Data Structures and Algorithms for Games

Module Information

2022.01, Approved

Summary Information

Module Code	5107COMP
Formal Module Title	Data Structures and Algorithms for Games
Owning School	Computer Science and Mathematics
Career	Undergraduate
Credits	20
Academic level	FHEQ Level 5
Grading Schema	40

Teaching Responsibility

LJMU Schools involved in Delivery
Computer Science and Mathematics

Learning Methods

Learning Method Type	Hours
Lecture	22
Practical	33

Module Offering(s)

Display Name	Location	Start Month	Duration Number Duration Unit
SEP-CTY	CTY	September	12 Weeks

Aims and Outcomes

Aims	To study abstract data types (ADTs) and common implementations of these data types.To gain an understanding of how a game application utilises both Parallel and Sequential Algorithms.To develop applications using the Boost/STL C++ containers in order to implement games applications.To understand the application of the data-oriented design and Object-Oriented Design paradigms on Data Structures. To build on programming skills through implementation of algorithms and use of appropriate data structures in problem solving for games development. To recognise and specify how complexity of operations on these ADTs and their overall performance characteristics are affected by both the ADT in question and its implementation strategy

After completing the module the student should be able to:

Learning Outcomes

Code	Number	Description
MLO1	1	Appraise a range of fundamental data structures and their operations.
MLO2	2	Analyse and differentiate fundamental algorithms' (such as insert, remove, search, sort) time and space complexity characteristics as applied to a range of ADT implementations.
MLO3	3	Evaluate Sequential vs Parallel Algorithm Strategies and their relationship to CPU / GPU architectures.
MLO4	4	Apply appropriate data structures in order to solve specific sub problems related to game and game engine development.
MLO5	5	Construct user-defined ADTs in C++ and utilise the Boost/STL APIs in order to store data related to both game mechanics and game engine architecture.
MLO6	6	Develop software that utilises appropriate algorithms and data structures to satisfy a technical game design specification.

Module Content

Outline Syllabus	Algorithms for structure operations; insert, remove, retrievalAlgorithms for structure navigation; search and sortAlgorithm types: iterative and recursiveRelationship between ADTs and computing fundamentals (e.g. Stack, Queue)Use of Big O notation to specify time complexity for simple algorithmsDebug / Profiling tools for Internal State Examination.Debug / Profiling tools for algorithmic execution analysis (e.g. Frame Analysis)Memory Management: Pointers, References, Heap, Stack. STL and Boost: algorithms, iterators, and containers.Introduction to Parallel Programming- Goals of parallelism (e.g., throughput) versus concurrency (e.g., controlling access to shared resources)- Parallelism, communication, and coordination- Programming constructs for coordinating multiple simultaneous computations- Need for synchronization- Data races (simultaneous read/write or write/write of shared state)- Higher-level races (interleavings violating program intention, undesired non-determinism)- Lack of liveness/progress (deadlock, starvation)- Game Specific Structures: - Scene Graphs - BSP/Quad Tree/kd-tree/OctTree- Frustums. - Linear Data Managers and Object Aliasing- Broadphase Collision Containers- Component-Based Game Objects.- Buffers and Queues – CPU vs GPU Representation.- BitSets and Enumerations- Enumerated Dictionaries. - Pipelines and Caches- Type Semantics, SISD vs SIMD Access Traits. - Review of physical memory and memory management hardwareData Oriented Programming vs. Object Programming solutions for Data Structure Representation. Brute Force/Greedy/Divide and ConquerC++ TemplatesC++ Class and Pointer Memory LayoutsC++ Specific Optimisation Strategies.Sequential vs. parallel processingParallel programming vs. concurrent programmingRequest parallelism vs. Task parallelismAbstract Data Types and common implementation strategies:- Linear ADTs:- Lists (Arrays, Linked Lists)- Stacks, Queues- Non-Linear ADTs:- Trees, Binary Trees, BSTs- Maps (ListMaps, BSTMaps, HashMaps)- Graphs- Undirected- Directed- Weighted- Graph Theory Fundamentals (Nodes, Edges etc).
Module Overview	
Additional Information	This module is a technical, skills-focused module. Students will be introduced to Data Structures as Abstract Data Types and cover the fundamentals of discrete mathematics which provide the foundations for data structure design and implementation. Students will gain exposure of how to create their own implementation in C++ and the C++ Standard Template Library Implementations. Each structure will be covered from a games programming perspective, demonstrating real-world usage within game engines and game applications. The module will culminate with a look into parallel algorithms and how games apply parallelisation across CPU and GPU hardware using data structures. It will require previous experience in programming; It will build on existing programming-based skills such as problem / functional decomposition and the use of an IDE to develop and test programs.

Assessments

Assignment Category	Assessment Name	Weight	Exam/Test Length (hours)	Module Learning Outcome Mapping
Artefacts	Extending Game Using DSA	60	0	MLO4, MLO5, MLO6
Centralised Exam	Examination	40	1.5	MLO1, MLO2, MLO3

Module Contacts

Module Leader

Contact Name	Applies to all offerings	Offerings
Yann Savoye	Yes	N/A

Partner Module Team

Contact Name	Applies to all offerings	Offerings