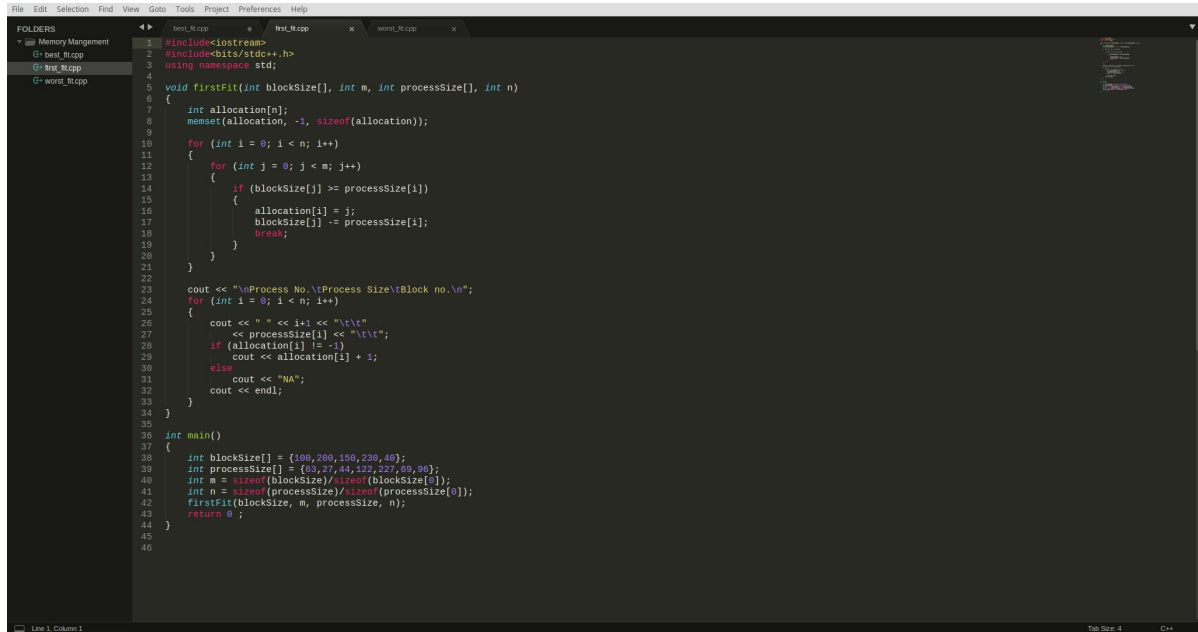


LAB MANUAL

JATIN KARTHIK TRIPATHY

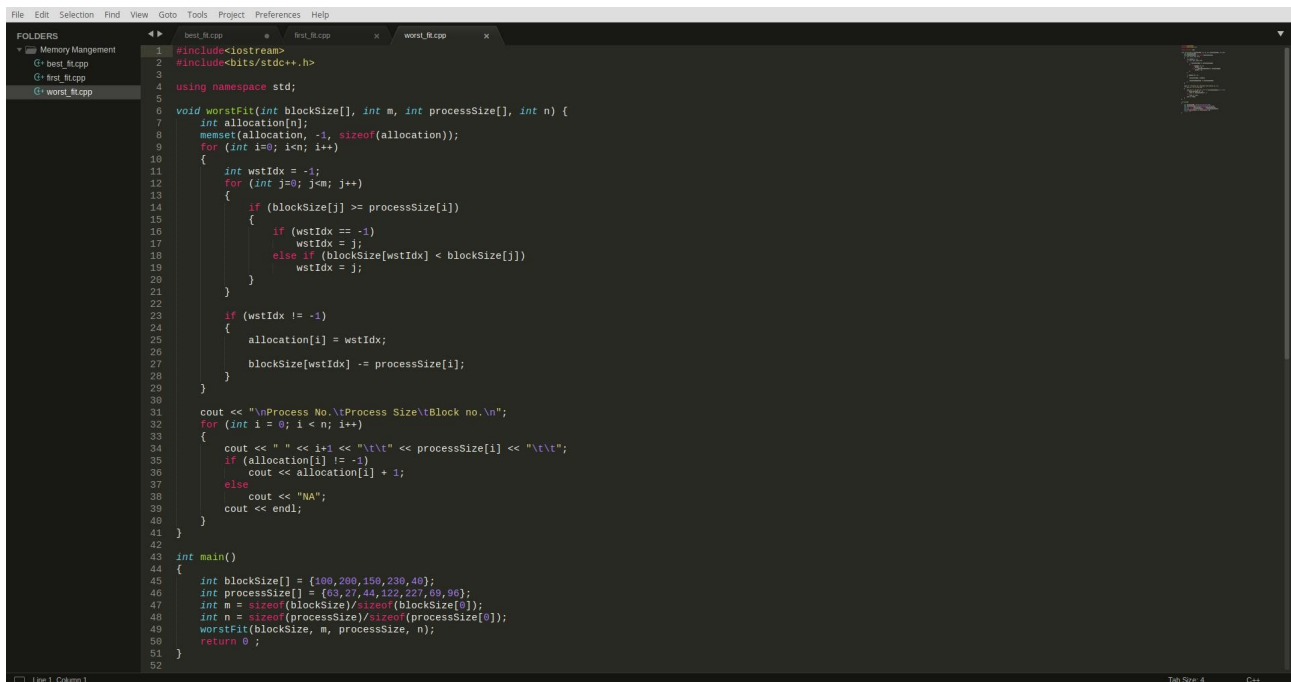
(17BCE7106)

