

**19BIT0292**

**Bhaumik Tandan**

**ASSESSMENT-5**

OPERATING SYSTEM

Laboratory

ITE2002

## Q1. Write a program to implement the first fit, best fit, and worst fit algorithm for memory allocation.

**CODE**

#include <iostream>

#include <iomanip>

using namespace std;

typedef struct

{

int n;

int v;

} block;

main()

{

cout << "1)First Fit\n2)Best Fit\n3)Worst Fit\nEnter your choice: ";

int o;

cin >> o;

cout << "Enter the number of processes: ";

int n;

cin >> n;

int \*p = (int \*)malloc(sizeof(int) \* n);

for (int i = 0; i < n; i++)

{

cout << "Enter the memory size for process " << i + 1 << ": ";

cin >> p[i];

}

cout << "Enter the number of memory blocks: ";

int bn;

cin >> bn;

block \*b = (block \*)malloc(sizeof(block) \* bn);

for (int i = 0; i < bn; i++)

{

cout << "Enter the memory size for block " << i + 1 << ": ";

cin >> b[i].v;

b[i].n = i + 1;

}

for (int i = 0; i < bn - 1; i++)

for (int j = 0; j < bn - i - 1; j++)

if (o == 2 && b[j].v > b[j + 1].v)

{

block t = b[j];

b[j] = b[j + 1];

b[j + 1] = t;

}

else if (o == 3 && b[j].v < b[j + 1].v)

{

block t = b[j];

b[j] = b[j + 1];

b[j + 1] = t;

}

cout << left << setw(25) << setfill(' ') << "Process Number";

cout << left << setw(25) << setfill(' ') << "Process Size";

cout << left << setw(25) << setfill(' ') << "Block Number";

for (int i = 0; i < n; i++)

{

cout << endl

<< left << setw(25) << setfill(' ') << i + 1;

cout << left << setw(25) << setfill(' ') << p[i];

int j;

for (j = 0; j < bn; j++)

if (p[i] <= b[j].v)

{

cout << left << setw(25) << setfill(' ') << b[j].n;

b[j].v -= p[i];

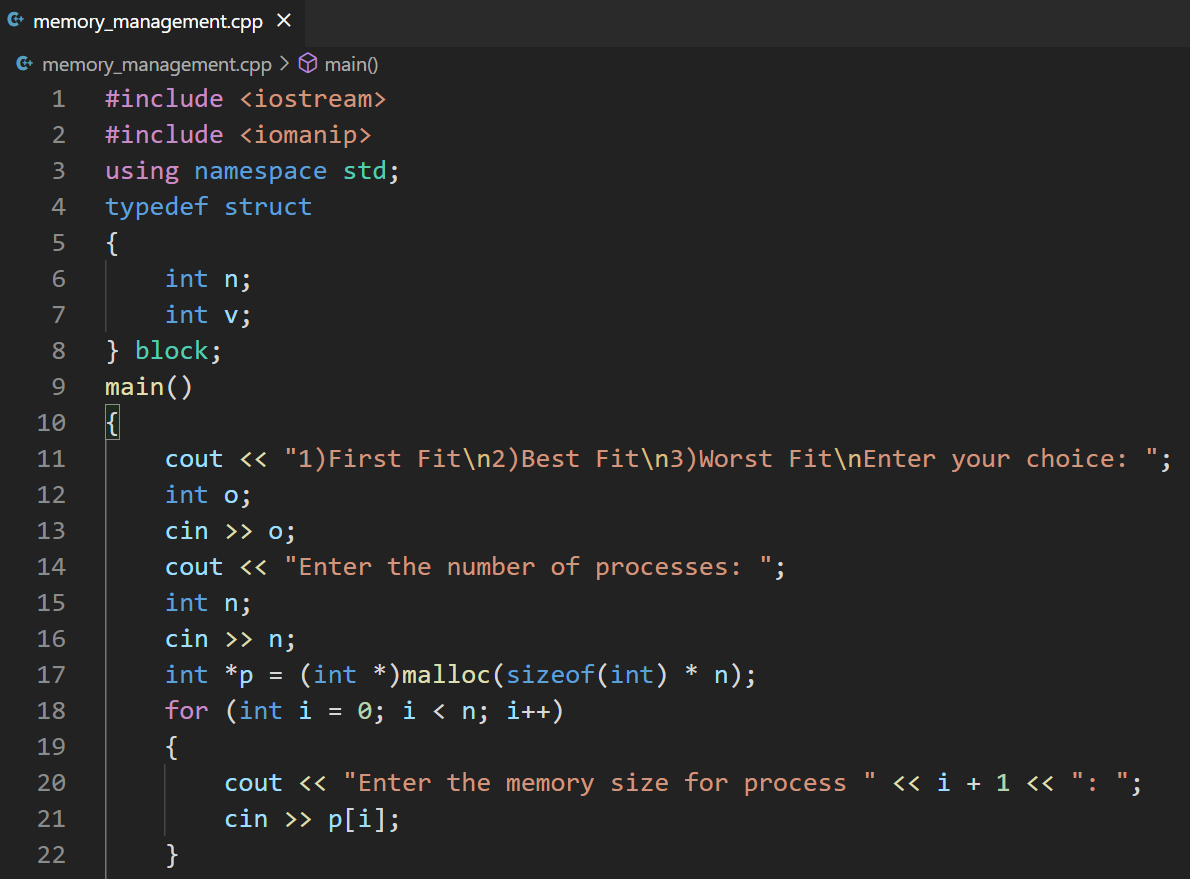
break;

