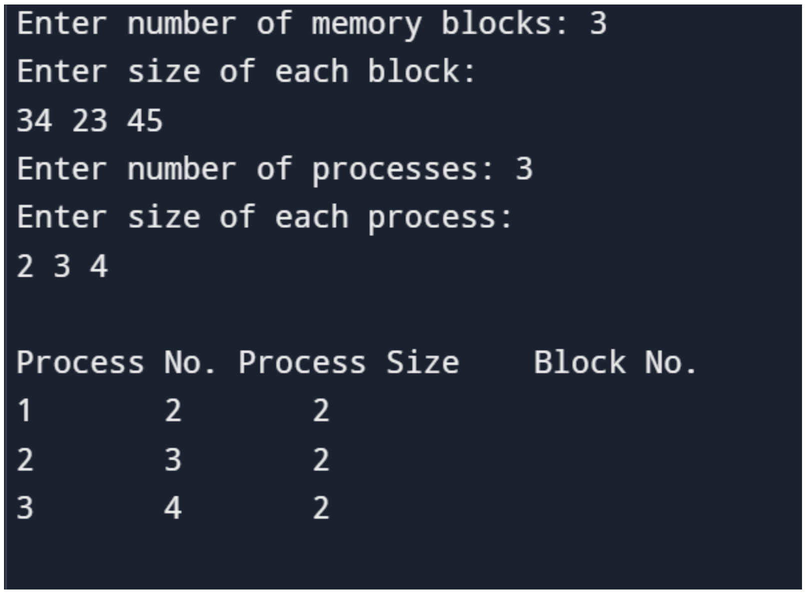
EXP 22: Construct a C program to implement best fit algorithm of memory management.

```
#include <stdio.h>

#define MAX 25

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

  // Initially, all processes are unallocated
  for (int i = 0; i < n; i++)
    allocation[i] = -1;

  // Pick each process and find the best-fit block
  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[j] < blockSize[bestIdx])
          bestIdx = j;
      }
    }

    // If a suitable block is found
    if (bestIdx != -1) {
      allocation[i] = bestIdx;
      blockSize[bestIdx] -= processSize[i];
```

```
        }
    }

    // Print the allocation results
    printf("\nProcess 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\n", allocation[i] + 1);
        else
            printf("Not Allocated\n");
    }
}

int main() {
    int blockSize[MAX], processSize[MAX];
    int m, n;

    printf("Enter number of memory blocks: ");
    scanf("%d", &m);
    printf("Enter size of each block:\n");
    for (int i = 0; i < m; i++)
        scanf("%d", &blockSize[i]);

    printf("Enter number of processes: ");
    scanf("%d", &n);
    printf("Enter size of each process:\n");
    for (int i = 0; i < n; i++)
        scanf("%d", &processSize[i]);
    bestFit(blockSize, m, processSize, n);

    return 0;
}
```

Sample Output



```
Enter number of memory blocks: 3
Enter size of each block:
34 23 45
Enter number of processes: 3
Enter size of each process:
2 3 4

Process No. Process Size    Block No.
1       2       2
2       3       2
3       4       2
```