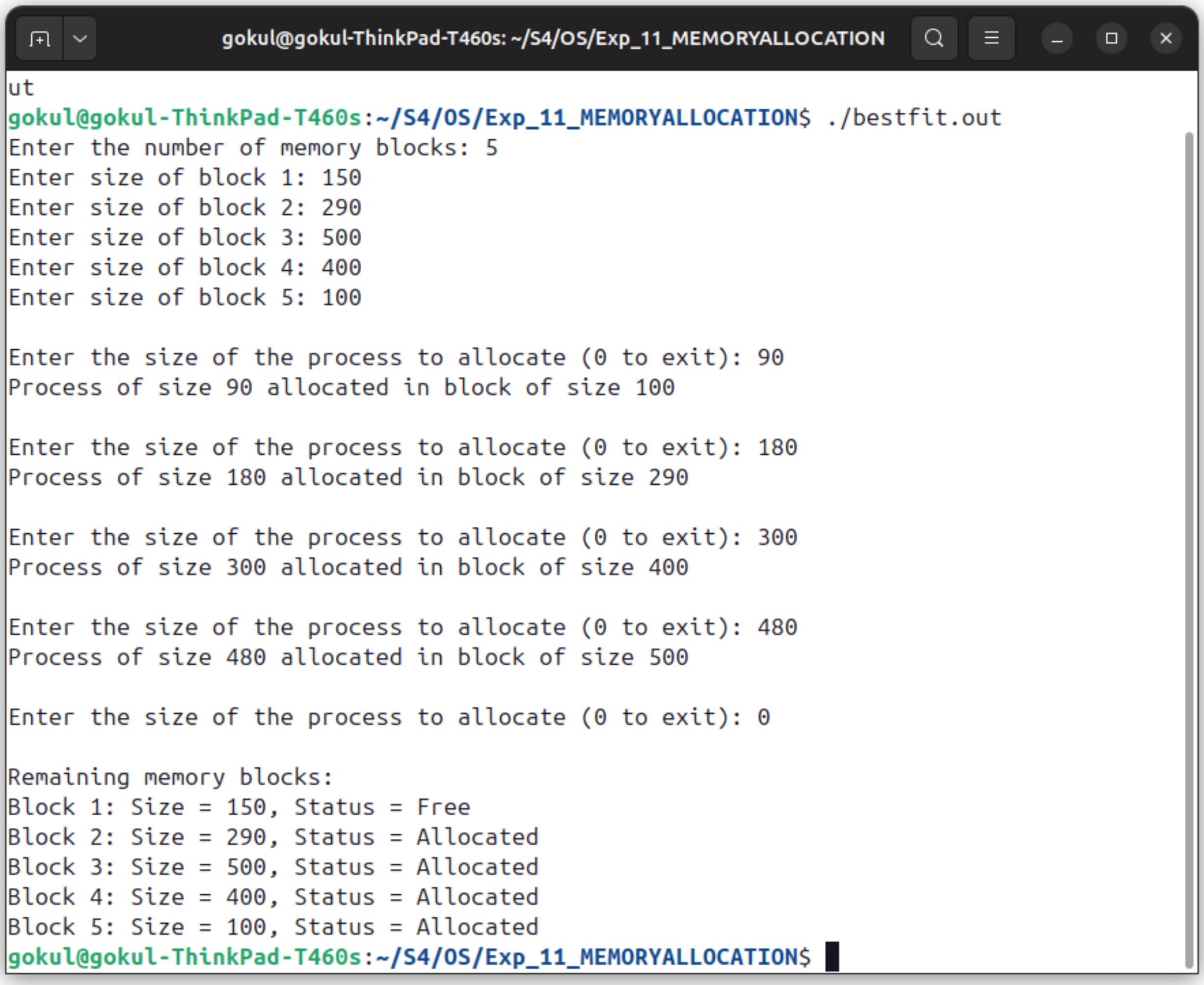
MEMORY ALLOCATION

BEST FIT

Program

```
#include <stdio.h>

typedef struct {
    int size;
    int isFree;
} MemoryBlock;

int bestFit(MemoryBlock blocks[], int numBlocks, int processSize) {
    int bestIndex = -1;
    int minSize = 999999;

    for (int i = 0; i < numBlocks; i++) {
        if (blocks[i].isFree && blocks[i].size >= processSize) {
            if (blocks[i].size < minSize) {
                minSize = blocks[i].size;
                bestIndex = i;
            }
        }
    }

    if (bestIndex != -1) {
        blocks[bestIndex].isFree = 0;
        printf("Process of size %d allocated in block of size %d\n", processSize,
blocks[bestIndex].size);
        return bestIndex;
    } else {
        printf("No suitable block found for process of size %d\n", processSize);
        return -1;
    }
}

int main() {
    MemoryBlock blocks[100];
    int numBlocks;

    printf("Enter the number of memory blocks: ");
    scanf("%d", &numBlocks);

    for (int i = 0; i < numBlocks; i++) {
        printf("Enter size of block %d: ", i + 1);
        scanf("%d", &blocks[i].size);
        blocks[i].isFree = 1;
    }

    int processSize;
    while (1) {
        printf("\nEnter the size of the process to allocate (0 to exit): ");
        scanf("%d", &processSize);
        if (processSize == 0) {
