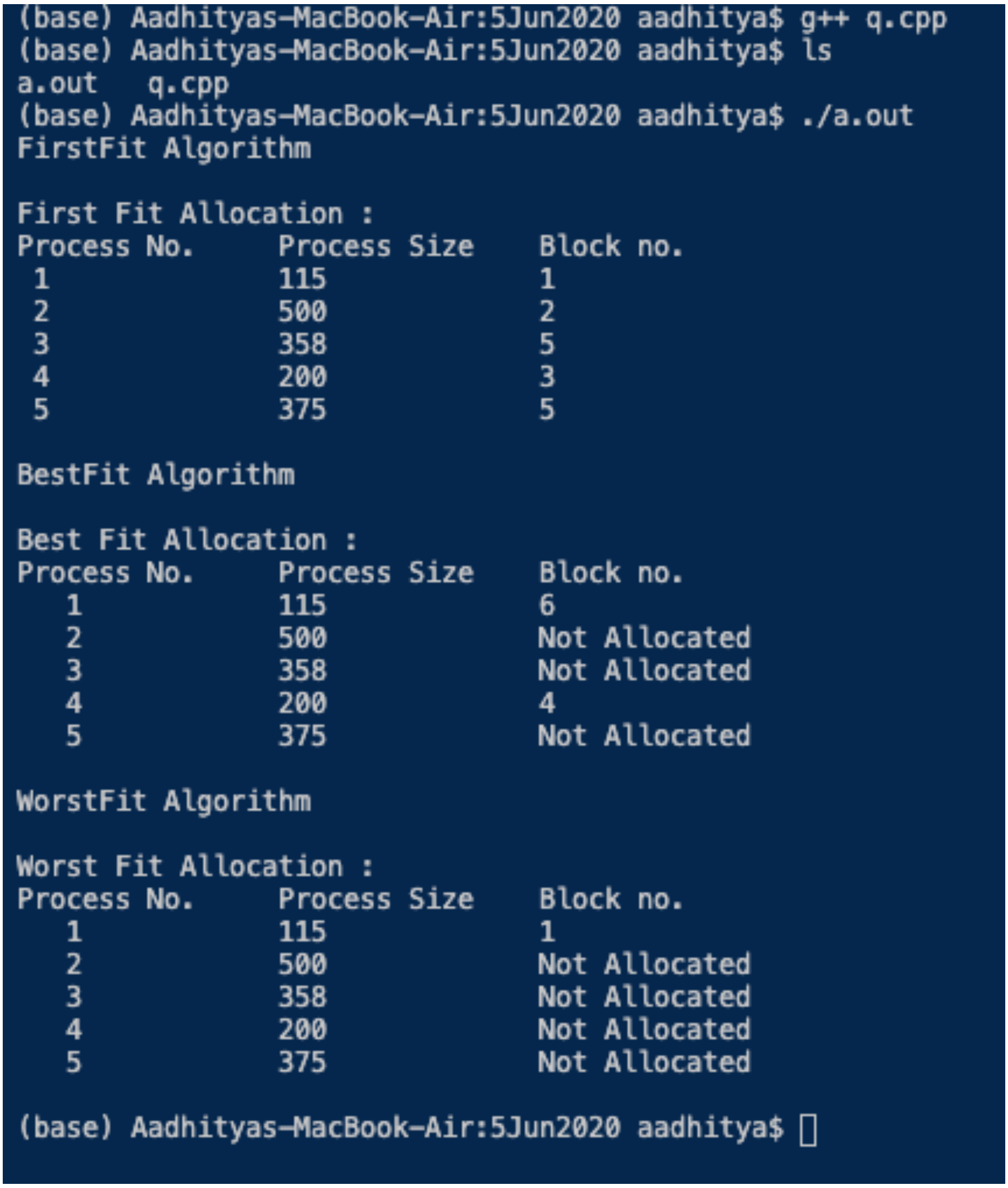
Operating Systems LAB

5-June 2020

Aadhitya Swarnesh I -

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


void firstFit(int blockSize[], int m, int processSize[],
int n)
{
    int allocation[n];
    memset(allocation, -1, sizeof(allocation));

