School of Computer Science

M. Sc. (Information Technology) Programme

Subject Code & Name: MITMC203- Practical based on MITMC201 and MITMC202

Teaching And Evaluation Scheme:

Teaching Scheme				Examination Scheme			
Credits				Component Weightage			
				CCE		SEE	
L	T	P	Total	TH	PWE	TH	PWE
0	0	4	4	0	50%	0	50%

Programme Name	MSc. (IT)
Semester	2
Course Code	MITMC203
Course Title	Practical based on MITMC201 and MITMC202
Course Content Type (Th. / Pr.)	Pr.
Course Credit	4
Sessions + Lab Per Week	8
Total Teaching/ Lab Hours	120
*** 2 Lab = 1 session**	

Learning Objectives
Upon successful completion of this course, students will be able to:

- Understand the fundamentals of Java programming, Java environment, and object-oriented programming concepts.
- Apply Java programming constructs such as operators, arrays, control statements, classes, and methods to develop programs.
- Analyze object-oriented features including inheritance, abstraction, interfaces, exception handling, and multithreading in Java applications.
- Develop database-driven and modular Java applications using Java Collections Framework and JDBC connectivity.
- Understand the fundamental concepts, history, evolution, and applications of Artificial Intelligence.
- Differentiate among Artificial Intelligence, Machine Learning, Deep Learning, and Generative AI technologies.
- Understand the architecture, capabilities, and limitations of Large Language Models (LLMs) and Generative AI systems.
- Apply prompt engineering techniques and frameworks to interact effectively with AI systems.
- Analyze AI-generated outputs and optimize prompts for improved accuracy and performance.
- Evaluate ethical, legal, security, and societal implications of AI technologies.
- Design AI-assisted workflows, intelligent applications, and domain-specific prompt solutions.
- Develop practical skills in using Generative AI tools for research, software development, data analysis, and content creation.

Learning Outcomes

After successful completion of the programming course, the student will be able to:

- Explain Java architecture, execution process, and core programming concepts
- Demonstrate the use of control structures, arrays, strings, and object-oriented programming techniques in Java
- Implement exception handling, multithreading, interfaces, and collection framework concepts in application development

- Design and develop Java applications with database connectivity using JDBC for real-world problem solving.
- Explain the concepts, evolution, classifications, technologies, and applications of Artificial Intelligence and its role in modern computing systems.
- Utilize Generative AI tools and Large Language Models to perform knowledge discovery, content generation, research assistance, and problem-solving tasks.
- Construct effective prompts using standard prompt engineering techniques and frameworks for obtaining desired outputs from AI systems.
- Analyse AI-generated responses and evaluate prompt effectiveness by identifying limitations, biases, hallucinations, and optimization opportunities.
- Assess ethical, legal, privacy, security, and governance challenges associated with the deployment and use of Artificial Intelligence technologies.
- Design and develop AI-assisted workflows, domain-specific prompt libraries, and innovative solutions using advanced prompt engineering and Generative AI technologies.

CLO – PLO Mapping Matrix									
CLO/ PLO	**PLO1**	**PLO2**	**PLO3**	**PLO4**	**PLO5**	**PLO6**	**PLO7**	**PLO8**	**PLO9**
CLO1	3	2	2	2	1	1	–	1	2
CLO2	3	3	3	2	1	1	–	1	3
CLO3	3	3	3	3	2	2	1	2	3
CLO4	2	3	3	3	2	2	1	2	3
CLO5	3	2	2	3	2	2	1	2	3
CLO6	3	3	2	3	3	2	2	2	3
CLO7	2	3	2	3	3	2	2	2	3
CLO8	2	3	2	3	3	3	2	2	3
CLO9	2	2	2	2	2	3	2	2	2
CLO10	3	3	3	3	3	3	2	3	3

Contents	
Topic/Sub Topic	**Practical Hours**
• Programming exercise based on MITMC201 OOP Using Java • Programming exercise based on MITMC202 Introduction to AI and Prompt Engineering	120

L:: Lecture, **T::** Tutorial, **P::** Practical

CCE:: Continuous & Comprehensive Evaluation

(CCE theory includes Mid Semester Examination, Assignment, MCQ Quizzes, Seminar, Reflective Notes, Class Participation, Case Analysis and Presentation, Slip Tests (Announced/ Surpirsed), Attendance etc. or any combination of these)

PWE: Practical Work Examination

(PWE includes Laboratory Practical Work, Project Work, Viva simulation exercise work etc.)

