```
#include <iostream>
#include <vector>
using namespace std;

struct Process
{
    char Pname[3];
    int memory;
    bool allocated = false;
};
struct Block
{
    int size;
    bool used = false;
    int rem;
    struct Process processAllocated;
};

int main()
{
    cout << "No. of block : ";
    int n;
    cin >> n;
    vector<Block> blocks;
    cout << "Enter Size of the " << n << " Blocks: ";
    for (int i = 0; i < n; i++)
    {
        Block tempBlock;
        cin >> tempBlock.size;
        tempBlock.rem = tempBlock.size;
        blocks.push_back(tempBlock);
    }
    cout << "No. of Process : ";
    int m;
    cin >> m;
    vector<Process> Processes;
    cout << "Enter Name and size of the Processes: ";
    for (int i = 0; i < m; i++)
    {
        Process tempProcess;
        cin >> tempProcess.Pname;
        cin >> tempProcess.memory;
        Processes.push_back(tempProcess);
    }
```

```
    // memory allocation
    for (int i = 0; i < m; i++)
    {
        bool exist = false;
        int index, max = INT16_MIN;
        for (int j = 0; j < n; j++)
        {
            if (Processes[i].memory <= blocks[j].rem && blocks[j].used ==
false && blocks[j].rem > max)
            {
                max = blocks[j].rem;
                exist = true;
                index = j;
            }
        }
        if (exist)
        {
            Processes[i].allocated = true;
            blocks[index].used = true;
            blocks[index].rem = blocks[index].size - Processes[i].memory;
            blocks[index].processAllocated = Processes[i];
        }
    }
    cout << "\tBlock Number\tSize\tProcess Allocated\tInternal
Fragmentation" << endl;
    for (int i = 0; i < n; i++)
    {
        if (blocks[i].used == true)
        {
            cout << "\t\t" << i + 1 << "\t" << blocks[i].size << "\t\t" <<
blocks[i].processAllocated.Pname << "\t\t\t" << blocks[i].rem << endl;
        }
        else
        {
            cout << "\t\t" << i + 1 << "\t" << blocks[i].size << "\t\t"
                 << "---"
                 << "\t\t\t"
                 << "---" << endl;
        }
    }
    bool flag = true;
    for (int i = 0; i < m; i++)
    {
        if (Processes[i].allocated == false)
        {
            flag = false;
```

```
            break;
        }
        else
        {
            continue;
        }
    }

    int IF = 0, EF = 0;
    for (int i = 0; i < n; i++)
    {
        if (blocks[i].used == true)
        {
            IF += blocks[i].rem;
        }
        else
        {
            if (flag == false)
            {
                EF += blocks[i].rem;
            }
        }
    }
    cout << "Total Internal Fragmentation = " << IF << endl;
    cout << "Total External Fragmentation = " << EF << endl;
    return 0;
}
```

Output

```
No. of block : 5
Enter Size of the 5 Blocks: 200 100 300 400 500
No. of Process : 4
Enter Name and size of the Processes: p1 450 p2 210 p3 210 p4 350
        Block Number    Size    Process Allocated       Internal Fragmentation
                1       200             ---                     ---
                2       100             ---                     ---
                3       300             p3                      90
                4       400             p2                      190
                5       500             p1                      50
Total Internal Fragmentation = 330
Total External Fragmentation = 0
```

```
No. of block : 5
Enter Size of the 5 Blocks: 200 100 300 400 500
No. of Process : 4
Enter Name and size of the Processes: p1 250 p2 210 p3 100 p4 350
        Block Number    Size    Process Allocated       Internal Fragmentation
                1       200             ---                     ---
                2       100             ---                     ---
                3       300             p3                      200
                4       400             p2                      190
                5       500             p1                      250
Total Internal Fragmentation = 640
Total External Fragmentation = 0
PS C:\Users\aadil\Desktop\CSE\OS Lab>
```

Thank you