**Practical 4:** Write a program to simulate Memory placement strategies – Best Fit, Worst Fit, First Fit.

**FIRST-FIT:**

#include <iostream>

#include <vector>

#define MAX 25

int main() {

int frag[MAX], b[MAX], f[MAX], i, j, nb, nf, temp;

std::vector<int> bf(MAX, 0), ff(MAX, 0);

std::cout<<"onkar lonsane";

std::cout<<"\nRoll no 28";

std::cout << "\n\tMemory Management Scheme - First Fit";

std::cout << "\nEnter the number of blocks: ";

std::cin >> nb;

std::cout << "Enter the number of files: ";

std::cin >> nf;

std::cout << "\nEnter the size of the blocks:-\n";

for(i = 1; i <= nb; i++) {

std::cout << "Block " << i << ": ";

std::cin >> b[i];

}

std::cout << "Enter the size of the files :-\n";

for(i = 1; i <= nf; i++) {

std::cout << "File " << i << ": ";

std::cin >> f[i];

}

for(i = 1; i <= nf; i++) {

for(j = 1; j <= nb; j++) {

if(bf[j] != 1) {

temp = b[j] - f[i];

if(temp >= 0) {

ff[i] = j;

break;

}

}

}

frag[i] = temp;

bf[ff[i]] = 1;

}

std::cout << "\nFile\_no:\tFile\_size :\tBlock\_no:\tBlock\_size:\tFragement";

for(i = 1; i <= nf; i++) {

std::cout << "\n" << i << "\t\t" << f[i] << "\t\t" << ff[i] << "\t\t" << b[ff[i]] << "\t\t" << frag[i];

}

return 0;

}

**Output:**

/tmp/RrfGjlz8gU.o

Onkar lonsane

Roll no 28

Memory Management Scheme - First Fit

Enter the number of blocks: 4

Enter the number of files: 4

Enter the size of the blocks: -

Block 1: 2

Block 2: 3

Block 3: 4

Block 4: 5

Enter the size of the files:-

File 1: 6

File 2: 7

File 3: 8

File 4: 9

File\_no: File\_size: Block\_no: Block\_size: Fragement

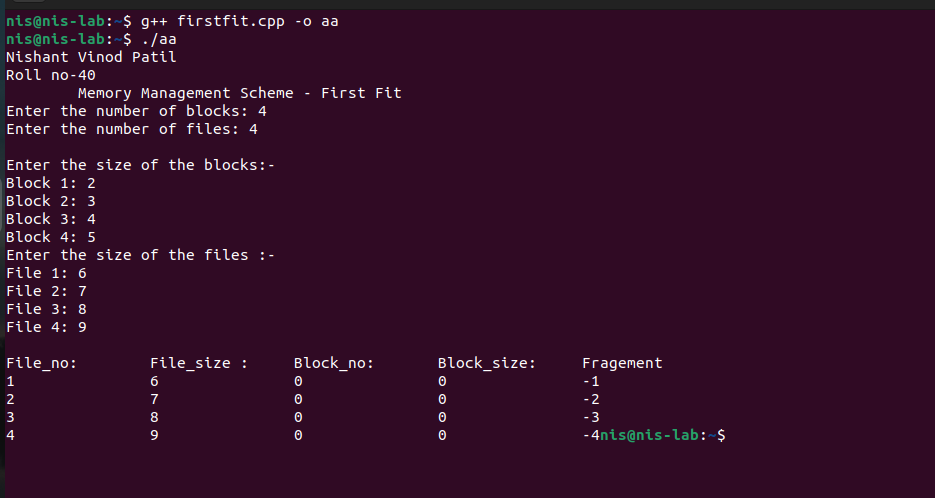
1 6 0 1731195016 -1

2 7 0 1731195016 -2

3 8 0 1731195016 -3

4 9 0 1731195016 -4

=== Code Execution Successful ===



**BEST-FIT:**

#include <iostream>

#include <vector>

#include <limits> // for std::numeric\_limits

#define MAX 25

int main() {

std::vector<int> frag(MAX, 0), b(MAX, 0), f(MAX, 0);

std::vector<int> bf(MAX, 0), ff(MAX, 0);

std::cout<<"onkar lonsane";

std::cout<<"\nRoll no 28";

int nb, nf, temp, lowest = std::numeric\_limits<int>::max();

