praveenkumar.R
192324012

Experiment-22:Construct a C program to implement the best fit algorithm of memory management.

Aim:

To implement the Best Fit memory allocation algorithm to allocate processes to memory blocks in an efficient way, ensuring the smallest leftover space in memory after allocation.

Procedure:

1. Input:
    - A list of memory block sizes.
    - A list of process sizes that need to be allocated memory.
2. Initialization:
    - Initialize an array to keep track of which block is allocated to which process. Set all values initially to -1 (indicating no allocation).
3. Best Fit Allocation:
    - For each process, find the memory block that can accommodate the process's size and has the least remaining space after allocation (the "best fit").
    - Allocate the process to this block by updating the allocation array and reducing the block size by the process size.
    - If no suitable block is found for a process, mark it as "Not Allocated".
4. Output:
    - Display the allocation of blocks to processes, showing the process number, process size, and the allocated block number (if allocated). If no block is found for the process, display "Not Allocated".

Code

```
#include <stdio.h>


void bestFit(int blockSize[], int m, int processSize[], int n) {
   int allocation[n];


   for (int i = 0; i < n; i++) {
```

```
    allocation[i] = -1;
}

for (int i = 0; i < n; i++) {
    int bestIdx = -1;
    for (int j = 0; j < m; j++) {
        if (blockSize[j] >= processSize[i]) {
            if (bestIdx == -1 || blockSize[bestIdx] > blockSize[j]) {
                bestIdx = j;
            }
        }
    }

    if (bestIdx != -1) {
        allocation[i] = bestIdx;
        blockSize[bestIdx] -= processSize[i];
    }
}

printf("Process No.\tProcess Size\tBlock no.\n");
for (int i = 0; i < n; i++) {
    printf("%d\t\t%d\t\t", i + 1, processSize[i]);
    if (allocation[i] != -1) {
        printf("%d", allocation[i] + 1);
    } else {
        printf("Not Allocated");
    }
```

```
        printf("\n");
    }
}

int main() {
    int blockSize[] = {100, 500, 200, 300, 600};
    int processSize[] = {212, 417, 112, 426};
    int m = sizeof(blockSize) / sizeof(blockSize[0]);
    int n = sizeof(processSize) / sizeof(processSize[0]);

    bestFit(blockSize, m, processSize, n);
    return 0;
}
```

Output:

Output

```
Process No. Process Size    Block no.
1       212     4
2       417     2
3       112     3
4       426     5
```