            break;
        }
        bestFit(blocks, numBlocks, processSize);
    }

    printf("\nRemaining memory blocks:\n");
    for (int i = 0; i < numBlocks; i++) {
        printf("Block %d: Size = %d, Status = %s\n", i + 1, blocks[i].size, blocks[i].isFree ? "Free" :
"Allocated");
    }

    return 0;
}
```



```
ut
gokul@gokul-ThinkPad-T460s:~/S4/OS/Exp_11_MEMORYALLOCATION$ ./bestfit.out
Enter the number of memory blocks: 5
Enter size of block 1: 150
Enter size of block 2: 290
Enter size of block 3: 500
Enter size of block 4: 400
Enter size of block 5: 100

Enter the size of the process to allocate (0 to exit): 90
Process of size 90 allocated in block of size 100

Enter the size of the process to allocate (0 to exit): 180
Process of size 180 allocated in block of size 290

Enter the size of the process to allocate (0 to exit): 300
Process of size 300 allocated in block of size 400

Enter the size of the process to allocate (0 to exit): 480
Process of size 480 allocated in block of size 500

Enter the size of the process to allocate (0 to exit): 0

Remaining memory blocks:
Block 1: Size = 150, Status = Free
Block 2: Size = 290, Status = Allocated
Block 3: Size = 500, Status = Allocated
Block 4: Size = 400, Status = Allocated
Block 5: Size = 100, Status = Allocated
gokul@gokul-ThinkPad-T460s:~/S4/OS/Exp_11_MEMORYALLOCATION$
```

