MSCS ASSESSMENT PLAN
School of Information Technology

Program Educational Objectives:

The program educational objectives (PEO) of the computer science program are as follows:

A. Be a successful practitioner or researcher in a computer science related field or pursuing a PhD.

B. Be able to apply theory and development skills to design creative and effective solutions to practical computing problems.

C. Demonstrate effective teamwork and leadership skills

D. Communicate effectively using different modalities

Student Outcomes:

At the time of graduation, a student in our computer science program will have the ability to:

1. Demonstrate an in-depth knowledge of software engineering and its application to the broader area of computer science.
2. Demonstrate an in-depth knowledge of operating systems and its application to the broader area of computer science.
3. Demonstrate an in-depth knowledge of theory, algorithm design, and their applications in computer science
4. Demonstrate an in-depth knowledge of machine learning and big data and their applications in computer science
5. Communicate effectively in various professional contexts
6. Function effectively as a member or leader of a team engaged in computer science-related activities.

Relationship of Student Outcomes to Program Educational Objectives

The table below summarizes the relationship between student outcomes and program educational objectives:

	Program Educational Objectives			
Student Outcomes	A	B	C	D
1	×	×		
2	×	×		
3	×	×		
4	×	×		
5	×			X
6			X	

1. Demonstrate an in-depth knowledge of software engineering and its application to the broader area of computer science.

Performance Indicator students should be able to	Delivery Course/Methods	Courses used for Assessment	Assessment Methods	Data Needed	Assessed Groups	Expected level of attainment*	Timeline
a. Demonstrate a comprehensive understanding of software methodologies, their usage, and ability to analyze their applicability to a problem.	IT 426	IT 426	Rubric 1(a)	IT 426: Project and Assignments	IT 426 students	70%	Odd Spring
b. Demonstrate knowledge of software architectures, their implementation and evaluate them to a problem	IT 426	IT 426	Rubric 1(b)	IT 426: Project	IT 426 students	70%	Odd Spring
c. Apply software design principles, design patterns, and their best practices towards software design and implementation	IT 426	IT 426	Rubric 1(c)	IT 426: Project or Assignments	IT 426 students	70%	Odd Spring
d. Understand software testing, testing strategies, and ability to write software tests	IT426	IT426	Rubric 1(d)	IT 426: Project	IT 426 students	70%	Odd Spring

Rubric 1(a): Software Methodologies				
	Poor or Non-Existent	Developing	Developed	Exemplary
Understands Agile methodologies their principles, and usage	Does not show any understanding of Agile methodologies and their principles	Understands the Agile methodologies and their principles but misunderstands their usage in various contexts	Understands the Agile methodologies, their principles and usage in general for various contexts	Understands the Agile methodologies, their principles, and usage in various context, while also explaining in detail the pros and cons of using an Agile methodology.

Rubric 1(b): Software Architecture				
	Poor or Non-Existent	Developing	Developed	Exemplary
Understands different software architecture and their principles	Does not show any understanding of software architectures and their principles	Understands the software architectures and their principles but misunderstands the difference between them	Understands software architectures, their principles, and differences between them	Understands software architectures, their principles, differences between them, and also pros and cons of using a software architecture for various contexts
Demonstrates the ability to use software architectures	Does not show any understanding of how to use software architectures to solve a problem	Demonstrates the ability to use software architectures to solve a problem but in general, is confused about the choices for a problem	Demonstrates the ability to use software architecture(s) to solve a problem	Demonstrates the ability to use software architecture(s) to solve a problem along with a detailed analysis of different architectural approaches for the problem

Rubric 1(c): Software Design Principles and Patterns				
	Poor or Non-Existent	Developing	Developed	Exemplary
Understands the software design principles and patterns	Does not show any understanding of software design principles and patterns	Understands software design principles and patterns but misunderstands the difference between them	Understands software design principles and patterns and the difference between them	Understands software design principles and patterns, differences between them, and also pros and cons of using them in various context

Demonstrates the ability to use software design principles and patterns to solve problems	Does demonstrates any understanding how to implement software design principles and patterns	Demonstrates the ability to use software design principles and patterns to solve a problem but is confused about the choices for a given problem	Demonstrates the ability to use software design principles and patterns to solve a problem	Demonstrates the ability to use software design principles and patterns to solve a problem with a detailed analysis of the different approach

Rubric 1(d): Software Testing				
	Poor or Non-Existent	Developing	Developed	Exemplary
Demonstrates the ability to perform software testing	Does not have the ability to perform software testing	Demonstrates the ability to perform software testing and its principles but misuses them for a given problem	Demonstrates the ability to perform software testing in line with software testing principles and best practices for a given problem	Demonstrates the ability to perform software testing in line with software testing principles and best practices by laying out a detailed testing strategy/plan for a given problem

