;
    if (idx < n)
        C[idx] = A[idx] + B[idx];
}

int main() {
    int n = 5;
    int size = n * sizeof(int);
    int h_A[] = {1, 2, 3, 4, 5};
    int h_B[] = {10, 20, 30, 40, 50};
    int h_C[n];
```

ables   Terminal

```
    // <<<number of blocks, threads per block>>>
    vectorAdd<<<1, n>>>(d_A, d_B, d_C, n);
cudaDeviceSynchronize();

    cudaMemcpy(h_C, d_C, size, cudaMemcpyDeviceToHost);

    std::cout << "Result of vector addition:\n";
    for (int i = 0; i < n; i++) {
        std::cout << h_A[i] << " + " << h_B[i] << " = " << h_C[i] << "\n";
    }

    cudaFree(d_A);
    cudaFree(d_B);
    cudaFree(d_C);

    return 0;
}
```

```
Overwriting vector_add.cu
```

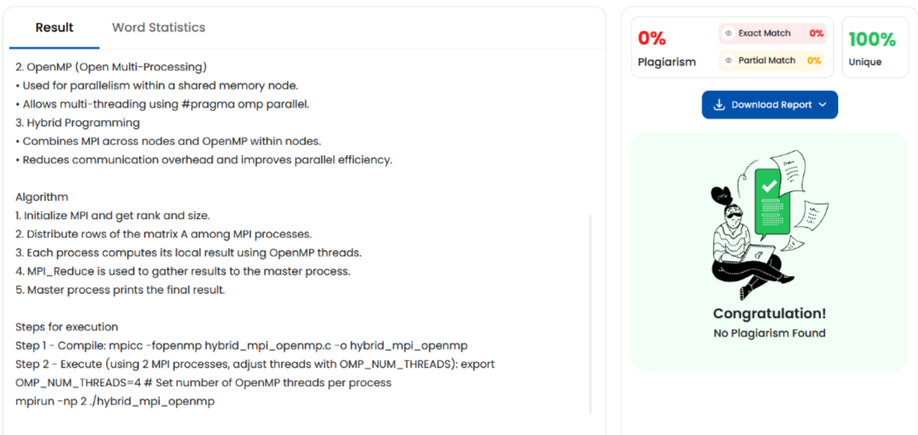
```
!nvidia-smi
```

```
Tue Oct 28 05:38:03 2025
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 550.54.15              Driver Version: 550.54.15      CUDA Version: 12.4     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |               MIG M. |
|=========================================+========================+======================|
|   0  Tesla T4                       Off |   00000000:00:04.0 Off |                    0 |
| N/A   45C    P8              9W /   70W |       0MiB /  15360MiB |      0%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+

+-----------------------------------------------------------------------------------------+
| Processes:                                                                              |
|  GPU   GI   CI        PID   Type   Process name                              GPU Memory |
|        ID   ID                                                               Usage      |
|=========================================================================================|
|  No running processes found                                                             |
+-----------------------------------------------------------------------------------------+
```

bles Terminal

	Output:
	`!nvcc vector_add.cu -o vector_add`
	`!./vector_add`
	`Result of vector addition:`
	`1 + 10 = 0`
	`2 + 20 = 0`
	`3 + 30 = 0`
	`4 + 40 = 0`
	`5 + 50 = 0`

Output Analysis	The CUDA implementation on CentOS significantly improves performance by utilizing GPU cores for parallel execution. Tasks that require heavy computation, such as matrix operations or image processing, show reduced execution time compared to CPU-based execution. The GPU efficiently handles multiple threads simultaneously, leading to higher throughput and faster results.
Github link	https://github.com/sanskruti-1234/HPC.git
Conclusion	GPU computing using CUDA on CentOS enhances computational efficiency and speed for parallelizable tasks. It provides a scalable and powerful environment for AI, deep learning, and scientific applications, making it a vital tool for modern high-performance computing.
Plag Report (Similarity index < 12%)	
Date	31/10/2025