Semester: VII	Department of Data Science	AY: 2026-27
Course Code:BDS701	Course Name: Parallel Programming	

Question Bank

Q.	Module-01: Introduction to parallel programming, parallel hardware and parallel software	M	L	CO
1.	Define Parallel Programming and explain the need of parallel programming in modern computer systems.	8	L2	CO1
2.	Explain the classification of parallel computers according to Flynn's taxonomy.	8	L2	CO1
3.	Describe a SIMD system with a neat diagram and explain data parallelism with a suitable example.	8	L2	CO1
4.	Explain MIMD systems and discuss their characteristics with suitable example MIMD systems.	8	L2	CO2
5.	Explain shared-memory and distributed-memory architectures with a neat diagram.	8	L2	CO2
6.	Explain the interconnection networks used in parallel computers and discuss their impact on system performance.	12	L2	CO2
7.	Explain cache coherence and discuss the cache coherence problem in a shared-memory system.	8	L2	CO2
8.	Compare shared-memory and distributed-memory systems with respect to communication, scalability and programming complexity.	12	L4	CO2
9.	Explain in detail the classification of parallel computers according to Flynn's taxonomy. Compare SIMD and MIMD.	8	L4	CO1
10.	Discuss shared-memory and distributed-memory architectures. Explain their working principle, advantages and disadvantages.	12	L2	CO2
11.	Explain cache coherence. Explain snooping-based and directory-based coherence mechanisms with a suitable example.	12	L2	CO2
12.	Explain non-determinism and race conditions in shared-memory programs. How can they be avoided?	8	L3	CO2
13.	Define parallel computing and differentiate it from concurrent computing.	8	L1	CO1
14.	Classify parallel computers based on Flynn's taxonomy.	8	L2	CO1
15.	Differentiate between SISD and MIMD systems.	8	L4	CO1
16.	Explain the interconnection networks used in parallel computers.	8	L2	CO2

Aditya College of Engineering and Technology

Q.	Module-01: Introduction to parallel programming, parallel hardware and parallel software	M	L	CO
17.	What is cache coherence? List out its importance.	8	L1	CO2
18.	Compare shared-memory and distributed-memory systems with a diagram.	8	L4	CO2
19.	Explain coordinating and synchronization of processes/threads in parallel computing.	8	L2	CO2
20.	Explain the shared-memory programming model.	8	L2	CO2
21.	Explain the distributed-memory programming model.	8	L2	CO2
22.	List out the challenges in maintaining cache coherence in a shared-memory system.	8	L2	CO2
23.	Explain the Single Instruction Multiple Data (SIMD) system. Why are vector processors widely regarded as SIMD systems? Highlight the characteristics of a vector processor.	8	L2	CO1
24.	What are Multiple Instruction Multiple Data (MIMD) systems? Differentiate between distributed-memory and shared-memory systems.	8	L2	CO2
25.	Discuss shared-memory interconnection and distributed-memory interconnection with suitable figures.	8	L2	CO2
26.	Explain shared-memory systems with neat diagrams of UMA and NUMA multicore systems.	10	L2	CO2
27.	Discuss direct interconnection and indirect interconnection with suitable examples.	8	L2	CO2
28.	Define latency and bandwidth in interconnection networks. How do they affect message transmission time?	8	L2	CO2

Q.	Module-02: GPU programming, programming hybrid systems, MIMD systems, GPUs, performance	M	L	CO
29.	Explain the architecture and programming model of a GPU.	10	L2	CO2
30.	Explain GPU programming and describe heterogeneous programming with suitable explanation.	8	L2	CO2
31.	Explain the organization of threads in GPU systems and discuss SIMD execution within thread groups.	8	L2	CO2
32.	Explain programming of hybrid systems and discuss the advantages and challenges of using multiple parallel programming APIs.	8	L2	CO2
33.	Define speedup and efficiency in MIMD systems and explain their significance in performance evaluation.	8	L2	CO2

Aditya College of Engineering and Technology

Q.	Module-02: GPU programming, programming hybrid systems, MIMD systems, GPUs, performance	M	L	CO
34.	Explain Amdahl's law with a suitable example and discuss its implications on parallel performance.	8	L2	CO2
35.	Explain scalability in MIMD systems and distinguish between strong scalability and weak scalability.	8	L2	CO2
36.	Explain the methods used for taking timings of MIMD programs and justify the use of wall clock time.	8	L2	CO2
37.	Explain GPU performance evaluation and discuss why traditional efficiency and scalability measures are not applicable to GPU programs.	8	L2	CO2
38.	What is a hybrid (GPU+CPU) programming system? Explain how it combines the advantages of MIMD and SIMD systems.	10	L2	CO2
39.	Describe the key features of MIMD systems.	6	L1	CO2
40.	Compare GPU and CPU in terms of architecture and performance.	8	L4	CO2
41.	Define speedup and efficiency in parallel computing.	6	L1	CO2
42.	State and explain Amdahl's law.	10	L2	CO2
43.	Write the formulae for speedup and efficiency with an example.	6	L3	CO2
44.	Explain the concept of scalability in MIMD systems.	6	L2	CO2
45.	How is the timing of an MIMD program measured?	6	L1	CO2
46.	List and explain the factors affecting GPU performance.	10	L2	CO2
47.	Discuss the memory hierarchy and memory performance in a GPU-based system.	8	L2	CO2
48.	List out the limitations of Amdahl's law.	6	L1	CO2
49.	Explain strong and weak scalability in parallelism.	8	L2	CO2
50.	Why is wall clock time preferred over CPU time when measuring the performance of a parallel program?	6	L2	CO2
51.	Discuss the impact of synchronization overhead on the performance of an MIMD program. Suggest techniques to reduce it.	6	L4	CO2
52.	What is Amdahl's law? Derive the expressions for speedup and efficiency in an MIMD system and discuss their implications.	10	L3	CO2
53.	Explain how speedup and efficiency vary as the problem size and number of processes change.	6	L4	CO2