2. Demonstrate an in-depth knowledge of operating systems and its application to the broader area of computer science.

Performance Indicator students should be able to	Delivery Course/Methods	Courses used for Assessment	Assessment Methods	Data Needed	Assessed Groups	Expected level of attainment*	Timeline
a. Demonstrate the knowledge of operating system.	IT 483	IT 483	Rubric 2(a)	IT 483: Tests and assignments that deal with operating system.	IT 483 students	70%	Odd Fall
b. Apply the knowledge of operating system to analyze and organize the infrastructure of a computing system	IT 483, IT 382	IT 483	Rubric 2(b)	IT 483: Tests, homework, programming assignments, and final project.	IT 483 students	70%	Odd Fall

Rubric 2(a): demonstration of operating system knowledge				
	Poor or Non-Existent	Developing	Developed	Exemplary
Demonstrates an understanding of the structure of an operating system and identify the policies and algorithms used for process management, memory management, process synchronization, file system, I/O device management.	Does not show any understanding of the major components of operating systems such as process, thread, paging, scheduler, interprocess communication, file system, I/O device drivers.	Explains the major components of operating systems such as process, thread, paging, scheduler, interprocess communication, file system, I/O device drivers, but does not demonstrate the understanding of policies and algorithms used for process management, memory management, process synchronization, file system, I/O device management.	Explains the major components of operating systems such as process, thread, paging, scheduler, interprocess communication, file system, I/O device drivers, and demonstrates the understanding of policies and algorithms used for process management, memory management, process synchronization, file system, I/O device management.	Explains the major components of operating systems and the policies and algorithms used for process management, memory management, process synchronization, file system, I/O device management. In addition, explains the benefits and limitations of different policies or algorithms (e.g., paging vs. segmentation, log file system and original Unix file system).
Demonstrates an understanding of software factors determining the computing performance and how to analyze the performance.	Does not show any understanding of software factors determining computing performance and how to analyze the performance using the given performance parameters.	Explains the software factors determining computing performance and how to analyze the performance using the given performance parameters (e.g., mutli-core CPU architecture, pipelining CPU architecture, cache hit rate, TLB miss rate, structure of page table, disk access time, and multithreaded programming), but does not demonstrate how to analyze performance using given performance parameters.	Explains the software factors determining computing performance and how to analyze performance using the given performance parameters (e.g., mutli-core CPU architecture, pipelining CPU architecture, cache hit rate, TLB miss rate, structure of page table, disk access time, and multithreaded programming).	Explains the software factors determining computing performance and how to analyze the performance using the given performance parameters and explains the benefits and limitations of different approaches to enhance computing performance.

Rubric 2(b): Apply operating system knowledge to solve problems				
	Poor or Non-Existent	Developing	Developed	Exemplary
Apply the knowledge operating system to analyze and organize the infrastructure of a computing system.	Does not a show comprehensive understanding of operating systems	Understands operating system and know how the performance of a computing system is affected by the operating system	Being to apply the knowledge to analyze the performance of a computing based on operating system.	Being to apply the knowledge to implement or deploy a operating system and analyze its performance based on application.

3. Demonstrate an in-depth knowledge of theory, algorithm design, and their applications in computer science

Performance Indicator Students should be able to	Delivery Course/Methods	Courses used for Assessment	Assessment Methods	Data Needed	Assessed Groups	Expected level of attainment*	Timeline
a. Demonstrate a comprehensive knowledge of computing theory including being able to solve computational problems and identify the limitation of computation.	IT 428	IT 428	Rubric 3(a)	IT 428: Tests, homework, and programming assignments.	IT 428 students	70%	Even Fall
b. Being able design, analyze, and implement varieties of algorithms to solve computing problem.	IT 427	IT 427	Rubric 3(b)	IT 427: Tests, homework, programming assignments, and final project.	IT 427 students	70%	Odd Fall