// Input number of blocks and files

std::cout << "\nEnter the number of blocks: ";

std::cin >> nb;

std::cout << "Enter the number of files: ";

std::cin >> nf;

// Input block sizes

std::cout << "\nEnter the size of the blocks:-\n";

for (int i = 1; i <= nb; ++i) {

std::cout << "Block " << i << ": ";

std::cin >> b[i];

}

// Input file sizes

std::cout << "Enter the size of the files :-\n";

for (int i = 1; i <= nf; ++i) {

std::cout << "File " << i << ": ";

std::cin >> f[i];

}

// First Fit Memory Allocation with Best Fit Calculation

for (int i = 1; i <= nf; ++i) {

lowest = std::numeric\_limits<int>::max();

for (int j = 1; j <= nb; ++j) {

if (bf[j] != 1) {

temp = b[j] - f[i];

if (temp >= 0 && lowest > temp) {

ff[i] = j;

lowest = temp;

}

}

}

frag[i] = lowest;

bf[ff[i]] = 1;

}

// Output the results

std::cout << "\nFile No\tFile Size \tBlock No\tBlock Size\tFragment";

for (int i = 1; i <= nf && ff[i] != 0; ++i) {

std::cout << "\n" << i << "\t\t" << f[i] << "\t\t" << ff[i] << "\t\t" << b[ff[i]] << "\t\t" << frag[i];

}

return 0;

}

**Output:**

/tmp/EAvCuylymw.o

Onkar lonsane

Roll no 28

Enter the number of blocks: 4

Enter the number of files: 5

Enter the size of the blocks:-

Block 1: 2

Block 2: 5

Block 3: 7

Block 4: 9

Enter the size of the files :-

File 1: 1

File 2: 3

File 3: 4

File 4: 6

File 5: 8

File No File Size Block No Block Size Fragment

1 1 1 2 1

2 3 2 5 2

3 4 3 7 3

4 6 4 9 3

=== Code Execution Successful ===

**Worstfit:**

#include <iostream>

#include <vector>

#include <limits> // For std::numeric\_limits

#define MAX 25

int main() {

std::vector<int> frag(MAX, 0), b(MAX, 0), f(MAX, 0);

std::vector<int> bf(MAX, 0), ff(MAX, 0);

std::cout<<"Onkar lonsane";

std::cout<<"\nRoll no 28";

int nb, nf, temp, highest = 0;

// Input number of blocks and files

std::cout << "\n\tMemory Management Scheme - Worst Fit";

std::cout << "\nEnter the number of blocks: ";

std::cin >> nb;

std::cout << "Enter the number of files: ";

std::cin >> nf;

// Input block sizes

std::cout << "\nEnter the size of the blocks:-\n";

for (int i = 1; i <= nb; ++i) {

std::cout << "Block " << i << ": ";

std::cin >> b[i];

}

// Input file sizes

std::cout << "Enter the size of the files :-\n";

for (int i = 1; i <= nf; ++i) {

std::cout << "File " << i << ": ";

std::cin >> f[i];

}

// Worst Fit Memory Allocation

for (int i = 1; i <= nf; ++i) {

highest = 0; // Reset highest for each file

for (int j = 1; j <= nb; ++j) {

if (bf[j] != 1) { // If the block is not allocated

temp = b[j] - f[i];

if (temp >= 0 && highest < temp) {

ff[i] = j;

highest = temp;

}

}

}

frag[i] = highest;

if (ff[i] != 0) {

bf[ff[i]] = 1;

}

}

// Output results

std::cout << "\nFile No\tFile Size \tBlock No\tBlock Size\tFragment";

for (int i = 1; i <= nf; ++i) {

if (ff[i] != 0) {

std::cout << "\n" << i << "\t\t" << f[i] << "\t\t" << ff[i] << "\t\t" << b[ff[i]] << "\t\t" << frag[i];

}

}

return 0;

}

**Output:**

/tmp/QdKvWjeyLY.o

Onkar lonsane

Roll no 28

Memory Management Scheme - Worst Fit

Enter the number of blocks: 3

Enter the number of files: 2

Enter the size of the blocks:-

Block 1: 3

Block 2: 8

Block 3: 9

Enter the size of the files :-

File 1: 7

File 2: 5

File No File Size Block No Block Size Fragment

1 7 3 9 2

2 5 2 8 3

=== Code Execution Successful ===