    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < m; j++)
        {
            if (blockSize[j] >= processSize[i])
            {
                allocation[i] = j;
                blockSize[j] -= processSize[i];
                break;
            }
        }
    }
    cout << "First Fit Allocation : ";
    cout << "\nProcess No.\tProcess Size\tBlock no.\n";
    for (int i = 0; i < n; i++)
    {
        cout << " " << i+1 << "\t\t"
             << processSize[i] << "\t\t";
        if (allocation[i] != -1)
            cout << allocation[i] + 1;
        else
            cout << "Not Allocated";
        cout << endl;
    }
}


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

    memset(allocation, -1, sizeof(allocation));

    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)
                    bestIdx = j;
                else if (blockSize[bestIdx] >
blockSize[j])
                    bestIdx = j;
            }
        }
        if (bestIdx != -1)
        {
            allocation[i] = bestIdx;
            blockSize[bestIdx] -= processSize[i];
        }
    }

    cout << "Best Fit Allocation : ";
    cout << "\nProcess No.\tProcess Size\tBlock no.\n";
    for (int i = 0; i < n; i++)
    {
        cout << "   " << i+1 << "\t\t" << processSize[i]
<< "\t\t";
        if (allocation[i] != -1)
            cout << allocation[i] + 1;
        else
            cout << "Not Allocated";
        cout << endl;
    }
}


void worstFit(int blockSize[], int m, int processSize[],
int n)
{
    int allocation[n];
    memset(allocation, -1, sizeof(allocation));
    for (int i=0; i<n; i++)
    {
        int wstIdx = -1;
        for (int j=0; j<m; j++)
        {
            if (blockSize[j] >= processSize[i])
            {
                if (wstIdx == -1)
                    wstIdx = j;
                else if (blockSize[wstIdx] <
blockSize[j])
                    wstIdx = j;
            }
        }
        if (wstIdx != -1)
        {
            allocation[i] = wstIdx;
            blockSize[wstIdx] -= processSize[i];
        }
    }

    cout << "Worst Fit Allocation : ";
    cout << "\nProcess No.\tProcess Size\tBlock no.\n";
    for (int i = 0; i < n; i++)
    {
        cout << "   " << i+1 << "\t\t" << processSize[i]
<< "\t\t";
        if (allocation[i] != -1)
            cout << allocation[i] + 1;
        else
            cout << "Not Allocated";
        cout << endl;
    }
}


int main()
{
    int blockSize[] = {300, 600, 350, 200, 750,125};
    int processSize[] = {115,500, 358, 200, 375};
    int m = sizeof(blockSize) / sizeof(blockSize[0]);
    int n = sizeof(processSize) / sizeof(processSize[0]);
    // FirstFit Algorithm
    firstFit(blockSize, m, processSize, n);
    cout<<"\n";
    // BestFit Algorithm
    bestFit(blockSize, m, processSize, n);
    cout<<"\n";
    // WorstFit Algorithm
    worstFit(blockSize, m, processSize, n);
    cout<<"\n";
    return 0 ;
}
```



```
(base) Aadhityas-MacBook-Air:5Jun2020 aadhitya$ g++ q.cpp
(base) Aadhityas-MacBook-Air:5Jun2020 aadhitya$ ls
a.out   q.cpp
(base) Aadhityas-MacBook-Air:5Jun2020 aadhitya$ ./a.out
FirstFit Algorithm

First Fit Allocation :
Process No.     Process Size    Block no.
 1              115             1
 2              500             2
 3              358             5
 4              200             3
 5              375             5

BestFit Algorithm

Best Fit Allocation :
Process No.     Process Size    Block no.
   1            115             6
   2            500             Not Allocated
   3            358             Not Allocated
   4            200             4
   5            375             Not Allocated

WorstFit Algorithm

Worst Fit Allocation :
Process No.     Process Size    Block no.
   1            115             1
   2            500             Not Allocated
   3            358             Not Allocated
   4            200             Not Allocated
   5            375             Not Allocated

(base) Aadhityas-MacBook-Air:5Jun2020 aadhitya$ ▯
```

Result :

We can see the memory allocation by the various methods.
We can analyse the three algorithms by the output above.

We can rank the three methods by efficiency as follows :

1. First Fit
2. Best Fit
3. Worst Fit

Though this order differs from what is expected, we can say that in most cases the best fit is a better choice when compared with the other two algorithms.