1. First Fit



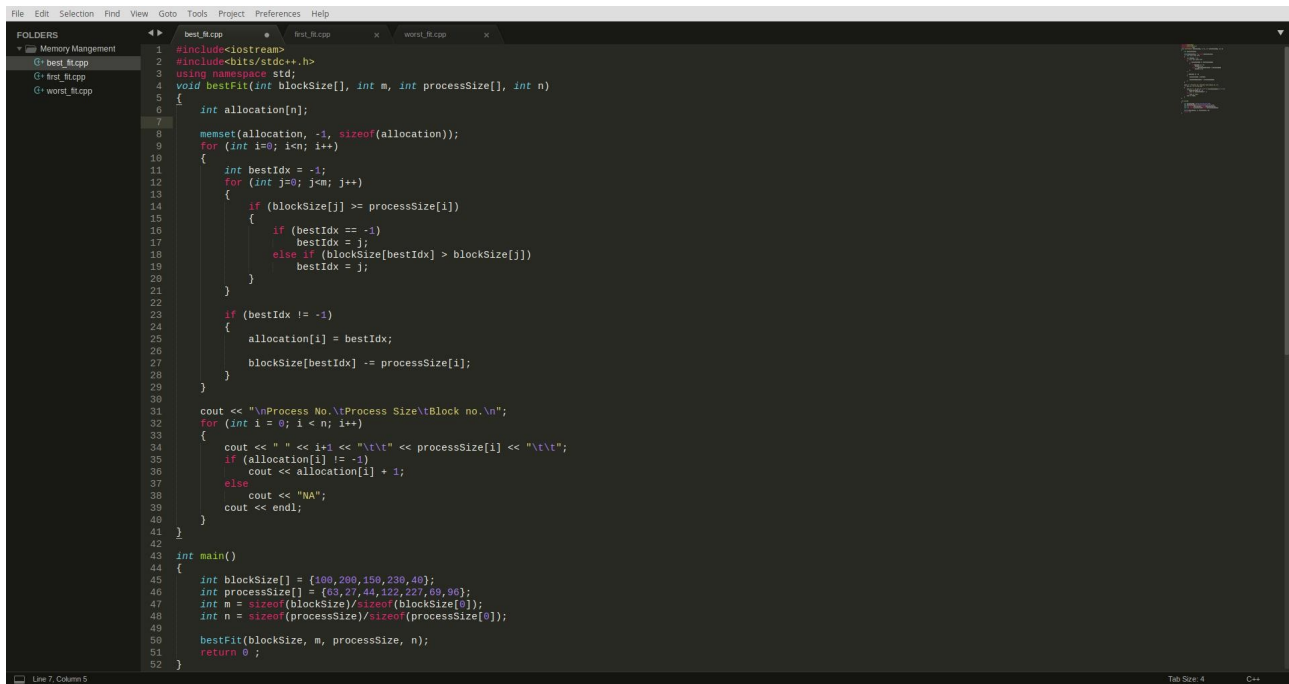
```
1 #include<iostream>
2 #include<bits/stdc++.h>
3 using namespace std;
4
5 void firstFit(int blockSize[], int m, int processSize[], int n)
6 {
7     int allocation[n];
8     memset(allocation, -1, sizeof(allocation));
9
10    for (int i = 0; i < n; i++)
11    {
12        for (int j = 0; j < m; j++)
13        {
14            if (blockSize[j] >= processSize[i])
15            {
16                allocation[i] = j;
17                blockSize[j] -= processSize[i];
18                break;
19            }
20        }
21    }
22
23    cout << "\nProcess No.\tProcess Size\tBlock no.\n";
24    for (int i = 0; i < n; i++)
25    {
26        cout << " " << i+1 << "\t\t";
27        cout << processSize[i] << "\t\t";
28        if (allocation[i] != -1)
29            cout << allocation[i] + 1;
30        else
31            cout << "NA";
32        cout << endl;
33    }
34 }
35
36 int main()
37 {
38     int blockSize[] = {100,200,150,230,40};
39     int processSize[] = {63,27,44,122,227,69,96};
40     int m = sizeof(blockSize)/sizeof(blockSize[0]);
41     int n = sizeof(processSize)/sizeof(processSize[0]);
42     firstFit(blockSize, m, processSize, n);
43     return 0 ;
44 }
45
46
```