Aditya College of Engineering and Technology

Q.	Module-03: Distributed memory programming with MPI	M	L	CO
54.	Define MPI. Explain the structure of an MPI program with an example.	8	L2	CO3
55.	Explain the trapezoidal rule in MPI.	8	L2	CO3
56.	Differentiate between MPI_Reduce, MPI_Allreduce and MPI_Bcast.	8	L4	CO3
57.	Define collective communication. Explain tree-structured communication for global sum and an alternative tree structure for global sum.	8	L2	CO3
58.	Explain the distributed-memory architecture and describe the message-passing programming model with a neat diagram.	8	L2	CO3
59.	Explain the structure of an MPI program and describe the role of MPI_Init, MPI_Finalize, MPI_Comm_size and MPI_Comm_rank.	8	L2	CO3
60.	Explain point-to-point communication in MPI and describe the working of the MPI_Send and MPI_Recv functions.	10	L2	CO3
61.	Explain the trapezoidal rule and describe how it is parallelized using MPI with a neat diagram.	10	L2	CO3
62.	Explain the issues involved in dealing with input and output in MPI programs.	8	L2	CO3
63.	Explain collective communication in MPI and describe the working of the MPI_Reduce and MPI_Allreduce functions.	8	L2	CO3
64.	Explain MPI_Scatter and MPI_Gather with syntax and an example. How do they differ from MPI_Bcast?	8	L2	CO3
65.	Explain MPI-derived datatypes. Why are they needed? Describe MPI_Type_create_struct with an example.	10	L2	CO3
66.	Explain MPI_Pack and MPI_Unpack as alternatives to derived datatypes.	6	L2	CO3
67.	Explain how the performance of an MPI program is evaluated. Describe the use of MPI_Wtime and MPI_Barrier for taking timings.	8	L2	CO3
68.	Describe the parallel odd-even transposition sort algorithm and explain how it is implemented in MPI.	12	L3	CO3
69.	Explain deadlock in point-to-point communication with an example. How can it be avoided using MPI_Sendrecv or by altering the call sequence?	8	L3	CO3
70.	Write an MPI program to demonstrate MPI_Send and MPI_Recv.	8	L3	CO3
71.	Write an MPI program to demonstrate the broadcast operation.	8	L3	CO3
72.	Write an MPI program to find the sum of an array using MPI_Scatter and MPI_Reduce.	10	L3	CO3

Aditya College of Engineering and Technology

Q.	Module-03: Distributed memory programming with MPI	M	L	CO
73.	Write an MPI program to demonstrate MPI_Reduce and MPI_Allreduce using MPI_MAX, MPI_MIN, MPI_SUM and MPI_PROD.	10	L3	CO3
74.	Discuss the semantics of MPI_Send and MPI_Recv (blocking vs buffering, message matching, status and tags).	8	L2	CO3
75.	Analyze the speedup and efficiency of a parallel trapezoidal rule MPI program when the number of processes is increased.	8	L4	CO3
76.	Explain a parallel sorting algorithm using MPI and discuss its scalability.	8	L4	CO3

Q.	Module-04: Shared-memory programming with OpenMP	M	L	CO
77.	Explain OpenMP pragmas and directives. Describe the structure of a basic OpenMP program with an example.	8	L2	CO4
78.	Explain the parallel directive and the clauses num_threads and private with an example.	8	L2	CO4
79.	Explain the trapezoidal rule and describe how it is parallelized using OpenMP.	10	L2	CO4
80.	Explain the scope of variables in OpenMP. Differentiate between shared, private, firstprivate and lastprivate variables.	8	L4	CO4
81.	Explain the reduction clause in OpenMP with an example. Why is it needed?	8	L2	CO4
82.	Explain the parallel for directive. What restrictions apply to loops that can be parallelized?	8	L2	CO4
83.	What is loop-carried dependency? Explain how to identify and remove it with an example.	10	L3	CO4
84.	Explain loop scheduling in OpenMP. Differentiate between static, dynamic, guided, auto and runtime schedules.	10	L4	CO4
85.	Explain the schedule(static, chunksize) clause with an example showing how iterations are assigned to threads.	8	L3	CO4
86.	Explain the producer-consumer problem in OpenMP. How are queues used for message passing between threads?	10	L2	CO4
87.	Explain critical sections, atomic directive and locks in OpenMP. Compare their usage.	10	L4	CO4
88.	Explain the barrier directive and the nowait clause in OpenMP.	6	L2	CO4
89.	Explain caches, cache coherence and false sharing in OpenMP. How can false sharing be avoided?	10	L2	CO4

