PROJECT AND TEAM INFORMATION

Project Title

Dynamic Memory Allocator(Heap Management System)

Student / Team Information

Team Name:	*MEMORY MINDS* *DAA-IV-T032*
Team member 1 (Team Lead)	*Gupta, Riya – 23022643* *23022643@geu.ac.in* 
Team member 2	*Dwivedi, Shubh – 230112175* *shubhdwivedi.230112175@gehu.ac.in* 

Team member 3	*Kumari,Bhumi-23022707* *23022707@geu.ac.in* 

PROPOSAL DESCRIPTION

Motivation

Modern operating systems and applications rely heavily on dynamic memory allocation to efficiently manage system resources during program execution. As processes frequently request and release memory at runtime, effective memory allocation becomes essential to ensure optimal performance and avoid resource wastage.A critical challenge in memory management is to dynamically allocate and deallocate memory blocks in a way that minimizes **internal and external fragmentation**, reduces memory overhead, and maximizes space utilization. Without proper memory allocation techniques, systems can experience performance degradation, increased latency, and even memory exhaustion.This project aims to develop an efficient **Dynamic Memory Allocator (Heap Management System)** capable of managing heap memory by implementing well-known memory allocation strategies such as **First-Fit**, **Best-Fit**, **Next-Fit**, and **Buddy System Allocation**. Each of these strategies has unique strengths and trade-offs in terms of allocation speed, memory utilization, and fragmentation reduction.

State of the Art / Current solution

Today, dynamic memory allocation problems are addressed using a variety of **memory allocation algorithms and strategies** implemented within operating systems and runtime environments. Common approaches include **First-Fit**, **Best-Fit**, and **Next-Fit** algorithms, which scan through memory to allocate the most suitable available block for a given request. These methods aim to balance allocation speed and space utilization but often suffer from fragmentation issues.To improve memory usage, more advanced techniques like the **Buddy Memory Allocation System** are used. This strategy splits memory into blocks of sizes that are powers of two and merges them when possible, reducing fragmentation and improving allocation efficiency.

Project Goals and Milestones

*The primary goal of this project is to design and implement an efficient **Dynamic Memory Allocator (Heap Management System)** that minimizes fragmentation and optimizes memory usage during allocation and deallocation. The project will simulate and compare various memory allocation strategies to evaluate their performance and efficiency in managing heap memory.*

General Goals:

- *Develop a structured representation of heap memory using either an array-based or linked list-based approach.*
- *Implement and evaluate key memory allocation algorithms: **First-Fit**, **Best-Fit**, **Next-Fit**, and **Buddy System Allocation**.*

- *Handle **memory deallocation**, **garbage collection**, and **fragmentation reduction**.*
- *Measure and compare **internal and external fragmentation** for different strategies.*
- *Optimize allocation methods to improve performance and memory utilization*

Initial Milestones:

1. ***Heap Memory Representation***
   Design the structure for managing free and allocated memory blocks.
2. ***Implementation of Basic Allocation Strategies***
   Implement First-Fit, Best-Fit, and Next-Fit algorithms with proper block management.
3. ***Testing & Performance Comparison***
   Analyze each strategy based on speed and fragmentation.

Further Milestones:

4. ***Buddy Memory Allocation***
   Implement dynamic splitting and merging of memory blocks using the Buddy System.
5. ***Memory Deallocation and Garbage Collection***
   Develop functions for freeing memory and coalescing adjacent blocks.
6. ***Performance Analysis and Optimization***
   Compare strategies, calculate fragmentation, and suggest improvements.
7. ***Final Report & Code Documentation***
   Document code, explain design choices, and summarize performance results.

Project Approach

To effectively address the problem of dynamic memory allocation and fragmentation, we plan to articulate and design a custom **Heap Management System** that simulates real-world memory allocation strategies. Our approach focuses on modular design, clarity, and performance evaluation.

We will represent heap memory using **dynamic data structures** such as **linked lists** or **arrays**, where each memory block will maintain metadata (e.g., block size, status: free/allocated, pointers to adjacent blocks). This structure allows flexible allocation, deallocation, and coalescing of memory blocks.

We will implement four key allocation algorithms:

- **First-Fit**
- **Best-Fit**
- **Next-Fit**
- **Buddy System**

Platforms and Technologies:

