Example of Buddy System

Initial State:

- Memory pool: 1024 KB (a single block)

Step 1: Process Requests 100 KB – 2^x

X can be any value varying from 1 to 10

2, 4, 8, 16, 32, 64, 128, 256, 512, 1024

- The smallest power of 2 greater than 100 KB is 128 KB.
- The 1024 KB block is split:
  - Two 512 KB blocks are created.
  - One 512 KB block is further split into two 256 KB blocks.
  - One 256 KB block is split into two 128 KB blocks.
- A 128 KB block is allocated to the process.

Memory State:

- 1 block of 512 KB
- 1 block of 256 KB
- 1 block of 128 KB (allocated)
- 1 block of 128 KB (free)

Step 2: Process Requests 200 KB

- The smallest power of 2 greater than 200 KB is 256 KB.
- A 256 KB block is allocated directly.

Memory State:

- 1 block of 512 KB
- 1 block of 256 KB (allocated)
- 1 block of 128 KB (allocated)
- 1 block of 128 KB (free)

Step 3: Process Frees 128 KB

- The 128 KB block is freed. Now, there are two free 128 KB blocks.
- These are "buddies" and are merged into a single 256 KB block.

Memory State:

- 1 block of 512 KB
- 1 block of 256 KB (allocated)
- 1 block of 256 KB (free)

Step 4: Process Frees 256 KB

- The 256 KB block is freed. Now, there are two free 256 KB blocks.
- These are "buddies" and are merged into a single 512 KB block.

Final Memory State:

- 1 block of 1024 KB (fully free)

Best Fit Example:

Initial Memory State:

Consider the following free memory blocks (in KB):

- Block 1: 200 KB
- Block 2: 300 KB
- Block 3: 400 KB
- Block 4: 100 KB
- Block 5: 150 KB

Processes to be Allocated:

1. Process A requires 120 KB.
2. Process B requires 250 KB.
3. Process C requires 50 KB.

Allocation Process:

1. **Process A (120 KB):**
    - Search for the smallest block ≥ 120 KB: Block 5 (150 KB).
    - Allocate Process A to Block 5.
    - Remaining free block: 150 KB - 120 KB = 30 KB.

    Updated Memory Blocks:

    - Block 1: 200 KB
    - Block 2: 300 KB
    - Block 3: 400 KB
    - Block 4: 100 KB
    - Block 5: 30 KB (remaining after allocation to Process A)

---

2. **Process B (250 KB):**
    - Search for the smallest block ≥ 250 KB: Block 2 (300 KB).
    - Allocate Process B to Block 2.
    - Remaining free block: 300 KB - 250 KB = 50 KB.

    Updated Memory Blocks: (Utilized memory blocks are not listed)

    - Block 1: 200 KB
    - Block 2: 50 KB (remaining after allocation to Process B)
    - Block 3: 400 KB
    - Block 4: 100 KB

- Block 5: 30 KB

---

3. **Process C (50 KB):**
   - Search for the smallest block ≥ 50 KB: Block 2 (50 KB).
   - Allocate Process C to Block 2.
   - Remaining free block: 50 KB - 50 KB = 0 KB.

   Updated Memory Blocks:

   - Block 1: 200 KB
   - Block 2: 0 KB (fully allocated to Process C)
   - Block 3: 400 KB
   - Block 4: 100 KB
   - Block 5: 30 KB

Final Memory State:

- Block 1: 200 KB (free)
- Block 2: 0 KB (allocated to Process C)
- Block 3: 400 KB (free)
- Block 4: 100 KB (free)
- Block 5: 30 KB (free)

Quick Fit Example:

Initial Memory State:

Memory blocks are divided into size-specific lists:

- 50 KB List: [Block 1, Block 2]
- 100 KB List: [Block 3, Block 4]
- 200 KB List: [Block 5]

Processes to Allocate:

1. Process A requires 50 KB.
2. Process B requires 100 KB.
3. Process C requires 200 KB.
4. Process D requires 80 KB.

Allocation Process:

1. **Process A (50 KB):**
   - Check the 50 KB list: [Block 1, Block 2].
   - Allocate Block 1 to Process A.
   - Remove Block 1 from the 50 KB list.