Aditya College of Engineering and Technology

Q.	Module-04: Shared-memory programming with OpenMP	M	L	CO
90.	Explain tasking in OpenMP. Describe the task and taskwait directives with a Fibonacci example.	10	L3	CO4
91.	Explain thread safety. Discuss thread-safe and non-thread-safe functions with an example such as strtok.	8	L2	CO4
92.	Write an OpenMP program to sort an array of n elements using both sequential and parallel merge sort (using sections). Record the difference in execution time.	10	L3	CO4
93.	Write an OpenMP program that divides iterations into chunks of 2 (OMP_SCHEDULE=static,2) and prints which iterations are executed by which thread.	8	L3	CO4
94.	Write an OpenMP program to calculate n Fibonacci numbers using tasks.	8	L3	CO4
95.	Write an OpenMP program to find the prime numbers from 1 to n using the parallel for directive. Record both serial and parallel execution times.	8	L3	CO4
96.	Differentiate between the sections directive and the parallel for directive with examples.	6	L4	CO4
97.	Explain how the performance of an OpenMP program can be measured using omp_get_wtime.	6	L2	CO4
98.	Discuss the OpenMP execution model (fork-join) with a neat diagram.	6	L2	CO4
99.	Analyze why a parallel version of a program using critical sections may run slower than the serial version. Suggest improvements.	8	L4	CO4
100.	List the common OpenMP runtime library functions and environment variables with their purpose.	6	L1	CO4

Q.	Module-05: GPU programming with CUDA	M	L	CO
101.	Explain GPUs and GPGPU. Why are GPUs suitable for general-purpose parallel computing?	8	L2	CO5
102.	Explain the architecture of a GPU with a neat diagram.	10	L2	CO5
103.	Explain heterogeneous computing. Describe the roles of the host and the device in a CUDA program.	8	L2	CO5
104.	Explain threads, blocks and grids in CUDA with a neat diagram.	10	L2	CO5
105.	Explain the built-in variables threadIdx, blockIdx, blockDim and gridDim and how a global thread index is computed.	8	L3	CO5
106.	Explain Nvidia compute capabilities and device architectures.	8	L2	CO5

Aditya College of Engineering and Technology

Q.	Module-05: GPU programming with CUDA	M	L	CO
107.	Write a CUDA program for vector addition. Explain memory allocation, data transfer and kernel launch.	12	L3	CO5
108.	Explain how results are returned from CUDA kernels. Describe cudaMalloc, cudaMemcpy and cudaFree.	8	L2	CO5
109.	Explain the use of the __global__, __device__ and __host__ qualifiers in CUDA.	6	L2	CO5
110.	Explain the CUDA trapezoidal rule I with a suitable kernel.	10	L3	CO5
111.	Explain the CUDA trapezoidal rule II: improving performance. How is shared memory used to speed up the reduction?	10	L3	CO5
112.	Explain the CUDA trapezoidal rule III: blocks with more than one warp.	10	L3	CO5
113.	What is a warp? Explain warp scheduling and warp divergence in CUDA.	8	L2	CO5
114.	Explain the CUDA memory hierarchy (global, shared, constant, register and local memory).	8	L2	CO5
115.	Differentiate between CPU and GPU architectures with respect to cores, threads and memory.	8	L4	CO5
116.	Explain the kernel launch configuration <<<blocks, threads>>> and discuss how to choose block size.	8	L3	CO5
117.	Explain thread synchronization in CUDA using __syncthreads() with an example.	8	L2	CO5
118.	Design a CUDA program to perform matrix addition using a 2D grid and thread blocks.	10	L3	CO5
119.	Design a CUDA program to compute the dot product of two vectors using shared memory reduction.	12	L3	CO5
120.	Discuss the factors that affect the performance of a CUDA program and suggest optimizations.	8	L4	CO5
121.	Compare CUDA and OpenMP programming models.	8	L4	CO5
122.	Explain how execution time of a CUDA kernel is measured using CUDA events.	6	L2	CO5
123.	Explain error handling in CUDA programs.	6	L2	CO5
124.	Analyze the effect of memory coalescing on GPU performance.	8	L4	CO5

Aditya College of Engineering and Technology

***Note:**

1. All modules should contain a minimum of **10** questions each and the total number of questions must be **50**.
2. Questions can carry marks of 06, 07, 08, 09, 10 and for problems: 04 or 05.
3. Follow the RBTL as per the keywords used to frame the questions.
4. Maintain the Course Outcomes (COs) as per the syllabus.
5. M: Marks, L: RBTL, C:COs.

Aditya College of Engineering and Technology