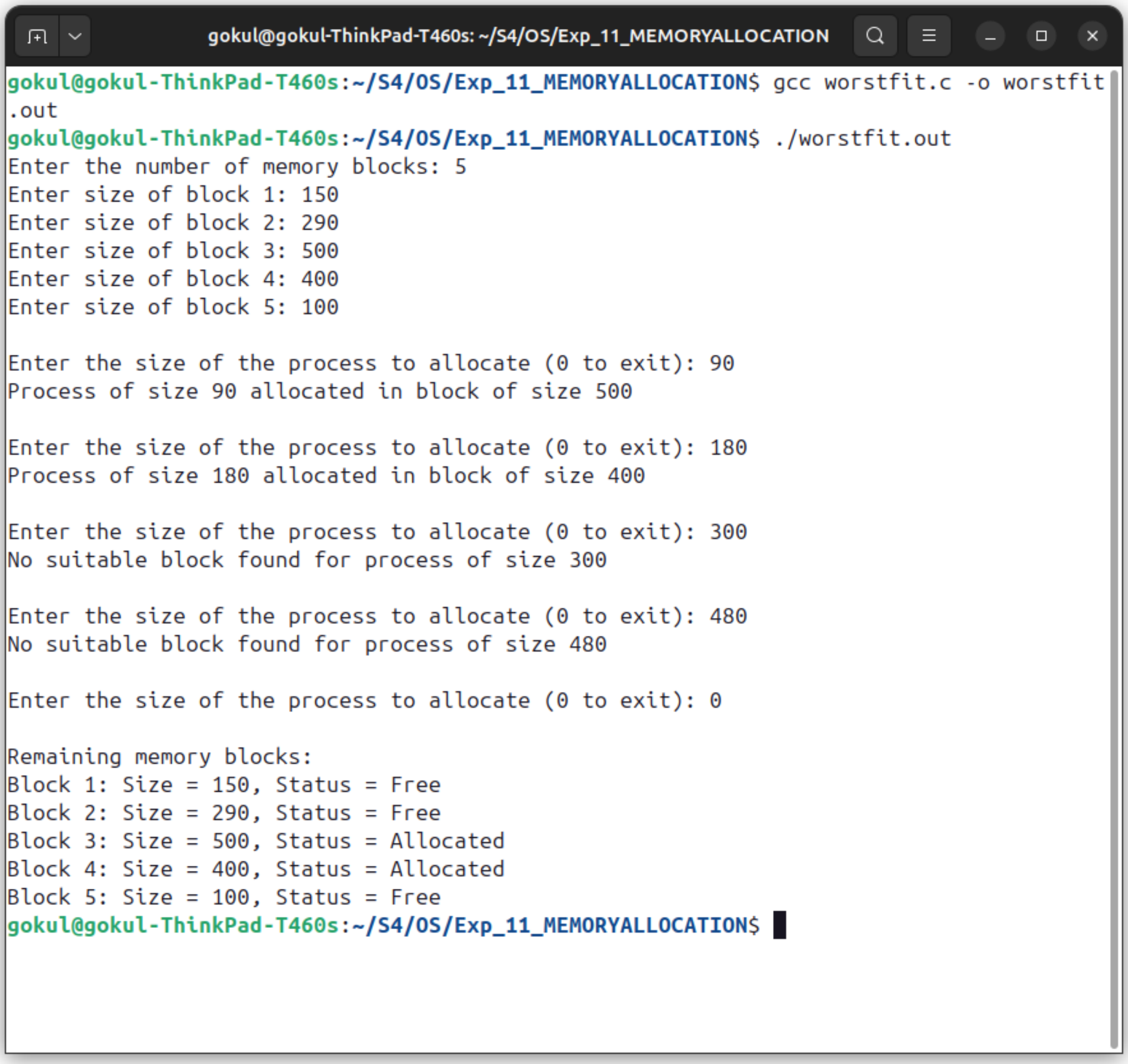
WORSTFIT

program

```
#include <stdio.h>

typedef struct {
    int size;
    int isFree;
} MemoryBlock;

int worstfit(MemoryBlock blocks[], int numBlocks, int processSize) {
    int bestIndex = -1;
    int maxsize = -1;

    for (int i = 0; i < numBlocks; i++) {
        if (blocks[i].isFree && blocks[i].size >= processSize) {
            if (blocks[i].size > maxsize) {
                maxsize = blocks[i].size;
                bestIndex=i;
            }
        }
    }

    if (bestIndex != -1) {
        blocks[bestIndex].isFree = 0;
        printf("Process of size %d allocated in block of size %d\n", processSize,
blocks[bestIndex].size);
        return bestIndex;
    } else {
        printf("No suitable block found for process of size %d\n", processSize);
        return -1;
    }
}
```

```
int main() {
    MemoryBlock blocks[100];
    int numBlocks;

    printf("Enter the number of memory blocks: ");
    scanf("%d", &numBlocks);

    for (int i = 0; i < numBlocks; i++) {
        printf("Enter size of block %d: ", i + 1);
        scanf("%d", &blocks[i].size);
        blocks[i].isFree = 1;
    }
    int processSize;
    while (1) {
        printf("\nEnter the size of the process to allocate (0 to exit): ");
        scanf("%d", &processSize);
        if (processSize == 0) {
            break;
        }
        worstfit(blocks, numBlocks, processSize);
    }

    printf("\nRemaining memory blocks:\n");
    for (int i = 0; i < numBlocks; i++) {
        printf("Block %d: Size = %d, Status = %s\n", i + 1, blocks[i].size, blocks[i].isFree ? "Free" :
"Allocated");
    }

    return 0;
}
```

