EXPERIMENT-6.1

Aim:

Wite a program for memory management using First Fit algorithm.

Theory:

Code:

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

void firstFit(int block[], int m, int process[], int n) {
    int allocation[n];

    for (int i = 0; i < n; i++) {
        allocation[i] = -1;
    }

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            if (block[j] >= process[i]) {
                allocation[i] = j;
                block[j] -= process[i];
                break;
            }
        }
    }

    cout << "First Fit Allocation:\n";
    for (int i = 0; i < n; i++) {
        if (allocation[i] != -1) {
            cout << "Process " << i + 1 << " allocated to Block " <<
allocation[i] + 1 << endl;
        } else {
            cout << "Process " << i + 1 << " cannot be allocated\n";
        }
    }
}

int main() {
    int m, n;

    cout << "Enter the number of memory blocks: ";
    cin >> m;
    int block[m];

    cout << "Enter the sizes of memory blocks:\n";
    for (int i = 0; i < m; i++) {
        cin >> block[i];
    }

    cout << "Enter the number of processes: ";
    cin >> n;
    int process[n];
    cout << "Enter the sizes of processes:\n";
    for (int i = 0; i < n; i++) {
        cin >> process[i];
    }

    firstFit(block, m, process, n);

    return 0;
}
```

Output:

```
Enter the number of memory blocks: 6
Enter the sizes of memory blocks:
200 400 600 500 300 250
Enter the number of processes: 4
Enter the sizes of processes:
357 210 468 491
First Fit Allocation:
Process 1 allocated to Block 2
Process 2 allocated to Block 3
Process 3 allocated to Block 4
Process 4 cannot be allocated
```

EXPERIMENT-6.2

Aim:

Wite a program for memory management using Best Fit algorithm.

Theory:

Code:

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

void bestFit(int block[], int m, int process[], int n) {
    int allocation[n];

    for (int i = 0; i < n; i++) {
        allocation[i] = -1;
    }

    for (int i = 0; i < n; i++) {
        int bestFitIdx = -1;
        for (int j = 0; j < m; j++) {
            if (block[j] >= process[i]) {
                if (bestFitIdx == -1 || block[j] < block[bestFitIdx]) {
                    bestFitIdx = j;
                }
            }
        }

        if (bestFitIdx != -1) {
            allocation[i] = bestFitIdx;
            block[bestFitIdx] -= process[i];
        }
    }

    cout << "Best Fit Allocation:\n";
    for (int i = 0; i < n; i++) {
        if (allocation[i] != -1) {
            cout << "Process " << i + 1 << " allocated to Block " <<
allocation[i] + 1 << endl;
        } else {
            cout << "Process " << i + 1 << " cannot be allocated\n";
        }
    }
}

int main() {
    int m, n;

    cout << "Enter the number of memory blocks: ";
    cin >> m;
    int block[m];

    cout << "Enter the sizes of memory blocks:\n";
    for (int i = 0; i < m; i++) {
        cin >> block[i];
    }

    cout << "Enter the number of processes: ";
    cin >> n;
    int process[n];

    cout << "Enter the sizes of processes:\n";
    for (int i = 0; i < n; i++) {
        cin >> process[i];
    }

    bestFit(block, m, process, n);

    return 0;
}
```

Output:

```
Enter the number of memory blocks: 6
Enter the sizes of memory blocks:
200 400 600 500 300 250
Enter the number of processes: 4
Enter the sizes of processes:
357 210 468 491
Best Fit Allocation:
Process 1 allocated to Block 2
Process 2 allocated to Block 6
Process 3 allocated to Block 4
Process 4 allocated to Block 3
```

<u>EXPERIMENT-6.3</u>

Aim:

Wite a program for memory management using Worst Fit algorithm.

Theory:

Code:

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

void worstFit(int block[], int m, int process[], int n) {
    int allocation[n];

    for (int i = 0; i < n; i++) {
        allocation[i] = -1;
    }

    for (int i = 0; i < n; i++) {
        int worstFitIdx = -1;
        for (int j = 0; j < m; j++) {
            if (block[j] >= process[i]) {
                if (worstFitIdx == -1 || block[j] > block[worstFitIdx]) {
                    worstFitIdx = j;
                }
            }
        }

        if (worstFitIdx != -1) {
            allocation[i] = worstFitIdx;
            block[worstFitIdx] -= process[i];
        }
    }

    cout << "Worst Fit Allocation:\n";
    for (int i = 0; i < n; i++) {
        if (allocation[i] != -1) {
            cout << "Process " << i + 1 << " allocated to Block " <<
allocation[i] + 1 << endl;
        } else {
            cout << "Process " << i + 1 << " cannot be allocated\n";
        }
    }
}

int main() {
    int m, n;

    cout << "Enter the number of memory blocks: ";
    cin >> m;
    int block[m];

    cout << "Enter the sizes of memory blocks:\n";
    for (int i = 0; i < m; i++) {
        cin >> block[i];
    }

    cout << "Enter the number of processes: ";
    cin >> n;
    int process[n];

    cout << "Enter the sizes of processes:\n";
    for (int i = 0; i < n; i++) {
        cin >> process[i];
    }

    worstFit(block, m, process, n);

    return 0;
}
```

Output:

```
Enter the number of memory blocks: 6
Enter the sizes of memory blocks:
200 400 600 500 300 250
Enter the number of processes: 4
Enter the sizes of processes:
357 210 468 491
Worst Fit Allocation:
Process 1 allocated to Block 3
Process 2 allocated to Block 4
Process 3 cannot be allocated
Process 4 cannot be allocated
```