2. Worst Fit



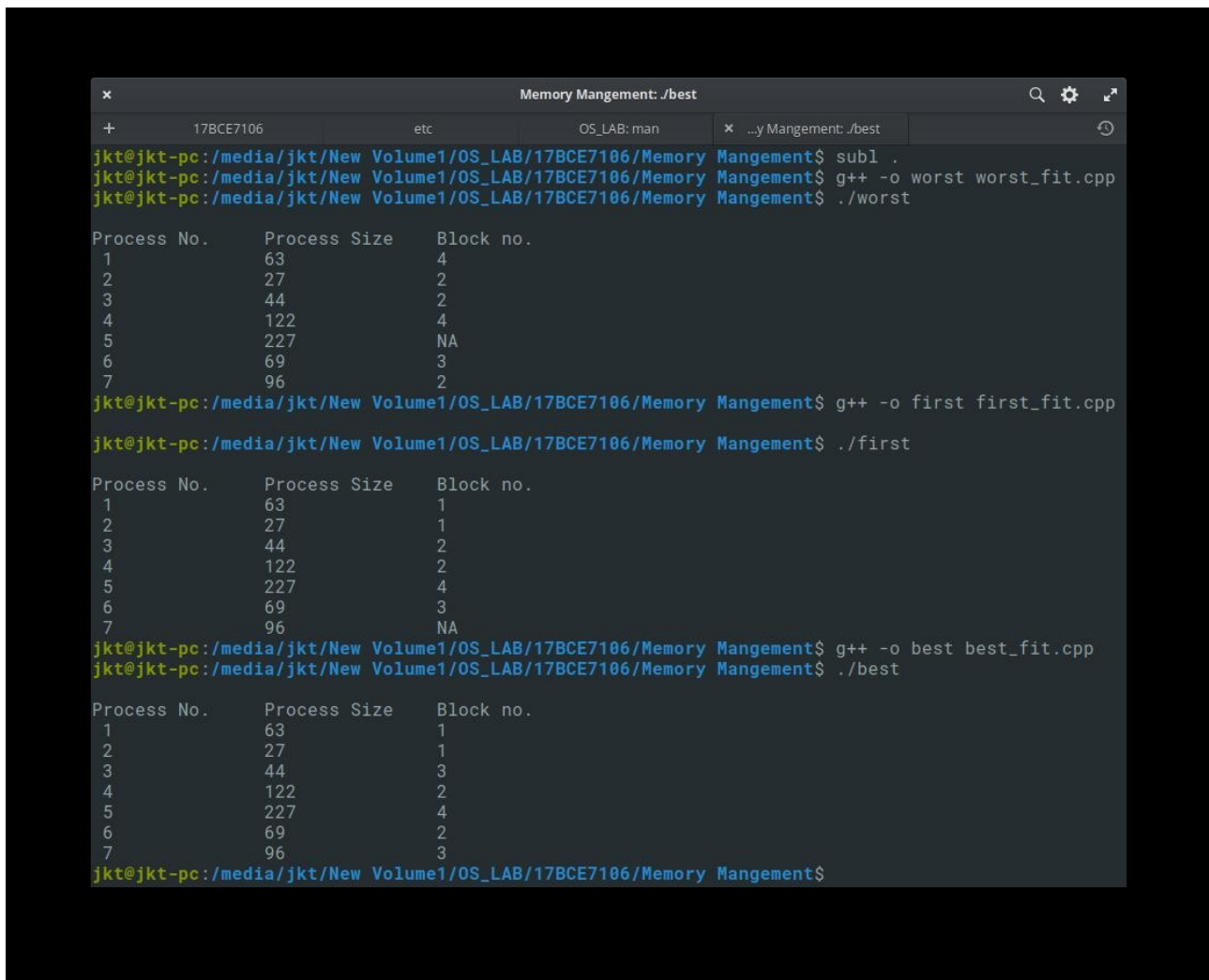
```
1 #include<iostream>
2 #include<bits/stdc++.h>
3 using namespace std;
4
5 void worstFit(int blockSize[], int m, int processSize[], int n) {
6     int allocation[n];
7     memset(allocation, -1, sizeof(allocation));
8     for (int i=0; i<n; i++)
9     {
10        int wstIdx = -1;
11        for (int j=0; j<m; j++)
12        {
13            if (blockSize[j] >= processSize[i])
14            {
15                if (wstIdx == -1)
16                    wstIdx = j;
17                else if (blockSize[wstIdx] < blockSize[j])
18                    wstIdx = j;
19            }
20        }
21
22        if (wstIdx != -1)
23        {
24            allocation[i] = wstIdx;
25            blockSize[wstIdx] -= processSize[i];
26        }
27    }
28
29    cout << "\nProcess No.\tProcess Size\tBlock no.\n";
30    for (int i = 0; i < n; i++)
31    {
32        cout << " " << i+1 << "\t\t" << processSize[i] << "\t\t";
33        if (allocation[i] != -1)
34            cout << allocation[i] + 1;
35        else
36            cout << "NA";
37        cout << endl;
38    }
39 }
40
41 int main()
42 {
43     int blockSize[] = {100,200,150,230,40};
44     int processSize[] = {63,27,44,122,227,69,96};
45     int m = sizeof(blockSize)/sizeof(blockSize[0]);
46     int n = sizeof(processSize)/sizeof(processSize[0]);
47     worstFit(blockSize, m, processSize, n);
48     return 0 ;
49 }
50
51
52
```

