**Dynamic Memory Management Visualizer**

**1. Introduction**

Dynamic memory management is a crucial aspect of modern operating systems, allowing processes to request and release memory during execution. This project implements a **Dynamic Memory Management Visualizer** that demonstrates three primary memory allocation techniques:

* **First-Fit**: Allocates the first available block that is large enough.
* **Best-Fit**: Allocates the smallest available block that meets the process's requirements.
* **Worst-Fit**: Allocates the largest available block to the process.

The visualizer provides a clear understanding of how these algorithms function and their efficiency in managing memory.

**2. Objective**

The main objective of this project is to:

* Implement and visualize First-Fit, Best-Fit, and Worst-Fit memory allocation algorithms.
* Allow users to input memory block sizes and simulate process allocations.
* Display real-time memory status after each allocation.

**3. Methodology**

1. **Input Handling**: Users input the number of memory blocks and their sizes.
2. **Algorithm Selection**: Users choose between First-Fit, Best-Fit, or Worst-Fit algorithms.
3. **Process Allocation**: For each algorithm, the system searches and allocates memory.
4. **Visualization**: The memory status is displayed after each operation.

**4. Implementation**

The project is implemented using the **C programming language** for efficient execution and direct memory manipulation.

**Key Features:**

* Supports up to 10 memory blocks.
* Dynamic allocation using user input.
* Real-time visualization of memory status.
* Simple menu-driven interface for ease of use.

**Execution Steps:**

1. Compile the code using:
2. gcc DynamicMemoryVisualizer.c -o visualizer
3. Run the program:
4. ./visualizer

**5. Code Explanation**

The code consists of the following main sections:

1. **Data Structures**:
   * A MemoryBlock structure to store the size and allocation status.
2. **Algorithms**:
   * first\_fit(): Finds the first suitable block.
   * best\_fit(): Finds the smallest suitable block.
   * worst\_fit(): Finds the largest suitable block.
3. **User Interface**:
   * Users can choose an allocation method and monitor memory status dynamically.

**6. Output**

Sample output for the system:

Enter the number of memory blocks (max 10): 5

Enter size of Block 1: 100

Enter size of Block 2: 200

Enter size of Block 3: 50

Enter size of Block 4: 300

Enter size of Block 5: 150

1. First-Fit

2. Best-Fit

3. Worst-Fit

4. Display Memory

5. Exit

Choose an option: 1

Enter process size: 120

Process of size 120 allocated in Block 2

**7. Conclusion**

The Dynamic Memory Management Visualizer effectively demonstrates the difference between First-Fit, Best-Fit, and Worst-Fit algorithms. This project provides a practical understanding of memory allocation in operating systems, which is essential for efficient resource management.

**8. Future Scope**

* Implementing additional algorithms like Next-Fit.
* Adding graphical visualization for better clarity.
* Simulating fragmentation and memory compaction techniques.

**9. References**

* Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). **Operating System Concepts**.
* Tanenbaum, A. S. (2015). **Modern Operating Systems**.