Operating System Lab

CEN-493

Program - 11

Code :-

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

typedef long long ll;

struct memoryBlocks
{
    bool isAllocated;
    int blockSize;
    int processSize;
    int internalFrag;
    string processName;
};

void printLines()
{
    for (int i = 0; i < 110; i++)
    {
```

```
        cout << "_";
    }
    cout << "\n";
}

void Display(vector<memoryBlocks> &memBlocks, int noOfBlocks, int
internalFrag, int externalFrag, vector<pair<int, string>>
&leftProcess)
{
    cout << "Best Fit Memory Allocation Table \n";
    cout << "-------------------------------------------------
--------------------------\n";
    cout << "| Block No\t"
         << "Size Of Block\t"
         << "Proces Allocated\t"
         << "Internal Fragmentation  |\n";
    cout << "-------------------------------------------------
--------------------------\n";
    for (int bindx = 0; bindx < noOfBlocks; bindx++)
    {
        if (memBlocks[bindx].isAllocated == false)
            cout << "|  " << bindx + 1 << "\t\t\t" <<
memBlocks[bindx].blockSize << "\t\t"
                 << "  ---  "
                 << "\t\t\t"
                 << "--"
                 << "\t\t|\n";

        else
            cout << "|  " << bindx + 1 << "\t\t\t" <<
memBlocks[bindx].blockSize << "\t\t"
                 << memBlocks[bindx].processSize << "[" <<
memBlocks[bindx].processName << "]"
                 << "\t\t\t" << memBlocks[bindx].internalFrag <<
"\t\t|\n";
    }
    cout << "-------------------------------------------------
--------------------------\n";

    cout << "\n";
    printLines();
    printLines();
    if (!leftProcess.empty())
    {
        cout << "Process Whom Memory Is Not Allocated : \n";
```

```
        for (int lindx = 0; lindx < leftProcess.size(); lindx++)
        {
            cout << leftProcess[lindx].second << " " <<
leftProcess[lindx].first << "\n";
        }
    }

    printLines();
    cout << "\n\n";
    printLines();
    cout << "Total Internal Fragmentation = " << internalFrag <<
"\n";
    cout << "Total External Fragmentation = " << externalFrag <<
"\n";
    printLines();
}

void Best_Fit(vector<memoryBlocks> &memBlocks, int noOfBlocks,
vector<pair<int, string>> &processSizes, int noOfProcess)
{
    vector<pair<int, string>> leftProcess;
    for (int pindx = 0; pindx < noOfProcess; pindx++)
    {
        bool isProcessMemAllocated = false;
        int emptyBlock = 0, bestBlockSize = 1e9;
        for (int bindx = 0; bindx < noOfBlocks; bindx++)
        {
            if (memBlocks[bindx].isAllocated == true ||
memBlocks[bindx].blockSize < processSizes[pindx].first)
                continue;

            isProcessMemAllocated = true;
            if (bestBlockSize > memBlocks[bindx].blockSize)
            {
                emptyBlock = bindx;
                bestBlockSize = memBlocks[bindx].blockSize;
            }
        }
        if (isProcessMemAllocated == false)
        {
            leftProcess.push_back(processSizes[pindx]);
        }
        else
        {
            memBlocks[emptyBlock].isAllocated = true;
```

```
            memBlocks[emptyBlock].processName =
processSizes[pindx].second;
            memBlocks[emptyBlock].processSize =
processSizes[pindx].first;
            memBlocks[emptyBlock].internalFrag =
memBlocks[emptyBlock].blockSize - processSizes[pindx].first;
        }
    }
    int externalFrag = 0, internalFrag = 0;
    if (leftProcess.empty() == false)
    {
        for (int bindx = 0; bindx < noOfBlocks; bindx++)
        {
            if (memBlocks[bindx].isAllocated == true)
                continue;
            externalFrag += memBlocks[bindx].blockSize;
        }
    }
    for (int bindx = 0; bindx < noOfBlocks; bindx++)
    {
        internalFrag += memBlocks[bindx].internalFrag;
    }
    Display(memBlocks, noOfBlocks, internalFrag, externalFrag,
leftProcess);
}

int main()
{
    system("cls");

    printLines();
    cout << "Vicky Gupta 20BCS070\n";
    cout << "Best Fit Memory Allocation Algorithm\n";
    printLines();
    printLines();

    int noOfBlocks;
    cout << "Enter The No Of Blocks Of Memory : ";
    cin >> noOfBlocks;

    printLines();
    int noOfProcess;
    cout << "Enter The No Of Process : ";
    cin >> noOfProcess;
```

```
    printLines();
    vector<memoryBlocks> memBlocks(noOfBlocks);
    cout << "Enter The Sizes Of Blocks : ";
    for (int i = 0; i < noOfBlocks; i++)
    {
        cin >> memBlocks[i].blockSize;
        memBlocks[i].isAllocated = false;
        memBlocks[i].processSize = 0;
        memBlocks[i].processName = "";
        memBlocks[i].internalFrag = 0;
    }

    printLines();
    vector<pair<int, string>> processSizes(noOfProcess);
    cout << "Enter The Sizes Of Process : ";
    for (int i = 0; i < noOfProcess; i++)
    {
        cin >> processSizes[i].first;
        processSizes[i].second = "P";
        processSizes[i].second += to_string(i + 1);
    }
    printLines();

    cout << "\n\n";
    printLines();
    printLines();
    Best_Fit(memBlocks, noOfBlocks, processSizes, noOfProcess);
    return 0;
}
```

Output :-

```
--------------------------------------------------------------------------------
Vicky Gupta 20BCS070
Best Fit Memory Allocation Algorithm
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Enter The No Of Blocks Of Memory : 5
--------------------------------------------------------------------------------
Enter The No Of Process : 4
--------------------------------------------------------------------------------
Enter The Sizes Of Blocks : 100 500 200 300 600
--------------------------------------------------------------------------------
Enter The Sizes Of Process : 212 417 112 426
--------------------------------------------------------------------------------



--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Best Fit Memory Allocation Table
-------------------------------------------------------------------------
| Block No      Size Of Block   Proces Allocated       Internal Fragmentation  |
-------------------------------------------------------------------------
|  1                  100             ---                  --             |
|  2                  500           417[P2]                83             |
|  3                  200           112[P3]                88             |
|  4                  300           212[P1]                88             |
|  5                  600           426[P4]                174            |
-------------------------------------------------------------------------



--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------



--------------------------------------------------------------------------------
Total Internal Fragmentation = 433
Total External Fragmentation = 0
--------------------------------------------------------------------------------
```