Output



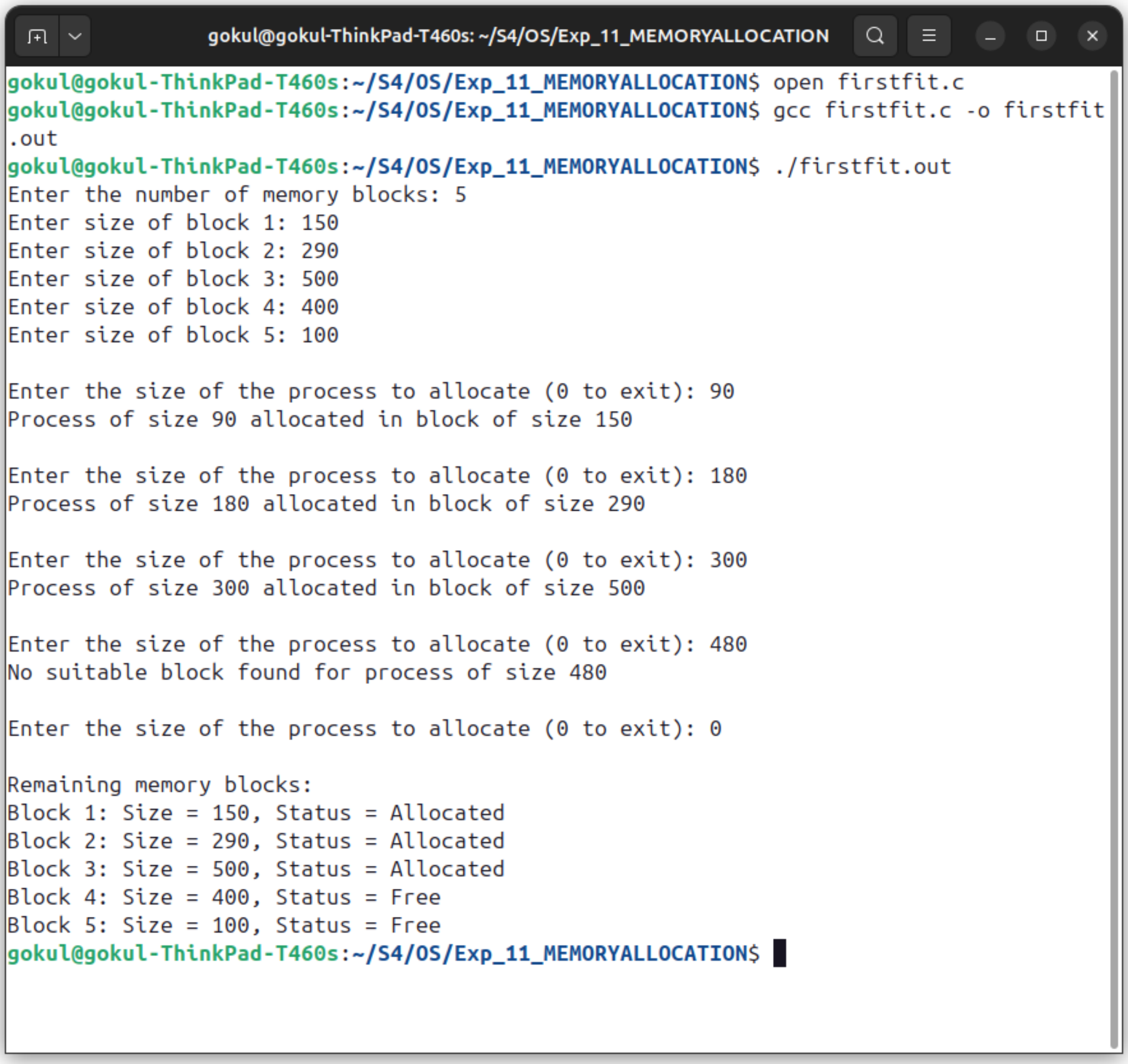
```
gokul@gokul-ThinkPad-T460s:~/S4/OS/Exp_11_MEMORYALLOCATION$ gcc worstfit.c -o worstfit
.out
gokul@gokul-ThinkPad-T460s:~/S4/OS/Exp_11_MEMORYALLOCATION$ ./worstfit.out
Enter the number of memory blocks: 5
Enter size of block 1: 150
Enter size of block 2: 290
Enter size of block 3: 500
Enter size of block 4: 400
Enter size of block 5: 100

Enter the size of the process to allocate (0 to exit): 90
Process of size 90 allocated in block of size 500

Enter the size of the process to allocate (0 to exit): 180
Process of size 180 allocated in block of size 400

Enter the size of the process to allocate (0 to exit): 300
No suitable block found for process of size 300

Enter the size of the process to allocate (0 to exit): 480
No suitable block found for process of size 480

Enter the size of the process to allocate (0 to exit): 0

Remaining memory blocks:
Block 1: Size = 150, Status = Free
Block 2: Size = 290, Status = Free
Block 3: Size = 500, Status = Allocated
Block 4: Size = 400, Status = Allocated
Block 5: Size = 100, Status = Free
gokul@gokul-ThinkPad-T460s:~/S4/OS/Exp_11_MEMORYALLOCATION$ 
```

