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.

**Name and Signature of Student and Date**

(Signature and Date in Handwritten)

Department of Computer Technology

**Vision the Department**

computer education innovative pedagogy, value-based

# Mission 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** | **HPC Lab** |
| **Semester** | **7** | **Course Code** | **22ADS706** |
| **Roll No** | **24** | **Name of Student** | **Shreya kudmethi** |

|  |  |
| --- | --- |
| Practical Number | **3** |
| Course Outcome | 1. Understand and Apply Parallel Programming Concepts 2. Analyze and Improve Program Performance.  3. Demonstrate Practical Skills in HPC Tools and Environments |
| Aim | **Introduction to OpenMP** |
| Problem Definition | 1. Introduction/Theory 2. Execution Steps (for a Linux environment like CentOS) 3. A Sample OpenMP Program (with code written for GCC on CentOS) |
| Theory  (100 words) | OpenMP stands for Open Multi-Processing. It is an API  (Application Programming Interface)  that supports multi-platform shared-memory multiprocessing programming in C, C++, and Fortran.  It allows developers to write parallel code easily using compiler directives, library routines, and environment variables.  Key Features:  Supports shared memory multiprocessing  Uses fork-join model: master thread forks a specified number of slave threads  Simple and readable syntax using pragma directives    Controlled by environment variables (e.g.,  OMP\_NUM\_THREADS)  Scales well for multi-core CPUs |
| Procedure and Execution  (100 Words) | Algorithm:    Execution Steps on CentOS/Linux  Follow these steps to compile and run an OpenMP program on a CentOS Linux system.  Step 1: Install GCC with OpenMP support  Most CentOS systems have GCC preinstalled. If not:  sudo yum install gcc |

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

|  |  |
| --- | --- |
|  | To verify OpenMP support:  gcc -fopenmp --version  Step 2: Write the OpenMP Program Create a file named openmp\_example.c. nano openmp\_example.c  Paste your OpenMP C code (example below).  Step 3: Compile the Program  Use -fopenmp flag to enable OpenMP: gcc -fopenmp -o openmp\_example openmp\_example.c  Step 4: Set Number of Threads (Optional)  You can set how many threads OpenMP should use: export OMP\_NUM\_THREADS=4  Step 5: Run the Program  ./openmp\_example |
| Code:  Example 1: OpenMP Scheduling Clause  #include &lt;stdio.h&gt; #include &lt;omp.h&gt;  int main() {  int i; int n = 12;  #pragma omp parallel for schedule(static, 3)  for (i = 0; i &lt; n; i++) { printf(&quot;Thread %d processing iteration %d\n&quot;, omp\_get\_thread\_num(), i);  }  return 0;  }    Example 2: Using Barrier for Synchronization  #include &lt;stdio.h&gt; #include &lt;omp.h&gt;  int main() {  #pragma omp parallel num\_threads(4)  {  int id = omp\_get\_thread\_num();    printf(&quot;Thread %d before barrier\n&quot;, id);  #pragma omp barrier  if (id == 0) { |

Department of Computer Technology

# 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**.**

|  |  |
| --- | --- |
|  | printf(&quot;All threads reached the barrier. Thread %d continuing.\n&quot;, id);  } }  return 0;  }    Example 3: Nested Parallelism  #include &lt;stdio.h&gt; #include &lt;omp.h&gt;  int main() { omp\_set\_nested(1); // Enable nested parallelism  #pragma omp parallel num\_threads(2)  {  int id = omp\_get\_thread\_num(); printf(&quot;Outer thread %d starting\n&quot;, id);  #pragma omp parallel num\_threads(2)  {  int inner\_id = omp\_get\_thread\_num(); printf(&quot; Inner thread %d of outer thread %d\n&quot;, inner\_id, id);  } }  return 0;  } |

# 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**.**

|  |  |
| --- | --- |
|  | Output: |

# 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**.**

|  |  |
| --- | --- |
|  |  |
| Output Analysis | The outputs confirm correct parallel execution:    In Example 1, threads were assigned loop iterations in chunks (static scheduling).    In Example 2, all threads waited at a barrier, ensuring synchronization.    Nested parallelism showed multiple threads spawning further threads, confirming OpenMP's ability to handle complex thread hierarchies.  Each output reflected correct thread IDs and sequence (though order may vary due to concurrency), validating OpenMP directives' behavior. |

# 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**.**

|  |  |
| --- | --- |
| Link of student Github profile where lab assignment has been uploaded |  |
| Conclusion | The practical successfully demonstrated the use of OpenMPractica in C for parallel programming. Basic directives like parallel, for, barrier, schedule, and nested were implemented and tested. The programs compiled and executed correctly on CentOS with GCC, proving that OpenMP offers a simple yet powerful interface for exploiting parallelism in shared-memory architectures. This lays the foundation for developing efficient multi-threaded applications. |
| Plag Report  (Similarity index <  12%) |  |
| Date | **02/09/25** |