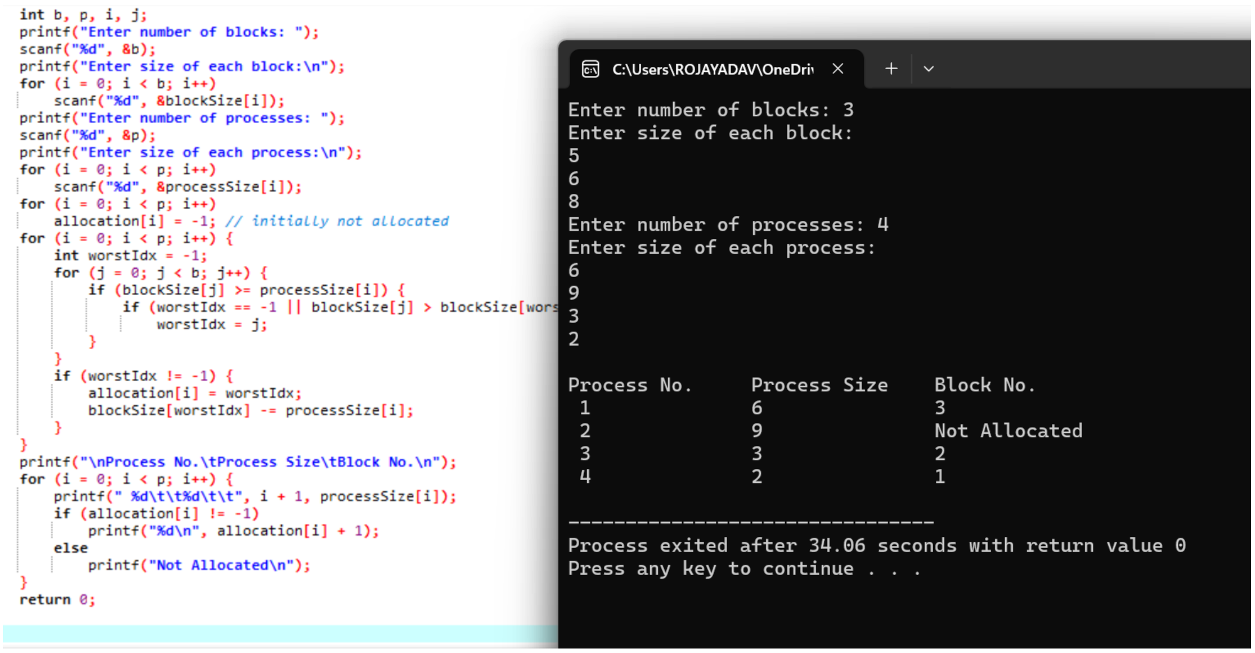
22. Construct a C program to implement best fit algorithm of memory management.

```
#include <stdio.h>

int main() {

    int blockSize[10], processSize[10], allocation[10];

    int b, p, i, j;

    printf("Enter number of blocks: ");

    scanf("%d", &b);

    printf("Enter size of each block:\n");

    for (i = 0; i < b; i++)

        scanf("%d", &blockSize[i]);

    printf("Enter number of processes: ");

    scanf("%d", &p);

    printf("Enter size of each process:\n");

    for (i = 0; i < p; i++)

        scanf("%d", &processSize[i]);

    for (i = 0; i < p; i++)

        allocation[i] = -1; // initially not allocated

    for (i = 0; i < p; i++) {

        int bestIdx = -1;

        for (j = 0; j < b; j++) {

            if (blockSize[j] >= processSize[i]) {

                if (bestIdx == -1 || blockSize[j] < blockSize[bestIdx])

                    bestIdx = j;

            }

        }

        if (bestIdx != -1) {
```

```c
            allocation[i] = bestIdx;
            blockSize[bestIdx] -= processSize[i];
        }
    }
    printf("\nProcess No.\tProcess Size\tBlock No.\n");
    for (i = 0; i < p; 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");
    }
    return 0;
}
```

OUTPUT-



```c
int b, p, i, j;
printf("Enter number of blocks: ");
scanf("%d", &b);
printf("Enter size of each block:\n");
for (i = 0; i < b; i++)
    scanf("%d", &blockSize[i]);
printf("Enter number of processes: ");
scanf("%d", &p);
printf("Enter size of each process:\n");
for (i = 0; i < p; i++)
    scanf("%d", &processSize[i]);
for (i = 0; i < p; i++)
    allocation[i] = -1; // initially not allocated
for (i = 0; i < p; i++) {
    int worstIdx = -1;
    for (j = 0; j < b; j++) {
        if (blockSize[j] >= processSize[i]) {
            if (worstIdx == -1 || blockSize[j] > blockSize[wors
                worstIdx = j;
        }
    }
    if (worstIdx != -1) {
        allocation[i] = worstIdx;
        blockSize[worstIdx] -= processSize[i];
    }
}
printf("\nProcess No.\tProcess Size\tBlock No.\n");
for (i = 0; i < p; 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");
}
return 0;
```

C:\Users\ROJAYADAV\OneDri

```
Enter number of blocks: 3
Enter size of each block:
5
6
8
Enter number of processes: 4
Enter size of each process:
6
9
3
2

Process No.     Process Size    Block No.
 1              6               3
 2              9               Not Allocated
 3              3               2
 4              2               1

--------------------------------
Process exited after 34.06 seconds with return value 0
Press any key to continue . . .
```