FIRST FIT

program

```
#include <stdio.h>

typedef struct {
    int size;
    int isFree;
} MemoryBlock;

int firstfit(MemoryBlock blocks[], int numBlocks, int processSize) {
            int bestIndex=-1,i;

            for (i=0;i<numBlocks;i++){
                        if(blocks[i].isFree && blocks[i].size >= processSize){
                        bestIndex = i;
                        break;
                }
            }

    if (bestIndex != -1) {
        blocks[bestIndex].isFree = 0;
        printf("Process of size %d allocated in block of size %d\n", processSize,
blocks[bestIndex].size);
        return bestIndex;
    } else {
        printf("No suitable block found for process of size %d\n", processSize);
        return -1;
    }
}

int main() {
    MemoryBlock blocks[100];
    int numBlocks,index=0;

    printf("Enter the number of memory blocks: ");
    scanf("%d", &numBlocks);

    for (int i = 0; i < numBlocks; i++) {
        printf("Enter size of block %d: ", i + 1);
        scanf("%d", &blocks[i].size);
        blocks[i].isFree = 1;
    }

    int processSize;
    while (1) {
        printf("\nEnter the size of the process to allocate (0 to exit): ");
        scanf("%d", &processSize);
        if (processSize == 0) {
            break;
        }
        firstfit(blocks, numBlocks, processSize);
    }

    printf("\nRemaining memory blocks:\n");
    for (int i = 0; i < numBlocks; i++) {
        printf("Block %d: Size = %d, Status = %s\n", i + 1, blocks[i].size, blocks[i].isFree ? "Free" :
"Allocated");
    }

    return 0;
}
```

Output



gokul@gokul-ThinkPad-T460s: ~/S4/OS/Exp_11_MEMORYALLOCATION

```
gokul@gokul-ThinkPad-T460s:~/S4/OS/Exp_11_MEMORYALLOCATION$ open firstfit.c
gokul@gokul-ThinkPad-T460s:~/S4/OS/Exp_11_MEMORYALLOCATION$ gcc firstfit.c -o firstfit
.out
gokul@gokul-ThinkPad-T460s:~/S4/OS/Exp_11_MEMORYALLOCATION$ ./firstfit.out
Enter the number of memory blocks: 5
Enter size of block 1: 150
Enter size of block 2: 290
Enter size of block 3: 500
Enter size of block 4: 400
Enter size of block 5: 100

Enter the size of the process to allocate (0 to exit): 90
Process of size 90 allocated in block of size 150

Enter the size of the process to allocate (0 to exit): 180
Process of size 180 allocated in block of size 290

Enter the size of the process to allocate (0 to exit): 300
Process of size 300 allocated in block of size 500

Enter the size of the process to allocate (0 to exit): 480
No suitable block found for process of size 480

Enter the size of the process to allocate (0 to exit): 0

Remaining memory blocks:
Block 1: Size = 150, Status = Allocated
Block 2: Size = 290, Status = Allocated
Block 3: Size = 500, Status = Allocated
Block 4: Size = 400, Status = Free
Block 5: Size = 100, Status = Free
gokul@gokul-ThinkPad-T460s:~/S4/OS/Exp_11_MEMORYALLOCATION$
```