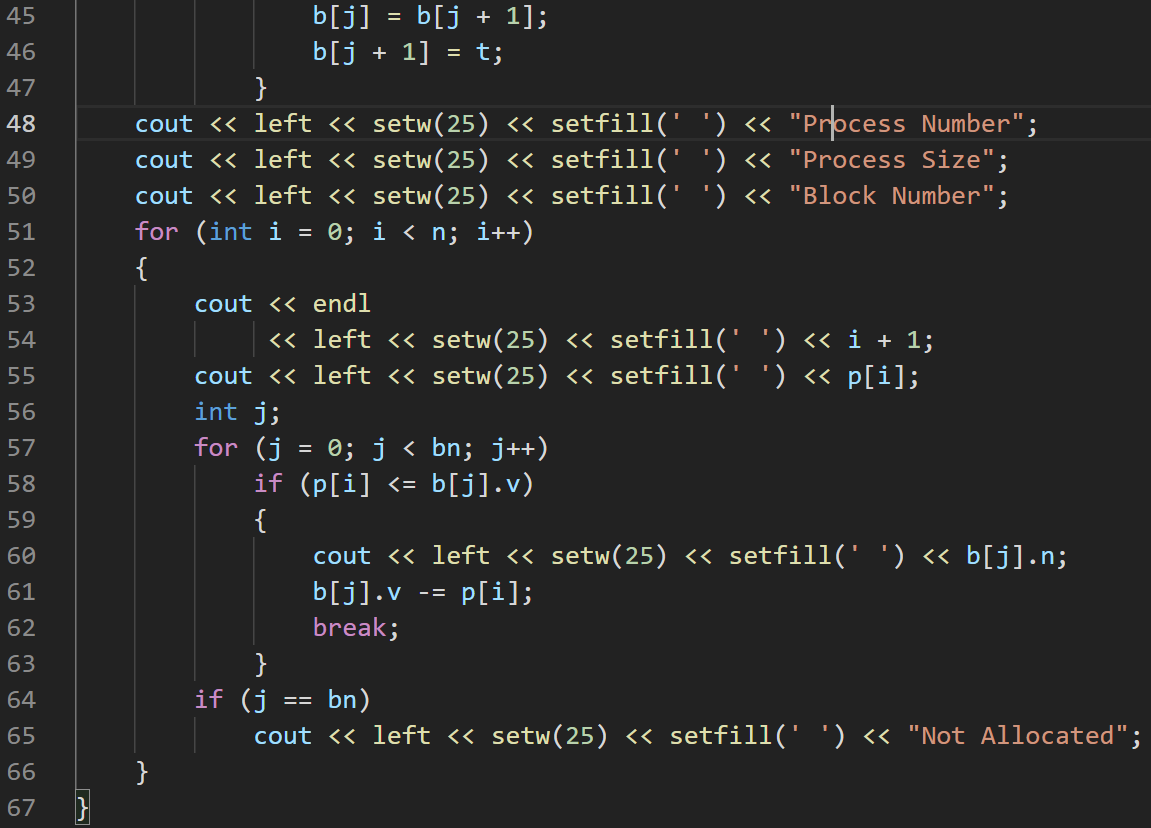
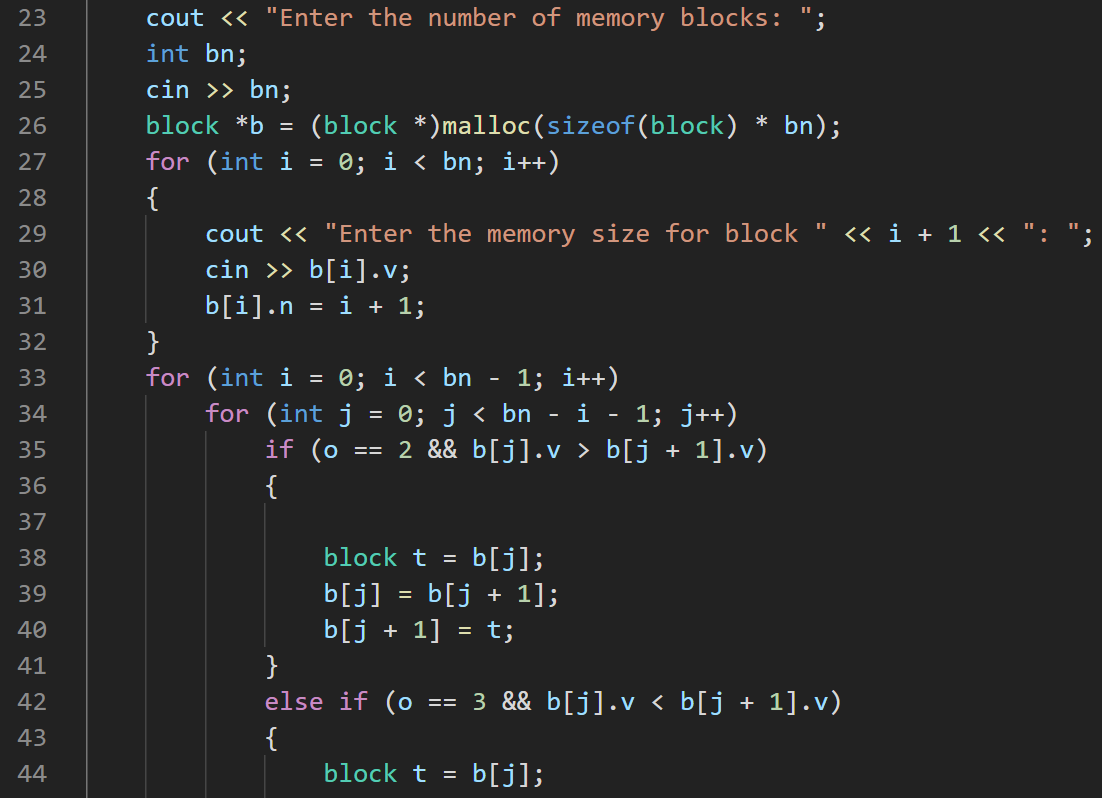
}

