

Name: A. HARSHA MUKUNDHA

Reg-No: 192324070

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

Aim:

To implement the Best Fit memory allocation algorithm in C, which allocates memory blocks to processes such that the block with the smallest size sufficient for the process is selected.

Algorithm:

1. **Input:** Sizes of memory blocks and processes.
2. **Sort:** Go through each process and find the smallest memory block that fits.
3. **Allocation:**
 - If a suitable block is found, allocate it to the process and update the memory block size.
 - If no suitable block is found, mark the process as unallocated.
4. **Output:** Display allocation details for each process.

Procedure:

1. Define the sizes of memory blocks and processes.
2. Iterate over each process.
3. For each process, check all memory blocks to find the smallest block that fits.
4. Allocate the block, and update its size or mark the process as unallocated.
5. Display the allocation results.

Code:

```
#include <stdio.h>
```

```
#include <limits.h>
```

```
int main() {
```

```
    int blockCount, processCount;
```

```
    // Input number of memory blocks and processes
```

```
    printf("Enter the number of memory blocks: ");
```

```
scanf("%d", &blockCount);

printf("Enter the number of processes: ");

scanf("%d", &processCount);


int blocks[blockCount], processes[processCount];

int allocation[processCount];


printf("Enter sizes of memory blocks: ");

for (int i = 0; i < blockCount; i++) scanf("%d", &blocks[i]);


printf("Enter sizes of processes: ");

for (int i = 0; i < processCount; i++) scanf("%d", &processes[i]);


for (int i = 0; i < processCount; i++) allocation[i] = -1;


for (int i = 0; i < processCount; i++) {

    int bestIdx = -1, bestFit = INT_MAX;

    for (int j = 0; j < blockCount; j++) {

        if (blocks[j] >= processes[i] && blocks[j] < bestFit) {

            bestFit = blocks[j];

            bestIdx = j;


        }


    }

}
```

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

Output:

 **OnlineGDB**
online compiler and debugger for c/c++

Welcome, **Harsha Aripaka** 

Create New Project


My Projects






Classroom new

Learn Programming

Programming Questions

Upgrade

Logout 

Enter the number of memory blocks: 2
Enter the number of processes: 2
Enter sizes of memory blocks: 200 220
Enter sizes of processes: 200 45

Process No.	Process Size	Block No.
1	200	1
2	45	2

...Program finished with exit code 0
Press ENTER to exit console. 