Rubric 3(a): Theory				
	Poor or Non-Existent	Developing	Developed	Exemplary
Deterministic vs. Nondeterministic models.	Do not know the meaning and difference between the two models.	Be aware of the difference between the two models and know how to use the nondeterministic model to describe/solve a few problems.	Know how to use the nondeterministic model to describe/solve all typical problems and understand its complexity implications in classical models.	Understand the complexity and computability implications of the nondeterministic model under different constraints and the separation between complexity

				classes. (e.g., NFA vs DFA, PDA vs DPDA, P vs NP)
Turing Machines and Computability.	Do not know the definition of Turing machines and the issue of computability.	Knows the definition, components, and operations of Turing machines and know how to use Turing machine to accept some non-regular and non-context-free languages.	Knows the concept of computability and Universal Turing Machines and know the limitation of Turing Machines and be able to identify a few undecidable problems. Also, know the technique of diagonalization and dove tailing in the context of computability.	Know the degrees beyond Turing computable such as Recursively Enumerable and Co- Recursively Enumerable and know the technique of Turing reduction.
Regular Languages and Finite State Machines.	Do not know and understand the definition of regular languages.	Know at least two of the three formalisms to describe regular languages, i.e., Regular expressions, Regular Grammars, and Finite State Automata.	Know all equivalent formalisms for regular languages and know how to convert from one to another.	Know how to optimize Finite State Machines (from NFA to DFA and minimize the state) and the limitation of regular languages.
Context-free Languages and Push-Down Automata.	Do not know and understand the definition of context-freer languages.	Know Context-free Grammars and Pushdown Automata.	Know how to convert between Context-free grammar and Pushdown Automata.	Can Identify some ambiguous Context-free languages and the limitation of deterministic Pushdown Automata.

Rubric 3(b): Algorithms				
	Poor or Non-Existent	Developing	Developed	Exemplary
Advanced Data Structures	Have trouble to understand and implement basic data structures, such as linked lists, tree, binary search trees.	Understand and being able to implement basic data structures and know how to analyze their time/space complexity.	Understand and being able to implement some advanced data structures such as red-black tree, B-tree, Huffman tree, and varieties of heaps.	Know how to implement and analyze time/space complexity of advanced data structure covered in the class and know how to use them in different algorithm.
Big-O and Asymptotic notations.	Do not know how to compare the efficiency of algorithms using big-O notations.	Being able to use big-O notations in the analysis of some simple algorithms (like straightforward nested loop in the worst cases)	Know how to use big-O notations to analyze most algorithms including recursive functions in the worst¸ best, and average cases.	In addition to the developed concepts described in the previous column, know other asymptotic notations and know their applications.

		but fail to analyze average cases and recursion.		
Divide and Conquer	Do not know what is divide and conquer approach.	Know the concept of divide and conquer algorithms but fail to correctly apply the approach to solve problems.	Being able to correctly implement divide and conquer algorithms to solve some typical problems.	Being able to correctly implement divide and conquer algorithms and know how to analyze their time complexity.
Dynamic Programming and Greedy Methods.	Do not know the concepts of dynamic programming and greedy methods.	Know the differences between dynamic programming and greedy methods but fail to apply the concepts to solve problems.	Being able to correctly implement dynamic programming and greedy methods to solve some typical problems.	Being able to correctly implement dynamic programming and greedy methods and know how to analysis their time/space complexity.
The concepts of NP-Completeness.	Do not know P vs NP.	Know the definitions of P, NP, and NPC.	Know how to prove some typical problems being NPC and the implication of NP=P.	Know common approaches to prove or disprove NP=P. Also know some problem and Co-NP and beyond.

4. Demonstrate an in-depth knowledge of machine learning and big data and their applications in computer science

Performance Indicator Students should be able to	Delivery Course/Methods	Courses used for Assessment	Assessment Methods	Data Needed	Assessed Groups	Expected level of attainment*	Timeline
a. Write queries to interact with big data storage systems efficiently	IT 441	IT 441	Rubric 4(a)	IT441: Projects, Exams or Assignments	IT 441 students	70%	Even Fall
b. Compare and contrast appropriate evaluation metrics for supervised learning predictive tasks	IT 448	IT 448	Rubric 4(b)	IT448: Projects, Exams or Assignments	IT 448 students	70%	Even Spring

c. Compare and contrast appropriate evaluation metrics for supervised learning predictive tasks	IT 448	IT 448	Rubric 4(c)	IT448: Projects, Exams or Assignments	IT 448 students	70%	Even Spring

Rubric 4(a): Big Data and Storage				
	Poor or Non-Existent	Developing	Developed	Exemplary
Write queries to interact with big data storage systems efficiently	Does not know the query syntax	Writes queries to retrieve, store, update big data in big data storage systems, but the queries may not always work correctly	Successfully writes and executes various queries to big data storage systems based on full understanding of the given big data architectures. The queries include retrieving, storing, and updating data	Successfully writes and executes various queries to big data storage systems based on full understanding of the given big data architectures. The queries include retrieving, storing, and updating data. Also, demonstrate how the query results can be used for data analysis

Rubric 4(b): Classification Problems				
	Poor or Non-Existent	Developing	Developed	Exemplary
Design and implement an effective solution to classification problem	Classification program does not work. Does not address effective strategies for data preprocessing, representations, and partitioning into training and testing sets.	Classification program works partially. Partially addresses effective strategies for data preprocessing, representations, and partitioning into training and testing sets.	Classification program works correctly. Addresses effective strategies for data preprocessing, representations, and partitioning into training and testing sets.	Classification program works correctly. Addresses effective strategies for data preprocessing, representations, partitioning into training and testing sets, and selecting hyperparameters.