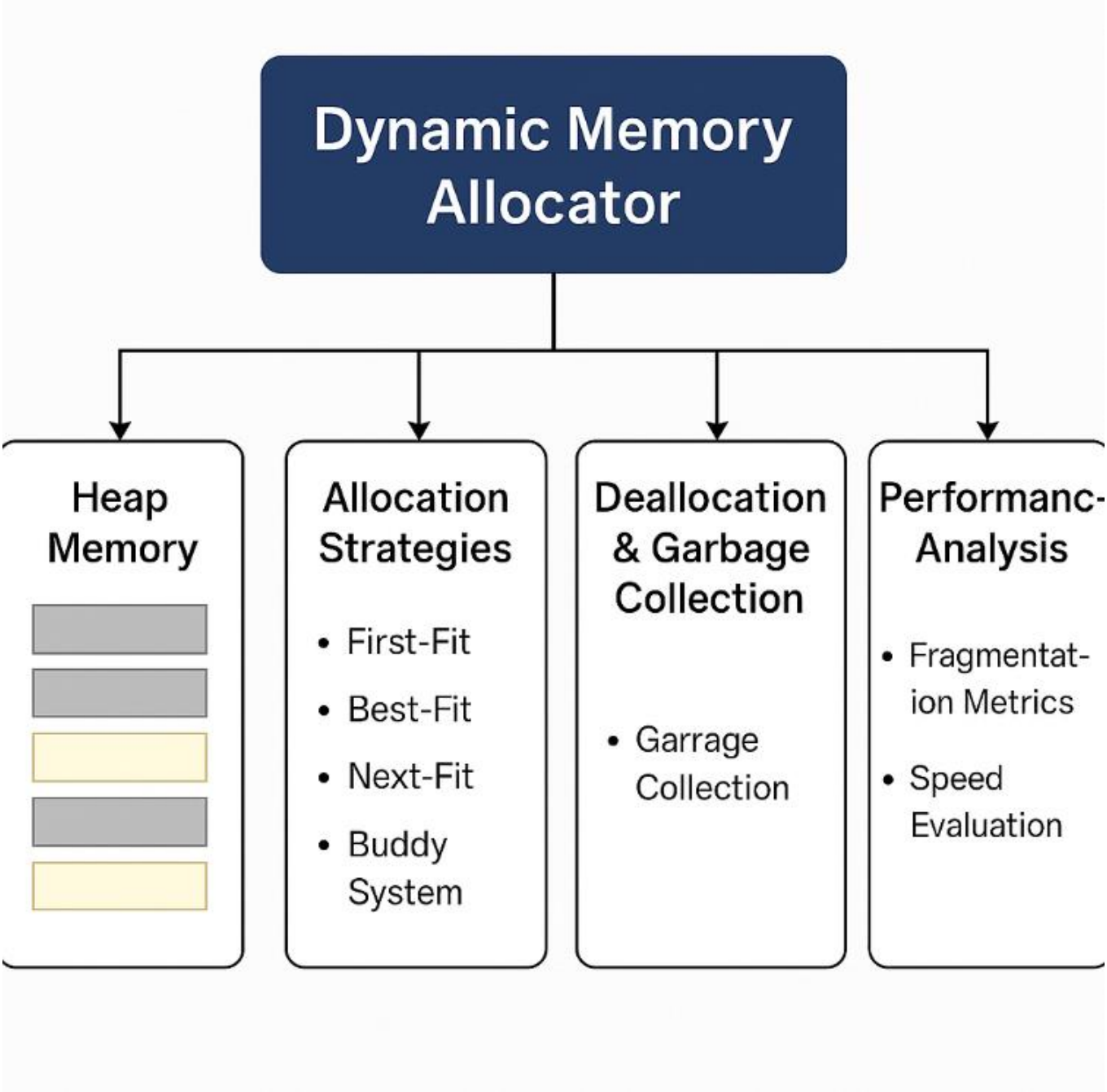
- ***Programming Language:** C/C++ – for low-level memory manipulation and system-level simulation.*
- ***IDE:** Code::Blocks or Visual Studio Code.*

- ***Operating System:*** *Windows or Linux (cross-platform compatibility).*
- ***Visualization (optional):*** *Python (for graphing fragmentation analysis).*
- ***Documentation:*** *Microsoft Word or LaTeX (for report writing), Git (for version control).*

Design Approach:

- *Start with the heap structure and implement allocation strategies one by one.*
- *Create test cases for each scenario to evaluate performance.*
- *Measure* ***internal and external fragmentation*** *and compare results.*
- *Use modular programming for clean, maintainable, and testable code.*
- *Conclude with a detailed performance analysis and improvement suggestions.*

System Architecture (High Level Diagram)



Project Outcome / Deliverables

The main outcome of this project is a fully functional ***Dynamic Memory Allocator (Heap Management System)*** *that simulates various memory allocation strategies used in operating systems. The system will demonstrate efficient allocation, deallocation , and fragmentation management of heap memory through multiple algorithms.*

Key Deliverables:

1. ***Source Code:***
   - *Well-structured and documented code implementing First-Fit, Best-Fit, Next-Fit, and Buddy System allocation strategies.*
   - *Includes functions for memory deallocation and garbage collection simulation.*
2. ***Final Project Report:***
   - *Describes problem definition, methodology, implementation details, performance comparison, and conclusions.*
   - *Includes diagrams, test results, and fragmentation analysis.*
3. ***Performance Analysis Charts:***
   - *Comparisons of fragmentation levels and speed across different allocation methods.*
4. ***Executable / Simulation Output:***
   - *A user-friendly program or command-line simulation demonstrating memory allocation and deallocation.*

Assumptions

To simplify the implementation of the Dynamic Memory Allocator, we assume a fixed-size heap memory block is available at the start of execution. All memory requests are handled sequentially, and memory sizes are provided in bytes. We assume there is no multitasking or concurrent memory access. Memory blocks are aligned properly, and metadata overhead is minimal. During simulation, user input or predefined requests will be used to test allocation and deallocation. External storage or disk memory is not considered, and all operations are limited to the simulated heap space within main memory.

References

[Memory Allocation in C (malloc, calloc, realloc, free)]
https://www.geeksforgeeks.org/memory-allocation-in-c-using-malloc-calloc-free-and-realloc/

[First Fit, Best Fit, and Worst Fit Allocation Strategies]
https://www.geeksforgeeks.org/first-fit-best-fit-and-worst-fit/

[Buddy Memory Allocation System]
https://www.geeksforgeeks.org/buddy-memory-allocation-program/

[Garbage Collection in Java (conceptual understanding)]
https://www.geeksforgeeks.org/garbage-collection-in-java/

[Introduction to Artificial Intelligence (GeeksforGeeks)]
https://www.geeksforgeeks.org/introduction-to-artificial-intelligence/

[Artificial Intelligence in Operating Systems]
https://www.geeksforgeeks.org/artificial-intelligence-in-operating-system/