3. Best Fit



```
1 #include<iostream>
2 #include<bits/stdc++.h>
3 using namespace std;
4 void bestFit(int blockSize[], int m, int processSize[], int n)
5 {
6     int allocation[n];
7
8     memset(allocation, -1, sizeof(allocation));
9     for (int i=0; i<n; i++)
10     {
11         int bestIdx = -1;
12         for (int j=0; j<m; j++)
13         {
14             if (blockSize[j] >= processSize[i])
15             {
16                 if (bestIdx == -1)
17                     bestIdx = j;
18                 else if (blockSize[bestIdx] > blockSize[j])
19                     bestIdx = j;
20             }
21         }
22         if (bestIdx != -1)
23         {
24             allocation[i] = bestIdx;
25             blockSize[bestIdx] -= processSize[i];
26         }
27     }
28
29     cout << "\nProcess No.\tProcess Size\tBlock no.\n";
30     for (int i = 0; i < n; i++)
31     {
32         cout << " " << i+1 << "\t\t" << processSize[i] << "\t\t";
33         if (allocation[i] != -1)
34             cout << allocation[i] + 1;
35         else
36             cout << "NA";
37         cout << endl;
38     }
39 }
40
41 int main()
42 {
43     int blockSize[] = {100,200,150,230,40};
44     int processSize[] = {63,27,44,122,227,69,96};
45     int m = sizeof(blockSize)/sizeof(blockSize[0]);
46     int n = sizeof(processSize)/sizeof(processSize[0]);
47     bestFit(blockSize, m, processSize, n);
48     return 0 ;
49 }
```

Output



```
jkt@jkt-pc:/media/jkt/New Volume1/OS_LAB/17BCE7106/Memory Mangement$ subl .
jkt@jkt-pc:/media/jkt/New Volume1/OS_LAB/17BCE7106/Memory Mangement$ g++ -o worst worst_fit.cpp
jkt@jkt-pc:/media/jkt/New Volume1/OS_LAB/17BCE7106/Memory Mangement$ ./worst

Process No.      Process Size      Block no.
1                63               4
2                27               2
3                44               2
4                122              4
5                227              NA
6                69               3
7                96               2

jkt@jkt-pc:/media/jkt/New Volume1/OS_LAB/17BCE7106/Memory Mangement$ g++ -o first first_fit.cpp
jkt@jkt-pc:/media/jkt/New Volume1/OS_LAB/17BCE7106/Memory Mangement$ ./first

Process No.      Process Size      Block no.
1                63               1
2                27               1
3                44               2
4                122              2
5                227              4
6                69               3
7                96              NA

jkt@jkt-pc:/media/jkt/New Volume1/OS_LAB/17BCE7106/Memory Mangement$ g++ -o best best_fit.cpp
jkt@jkt-pc:/media/jkt/New Volume1/OS_LAB/17BCE7106/Memory Mangement$ ./best

Process No.      Process Size      Block no.
1                63               1
2                27               1
3                44               3
4                122              2
5                227              4
6                69               2
7                96               3

jkt@jkt-pc:/media/jkt/New Volume1/OS_LAB/17BCE7106/Memory Mangement$
```