Rubric 4(c): Metrics for Supervised Learning				
	Poor or Non-Existent	Developing	Developed	Exemplary

Compare and contrast appropriate evaluation metrics for supervised learning predictive tasks	Does not use correct evaluation metrics and does not reach correct conclusions	Develops partially correct evaluation metrics and reaches partially correct conclusions	Develops correct evaluation metrics and reaches correct conclusions	Identifies and presents detailed and correct evaluation and reaches correct conclusion.

5. Communicate effectively in various professional contexts

Performance Indicator	Delivery Methods	Courses used for Assessment	Assessment Methods	Data Needed	Assessed Groups	Expected level of attainment *	Timeline
a. Communicates effectively with other professional (Oral)	IT 426, IT 483, IT 388	IT 426	Rubric 5(a)	Project Presentation	IT 426 students	70%	Odd Spring

Rubric 5(a)				
	Poor or Non-Existent	Developing	Developed	Exemplary
Clarity	Not assertive or clear	Vague or inconsistent in presentation	Clear and easy to understand	Clear, assertive, and very easy to understand
Organization	Not well organized, no logical flow	Inconsistent flow, lacking macro or micro-organization	Logically organized at micro and macro level	Entire communication has logical flow, flow is reinforced throughout
Audience	Not aimed at the intended audience	Reflects own knowledge rather than targeting audience, should have taken more efforts to direct talk at audience	Directed at appropriate audience	Targeting audience well enough to enhance communication
Engaging the audience	Not captivating, could not engage audience, little to no interaction with audience	Some engagement, but not enough interaction with audience	Keeps the audience interested and has some interaction	Keeps the audience awake and involved, occasionally adapting to audience's feedback

Delivery	Two or more of: Spoke too fast/too slow, did not address intended questions, inappropriate attire, took significantly longer or shorter than allotted time	One of: Spoke too fast/too slow, too many pauses, awkward body language	Spoke at appropriate pace, comfortable and appropriate body language	Calm. Clear diction. Good tone. Good pacing. Appropriate attire and personal grooming.

6. Function effectively as a member or leader of a team engaged in computer science-related activities

Performance Indicator	Delivery Methods	Courses used for Assessment	Assessment Methods	Data Needed	Assessed Groups	Expected level of attainment *	Timeline
a. Participates in team activities	IT 426, IT 388	IT 426	Rubric 6(a)	Peer evaluation	IT 426 students	70%	Odd Spring
b. Completes the project on time	IT 426, IT 388	IT 426	Rubric 6(b)	Project deliverables	IT 426 students	70%	Odd Spring
c. Leads team activities	IT 426, IT 388	IT 426	Rubric 6(c)	Peer evaluation	IT 426 students	70%	Odd Spring

Rubric 6: Teamwork and leadership				
	Poor or Non-Existent	Developing	Developed	Exemplary
a. Participates in team activities	Does not contribute to any team activities	Contributes occasionally to team activities	Contributes equally to team activities	Contributes to team activities, takes initiative, and helps other team members
b. Completes the project on time	Does not complete the project on time	Completes a subset of project requirements	Satisfactorily completes all the project requirements	Completes all the project requirements and establishes detailed documentation on the

				project design, implementation, and other artifacts
c. Leads team activities	Does not establish a project plan, assign tasks to individual team members, and track their progress	Establishes a project plan, but does not assign responsibilities to team members and track their progress	Establishes a project plan, assigns responsibilities to all team members, and tacks their progress periodically satisfactorily	Establishes a project plan, assigns responsibilities to all team members, tacks their progress periodically, mentor and motivate the team members to meet individual and team goals

MSCS-2-year assessment cycle (Quick overview for implementation)		
Semester	Course to be Assessed	What is assessed
Even Spring	448	4(b), 4(c)
Odd Fall	427	3(b)
	483	2(a), 2(b)
Odd Spring	426	1(a), 1(b), 1(c), 1(d), 5(a), 6(a), 6(b), 6(c)
Even Fall	428	3(a)
	441	4(a)