Updated Lists:

- 50 KB List: [Block 2]
- 100 KB List: [Block 3, Block 4]
- 200 KB List: [Block 5]

---

2. **Process B (100 KB):**
   - Check the 100 KB list: [Block 3, Block 4].
   - Allocate Block 3 to Process B.
   - Remove Block 3 from the 100 KB list.

   Updated Lists:

   - 50 KB List: [Block 2]
   - 100 KB List: [Block 4]
   - 200 KB List: [Block 5]

---

3. **Process C (200 KB):**
   - Check the 200 KB list: [Block 5].
   - Allocate Block 5 to Process C.
   - Remove Block 5 from the 200 KB list.

   Updated Lists:

   - 50 KB List: [Block 2]
   - 100 KB List: [Block 4]
   - 200 KB List: []

---

4. **Process D (80 KB):**
   - No exact match in the size-specific lists.
   - Use **best-fit** or another strategy to allocate memory from available blocks. Assume Block 4 (100 KB) is allocated.
   - After allocation, split Block 4 into an 80 KB allocated block and a 20 KB free block (if splitting is allowed).

   Updated Lists:

   - 50 KB List: [Block 2]
   - 100 KB List: []
   - 200 KB List: []
   - New 20 KB Block is added to a separate small block list.

Final Memory State:

- Allocated Blocks:
  - Process A: 50 KB (Block 1)

    - Process B: 100 KB (Block 3)
    - Process C: 200 KB (Block 5)
    - Process D: 80 KB (from Block 4)
- Free Blocks:
    - 50 KB List: [Block 2]
    - 20 KB List: [New small block from Block 4]

Next Fit Example:

Initial Memory State:

Consider memory blocks of the following sizes (in KB):

- Block 1: 200 KB (free)
- Block 2: 300 KB (free)
- Block 3: 100 KB (free)
- Block 4: 500 KB (free)
- Block 5: 50 KB (free)

Processes to Allocate:

1. Process A requires 120 KB.
2. Process B requires 450 KB.
3. Process C requires 90 KB.

Allocation Process:

1. **Process A (120 KB):**
    - Start from the beginning.
    - Block 1 (200 KB) is sufficient.
    - Allocate Process A to Block 1.
    - Update the search pointer to the end of Block 1.

    Updated Memory State:

    - Block 1: 80 KB (free, remaining after allocation to Process A)
    - Block 2: 300 KB (free)
    - Block 3: 100 KB (free)
    - Block 4: 500 KB (free)
    - Block 5: 50 KB (free)

---

2. **Process B (450 KB):**
    - Start searching from Block 1 (end of the last allocation).
    - Block 2 (300 KB): Insufficient.
    - Block 3 (100 KB): Insufficient.
    - Block 4 (500 KB): Sufficient.
    - Allocate Process B to Block 4.
    - Update the search pointer to the end of Block 4.

    Updated Memory State:

- Block 1: 80 KB (free)
- Block 2: 300 KB (free)
- Block 3: 100 KB (free)
- Block 4: 50 KB (free, remaining after allocation to Process B)
- Block 5: 50 KB (free)

---

3. **Process C (90 KB):**
   - Start searching from Block 4 (end of the last allocation).
   - Block 5 (50 KB): Insufficient.
   - Loop around to Block 1.
   - Block 1 (80 KB): Insufficient.
   - Block 2 (300 KB): Sufficient.
   - Allocate Process C to Block 2.
   - Update the search pointer to the end of Block 2.

   Updated Memory State:

   - Block 1: 80 KB (free)
   - Block 2: 210 KB (free, remaining after allocation to Process C)
   - Block 3: 100 KB (free)
   - Block 4: 50 KB (free)
   - Block 5: 50 KB (free)

Final Memory Allocation:

- Process A: Allocated 120 KB in Block 1.
- Process B: Allocated 450 KB in Block 4.
- Process C: Allocated 90 KB in Block 2.