MEMORY MANAGEMENT SIMULATION PROJECT

Dynamic Memory Management Simulation with Compaction and First-Fit Allocation Strategy

CREATED BY:
ASIYA FATIMA TABASSUM (CS-22064)

SUBMITTED TO: DR. ZAREEN SADIQ

BATCH:2022

Objective

- The purpose of this project is to simulate dynamic memory allocation and deallocation in a system, handling fragmentation through memory compaction.
- Processes are read from a file, and memory is allocated based on a First-Fit strategy.
- If allocation fails, processes enter a waiting queue and compaction is attempted to reduce fragmentation.

System Overview

The system simulates:

- Arrival of processes over time
- Memory allocation using First-Fit
- Handling fragmentation via memory compaction
- Waiting queue for processes that could not be immediately allocated
- Deallocation upon process completion
- Statistics reporting at the end

Key Data Structures

Structure	Description
MemoryBlock	Represents a block of memory with start address, size, and status (FREE or ALLOCATED).
Process	Contains process details: arrival time, size, burst time, and process ID.
RunningProcess	Tracks currently running processes and their finish times.
WaitingProcess	Queue for processes waiting for memory allocation.

Main Functional Components

a. Memory Initialization

Entire memory initialized as a single free block.

Function:

- MemoryBlock* initialize_memory(int size);

b. Memory Allocation

First-Fit Algorithm: Find the first free block large enough for the process.

If block is larger than required, split it into allocated and free parts.

Function:

- int allocate_first_fit(MemoryBlock **head, int size, int *start_addr);

c. Memory Deallocation

On process completion, memory block is freed.

Adjacent free blocks are merged to reduce external fragmentation.

Functions:

- void deallocate(MemoryBlock **head, int start_addr);
- void merge_free_blocks(MemoryBlock **head);

d. Memory Compaction

Compaction moves all allocated blocks to the beginning of memory.

Free space consolidated at the end.

Updates all running process start addresses accordingly.

Function:

- void compact_memory(MemoryBlock **head, int mem_size, RunningProcess **running);

e. Waiting Queue Management

If immediate allocation fails, process enters a waiting queue.

Retry allocations from the queue, up to 3 attempts, compacting memory if necessary.

Functions:

- void add_to_waiting(WaitingProcess **head, Process p);
- void retry_waiting(WaitingProcess **waiting, MemoryBlock **memory, RunningProcess **running, int current_time, int mem_size, int *alloc_count);

f. Process Execution Cycle

Every time unit:

- Check for completed processes and deallocate memory.
- Allocate memory to newly arrived processes.
- Attempt to allocate memory for waiting processes. Print the current memory state.

g. Statistics Generation

At the end of simulation:

- Total number of processes
- Number of successfully allocated processes
- Average, largest, and smallest process size

Function:

- void print_stats(Process *processes, int count, int allocated);

6. Sample Flow of Program Execution

User inputs:

1. Total memory size (RAM size in KB)
2. Process file name containing process information (arrival time, size, burst time)

Simulation starts:

- Memory layout printed at each time unit
- Arrival and allocation (or queuing) of processes
- Completion and deallocation of running processes
- Retry allocation for waiting processes with memory compaction if needed

At the end:

- Detailed final statistics are printed.
- Memory is deallocated and program exits gracefully.

Important Features

- First-Fit Memory Allocation strategy
- Dynamic splitting and merging of memory blocks

- Memory Compaction for handling fragmentation
- Process waiting and retrying logic with limited attempts
- Clear visual display of the memory map at each time step
- Efficient memory cleanup to avoid leaks

Error Handling

- Checks if process file opens successfully.
- Verifies that at least 10 processes are loaded (as per requirement).
- Handles allocation failure with a waiting and retry mechanism.
- Graceful handling of memory deallocation.

Files and Inputs

Process Input File format: each line contains

<arrival_time> <process_size> <burst_time>

1. 0 150 5
2. 1 300 3
3. 2 75 4
4. 3 300 2 5. 4 280 6
6. 5 180 3
7. 6 80 4
8. 7 220 5 9. 8 450 2 10. 9 280 3

Potential Improvements

- Implement Best-Fit and Worst-Fit strategies for comparison.
- Introduce priority for waiting processes.
- Add visual animations of memory movement during compaction. Handle process suspension and resumption.

Conclusion

This project successfully models a simplified dynamic memory management system.

It gives a clear understanding of:

- Challenges in memory allocation
- Effects of fragmentation
- Importance of memory compaction
- Real-world problems in operating systems regarding memory handling.

```
hp@DESKTOP-4PPU8MQ UCRT64 /c/Users/hp/OS OEL
$ ./memory_manager.exe
Memory Management Module
Enter total RAM size: 1000
Enter process file: processes.txt

Simulation Start (Memory: 1000 KB)

Time 0:
Process 1 arrives (size 150)
Allocated immediately at 0
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |       150 | Alloc    |
|           150 |       850 | Free     |
+---------------+-----------+----------+

Time 1:
Process 2 arrives (size 300)
Allocated immediately at 150
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |       150 | Alloc    |
|           150 |       300 | Alloc    |
|           450 |       550 | Free     |
+---------------+-----------+----------+

Time 2:
Process 3 arrives (size 75)
Allocated immediately at 450
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |       150 | Alloc    |
|           150 |       300 | Alloc    |
|           450 |        75 | Alloc    |
|           525 |       475 | Free     |
+---------------+-----------+----------+
```

```
Time 3:
Process 4 arrives (size 300)
Allocated immediately at 525
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |       150 | Alloc    |
|           150 |       300 | Alloc    |
|           450 |        75 | Alloc    |
|           525 |       300 | Alloc    |
|           825 |       175 | Free     |
+---------------+-----------+----------+
```

```
Time 4:
Process 300 at 150 (size 300) completed
Process 5 arrives (size 280)
Allocated immediately at 150
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |       150 | Alloc    |
|           150 |       280 | Alloc    |
|           430 |        20 | Free     |
|           450 |        75 | Alloc    |
|           525 |       300 | Alloc    |
|           825 |       175 | Free     |
+---------------+-----------+----------+

Time 5:
Process 300 at 525 (size 300) completed
Process 150 at 0 (size 150) completed
Process 6 arrives (size 180)
Allocated immediately at 525
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |       150 | Free     |
|           150 |       280 | Alloc    |
|           430 |        20 | Free     |
|           450 |        75 | Alloc    |
|           525 |       180 | Alloc    |
|           705 |       295 | Free     |
+---------------+-----------+----------+
```

```
Time 6:
Process 75 at 450 (size 75) completed
Process 7 arrives (size 80)
Allocated immediately at 0
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |        80 | Alloc    |
|            80 |        70 | Free     |
|           150 |       280 | Alloc    |
|           430 |        95 | Free     |
|           525 |       180 | Alloc    |
|           705 |       295 | Free     |
+---------------+-----------+----------+

Time 7:
Process 8 arrives (size 220)
Allocated immediately at 705
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |        80 | Alloc    |
|            80 |        70 | Free     |
|           150 |       280 | Alloc    |
|           430 |        95 | Free     |
|           525 |       180 | Alloc    |
|           705 |       220 | Alloc    |
|           925 |        75 | Free     |
+---------------+-----------+----------+
```

```
Time 8:
Process 180 at 525 (size 180) completed
Process 9 arrives (size 450)
Immediate allocation failed. Adding to queue
[QUEUE] Process 9 (size 450) added to waiting queue
[QUEUE] Trying process 9 (size 450, attempts: 0)
[QUEUE] Insufficient total memory (420 < 450)
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |        80 | Alloc    |
|            80 |        70 | Free     |
|           150 |       280 | Alloc    |
|           430 |       275 | Free     |
|           705 |       220 | Alloc    |
|           925 |        75 | Free     |
+---------------+-----------+----------+
```

```
Time 9:
Process 10 arrives (size 280)
Immediate allocation failed. Adding to queue
[QUEUE] Process 10 (size 280) added to waiting queue
[QUEUE] Trying process 9 (size 450, attempts: 1)
[QUEUE] Insufficient total memory (420 < 450)
[QUEUE] Trying process 10 (size 280, attempts: 0)
[COMPACT] Fragmentation detected. Attempt 1
[COMPACT] Starting memory compaction...
[COMPACT] Compaction completed. New memory layout:
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |        80 | Alloc    |
|            80 |       280 | Alloc    |
|           360 |       220 | Alloc    |
|           580 |       420 | Free     |
+---------------+-----------+----------+

[QUEUE] Allocation successful after 1 attempts
Allocated process 10 at 580
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |        80 | Alloc    |
|            80 |       280 | Alloc    |
|           360 |       220 | Alloc    |
|           580 |       280 | Alloc    |
|           860 |       140 | Free     |
+---------------+-----------+----------+
```

```
Time 10:
Process 80 at 0 (size 80) completed
Process 280 at 80 (size 280) completed
[QUEUE] Trying process 9 (size 450, attempts: 2)
[COMPACT] Fragmentation detected. Attempt 1
[COMPACT] Starting memory compaction...
[COMPACT] Compaction completed. New memory layout:
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |       220 | Alloc    |
|           220 |       280 | Alloc    |
|           500 |       500 | Free     |
+---------------+-----------+----------+
```

```
[QUEUE] Allocation successful after 3 attempts
Allocated process 9 at 500
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |       220 | Alloc    |
|           220 |       280 | Alloc    |
|           500 |       450 | Alloc    |
|           950 |        50 | Free     |
+---------------+-----------+----------+

Time 11:
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |       220 | Alloc    |
|           220 |       280 | Alloc    |
|           500 |       450 | Alloc    |
|           950 |        50 | Free     |
+---------------+-----------+----------+

Time 12:
Process 450 at 500 (size 450) completed
Process 280 at 220 (size 280) completed
Process 220 at 0 (size 220) completed
Current Memory Map:
+---------------+-----------+----------+
| Start Address |   Size    |  Status  |
+---------------+-----------+----------+
|             0 |      1000 | Free     |
+---------------+-----------+----------+


Final Statistics:
+--------------------+----------+
| Metric             | Value    |
+--------------------+----------+
| Total Processes    |       10 |
| Successfully Alloc'd |       10 |
| Average Process Size |   231.50 |
| Largest Process    |      450 |
| Smallest Process   |       75 |
+--------------------+----------+
```