if (j == bn)

cout << left << setw(25) << setfill(' ') << "Not Allocated";

}

}



# OUTPUT

# FIRST FIT

## 

# **BEST FIT**

## 

# **WORST FIT**

## 

## **Q2.** Write a program to implement the page replacement algorithms.

## **a. FIFO**

## **b. LRU**

## **c. OPTIMAL**

**CODE**

#include <iostream>

#include <iomanip>

using namespace std;

main()

{

cout << "1)FIFO\n2)LRU\n3)OPTIMAL\nEnter your choice : ";

int o;

cin >> o;

cout << "Enter the number of elements in page reference string: ";

int n;

cin >> n;

int \*p = (int \*)malloc(sizeof(int) \* n);

cout << "Enter the page reference string: ";

for (int i = 0; i < n; i++)

cin >> p[i];

cout << "Enter the number of page frames: ";

int fn;

cin >> fn;

int \*f = (int \*)malloc(sizeof(int) \* fn), \*q;

for (int i = 0; i < fn; i++)

f[i] = -1;

cout << "\n\n"

<< left << setw(25) << setfill(' ') << "Action";

cout << left << setw(20) << setfill(' ') << "Before";

cout << left << setw(20) << setfill(' ') << "After";

int ph = 0, pf = 0, re = 0, e = 0;

for (int i = 0; i < n; i++)

{

int fl = 0;

for (int j = 0; j < fn; j++)

if (f[j] == p[i])

{

fl = 1;

cout << endl

<< left << setw(25) << setfill(' ') << "Page Hit";

ph++;

break;

}

if (fl == 0)

{

cout << endl

<< left << setw(25) << setfill(' ') << "Page Fault";

for (int j = fn - 1; j >= 0; j--)

cout << f[j] << " ";

if (o == 1)

{

f[re] = p[i];

re = (re + 1) % fn;

}

else if (o == 2)

{

int t = 0, b;

for (int j = 0; j < fn; j++)

{

int k;

for (k = i - 1; k >= 0; k--)

if (f[j] == p[k])

break;

if (i - k > t)

{

t = i - k;

b = j;

}

}

f[b] = p[i];

}

else

{

if (e < fn)

f[e++] = p[i];

else

{

int t = 0, b;

for (int j = 0; j < fn; j++)

{

int k;

for (k = i + 1; k < n; k++)

if (f[j] == p[k])

break;

if (k - i > t)

{

t = k - i;

b = j;

}

}

f[b] = p[i];

}

}

cout << left << setw(15) << setfill(' ') << " ";

for (int j = fn - 1; j >= 0; j--)

cout << f[j] << " ";

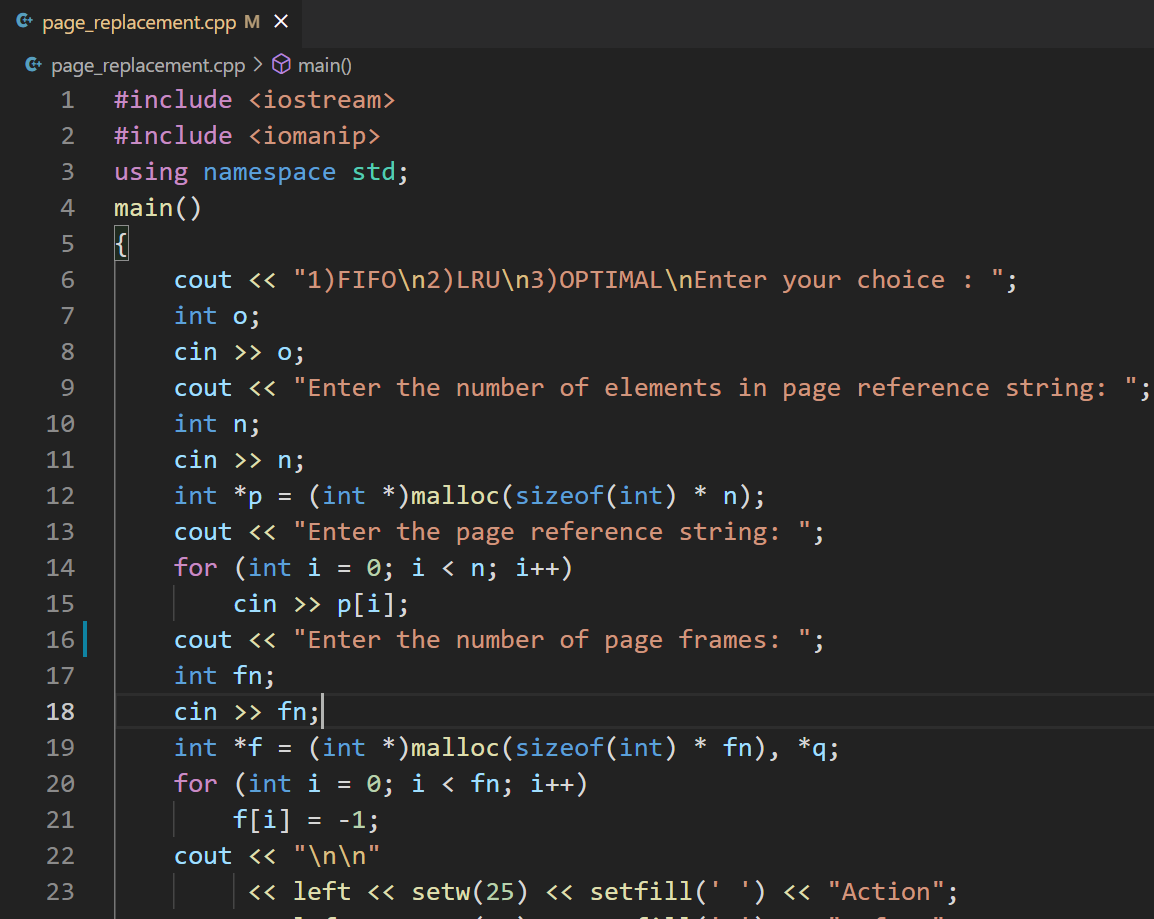
pf++;