SEE:: Semester End Evaluation

Sample Programming exercise based on MITMC201 OOP Using Java

1. Write a Java program to demonstrate the basic structure of a Java program and display a welcome message.
2. Write a Java program to perform arithmetic operations using different data types and operators.
3. Write a Java program to find the greater among two and three numbers using conditional statements.
4. Write a Java program to check whether a given number is even or odd.
5. Write a Java program to check whether a given year is a leap year or not.
6. Write a Java program to generate the multiplication table and factorial of a given number using looping statements.
7. Write a Java program to generate the Fibonacci series up to a specified number of terms.

8. Write a Java program to check whether a given number is prime or not.
9. Write a Java program to reverse the digits of a given number and check whether it is a palindrome number.
10. Write a Java program to check whether a given string is a palindrome string.
11. Write a Java program to generate random numbers and perform basic statistical calculations on them.
12. Write a Java program to calculate the sum of digits and determine whether a given number is an Armstrong number.
13. Write a Java program to find the largest and smallest elements in an array.
14. Write a Java program to sort an array in ascending order using Bubble Sort.
15. Write a Java program to perform matrix addition using two-dimensional arrays.
16. Write a Java program to perform string operations such as concatenation, comparison, reversal, and length calculation.
17. Write a menu-driven Java program using switch-case statements to perform calculator operations.
18. Create a class to store and display student information using data members and member methods.
19. Create a class to calculate the area and perimeter of geometric shapes using methods and objects.
20. Create a class to manage employee details and display employee records using member functions.
21. Create a class with default and parameterized constructors to initialize student information.
22. Create a class demonstrating constructor overloading for different ways of initializing an object.
23. Create a class to demonstrate the use of public, private, protected, and static members.
24. Create a class named BankAccount to perform deposit, withdrawal, and balance inquiry operations.
25. Create a class to demonstrate object passing as a method parameter for performing addition of two complex numbers.

26. Create a base class Person and derive a class Student to demonstrate single inheritance.
27. Create classes Student, Examination, and Result to demonstrate multilevel inheritance.
28. Create a hierarchical inheritance model for different types of vehicles and display their properties.
29. Create a class hierarchy for employees and demonstrate method overriding and the use of the super keyword.
30. Create an abstract class Shape and implement derived classes to calculate the area of various geometric figures.
31. Create an interface containing arithmetic operation methods and implement it in a concrete class.
32. Create multiple interfaces and implement them in a single class to demonstrate multiple inheritance through interfaces.
33. Write a Java program to handle divide-by-zero and array index exceptions using try-catch blocks.
34. Write a Java program to create and throw a user-defined exception and demonstrate the use of finally.
35. Create a thread by extending the Thread class and implementing the Runnable interface.
36. Write a Java program to demonstrate thread synchronization using a shared resource.
37. Write a Java program to demonstrate HashSet, TreeSet, and HashMap collections with sorting and searching operations.
38. Create a user-defined package and demonstrate package access protection.
39. Develop a database-driven Student Management System using JDBC, Collections, Exception Handling, and Object-Oriented Programming concepts.

Sample Programming exercise based on MITMC202 Introduction to AI and Prompt Engineering

1. Generate a short story using Generative AI.
2. Generate a poem using AI.

3. Generate a business letter using AI.
4. Generate a resume using AI assistance.
5. Demonstrate tokenization using sample text.
6. Analyse token count of different prompts.
7. Create examples of AI hallucinations.
8. Write a simple prompt and analyse output.
9. Improve an ineffective prompt.
10. Create prompts with clear instructions.
11. Create prompts with contextual information.
12. Design prompts for educational content generation.
13. Design prompts for business report generation.
14. Design prompts for coding assistance.
15. Design prompts for data analysis.
16. Create prompts for translation tasks.
17. Create prompts for summarization tasks.
18. Perform Zero-Shot Prompting.
19. Perform One-Shot Prompting.
20. Perform Few-Shot Prompting.
21. Implement Chain-of-Thought prompting.
22. Create Role-Based prompts for teachers.
23. Create Role-Based prompts for software developers.
24. Create Role-Based prompts for researchers.
25. Implement Step-by-Step prompting.
26. Experiment with Tree-of-Thought prompting.
27. Design prompts using RTF framework.
28. Design prompts using CARE framework.
29. Design prompts using RISEN framework.
30. Design prompts using CRISPE framework.

31. Evaluate prompt quality using defined metrics.
32. Generate blog articles using AI.
33. Generate marketing content using AI.
34. Generate social media posts using AI.
35. Create presentation outlines using AI.
36. Use AI for code generation.
37. Use AI for debugging Python programs.
38. Generate SQL queries using AI.
39. Generate charts and visualization suggestions using AI.
40. Create lesson plans using AI.