}

}

cout << "\n\nTotal Page Faults: " << pf << "\nTotal Page Hits: " << ph;

}



## 

## 

# OUTPUT

# 

# FIFO

# LRU

# 

# OPTIMAL

# 

## **Q3. Implement the following algorithms to perform file allocation.**

## **a.** Sequential

## **b.** Linked

## **c.** Indexed

**CODE**

#include <iostream>

using namespace std;

main()

{

int \*h = (int \*)calloc(sizeof(int), 500);

cout << "1)Sequential\n2)Linked\n3)Indexed\nEnter your choice :";

int o;

cin >> o;

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

int n;

cin >> n;

int \*\*ind;

if (o == 3)

ind = (int \*\*)malloc(sizeof(int \*) \* 500);

int \*fi = (int \*)malloc(sizeof(int) \* n);

for (int i = 0; i < n; i++)

{

cout << "\nEnter the size of file " << i + 1 << ": ";

int s;

cin >> s;

if (o == 1)

while (true)

{

int j, a = rand() % (501 - s), f = 1;

for (j = a; j < a + s; j++)

if (h[j] != 0)

{

f = 0;

break;

}

if (f == 0)

continue;

fi[i] = a;

for (j = a; j < a + s; j++)

h[j] = i + 1;

break;

}

else if (o == 2)

{

int \*p = fi + i;

for (int j = 0; j <= s; j++)

{

int b;

while (true)

{

b = rand() % 500;

if (h[b] == 0)

break;

}

\*p = b;

p = h + b;

}

\*p = -1;

}

else

{

int in;

while (true)

{

in = rand() % 500;

if (h[in] == 0)

break;

}

h[in] = s;

fi[i] = in;

ind[in] = (int \*)malloc(sizeof(int) \* s);

for (int j = 0; j < s; j++)

{

while (true)

{

in = rand() % 500;

if (h[in] == 0)

break;

}

h[in] = i + 1;

ind[fi[i]][j] = in;

}

}

}

cout << "\nAssigned blocks are:-\n\n";

for (int i = 0; i < n; i++)

{

int j = fi[i];

if (o == 1)

while (h[fi[i]] == h[j++])

cout << j + 1 << "-->" << i + 1 << endl;

else if (o == 2)

while (h[j] != -1)

{

cout << j + 1 << "-->" << i + 1 << endl;

j = h[j];

}

else

for (int k = 0; k < h[j]; k++)

cout << j + 1 << ":" << ind[j][k] << "-->" << i + 1 << endl;

cout << endl;

}

}

**SCREENSHOT**

# 

# 

## 

**OUTPUT**

# SEQUENTIAL

# 

# LINKED

# 

# INDEXED

# [**CLICK HERE FOR GITHHUB LINK**](https://github.com/Bhaumik-Tandan/OS_LABDA_3)