**EXPERIMENT: 13** ToConstruct a C program for implementation of the various memory allocation strategies.

**PROGRAM:**

#include <stdio.h>

#define BLOCKS 5

#define PROCESSES 4

// Function for First Fit allocation

void firstFit(int blockSize[], int processSize[]) {

int allocation[PROCESSES];

for (int i = 0; i < PROCESSES; i++) allocation[i] = -1;

for (int i = 0; i < PROCESSES; i++) {

for (int j = 0; j < BLOCKS; j++) {

if (blockSize[j] >= processSize[i]) {

allocation[i] = j;

blockSize[j] -= processSize[i];

break;

}

}

}

printf("\nFirst Fit Allocation:\n");

for (int i = 0; i < PROCESSES; i++) {

if (allocation[i] != -1)

printf("Process %d (Size %d) -> Block %d\n", i+1, processSize[i], allocation[i]+1);

else

printf("Process %d (Size %d) -> Not Allocated\n", i+1, processSize[i]);

}

}

// Function for Best Fit allocation

void bestFit(int blockSize[], int processSize[]) {

int allocation[PROCESSES];

for (int i = 0; i < PROCESSES; i++) allocation[i] = -1;

for (int i = 0; i < PROCESSES; i++) {

int bestIdx = -1;

for (int j = 0; j < BLOCKS; j++) {

if (blockSize[j] >= processSize[i]) {

if (bestIdx == -1 || blockSize[j] < blockSize[bestIdx])

bestIdx = j;

}

}

if (bestIdx != -1) {

allocation[i] = bestIdx;

blockSize[bestIdx] -= processSize[i];

}

}

printf("\nBest Fit Allocation:\n");

for (int i = 0; i < PROCESSES; i++) {

if (allocation[i] != -1)

printf("Process %d (Size %d) -> Block %d\n", i+1, processSize[i], allocation[i]+1);

else

printf("Process %d (Size %d) -> Not Allocated\n", i+1, processSize[i]);

}

}

// Function for Worst Fit allocation

void worstFit(int blockSize[], int processSize[]) {

int allocation[PROCESSES];

for (int i = 0; i < PROCESSES; i++) allocation[i] = -1;

for (int i = 0; i < PROCESSES; i++) {

int worstIdx = -1;

for (int j = 0; j < BLOCKS; j++) {

if (blockSize[j] >= processSize[i]) {

if (worstIdx == -1 || blockSize[j] > blockSize[worstIdx])

worstIdx = j;

}

}

if (worstIdx != -1) {

allocation[i] = worstIdx;

blockSize[worstIdx] -= processSize[i];

}

}

printf("\nWorst Fit Allocation:\n");

for (int i = 0; i < PROCESSES; i++) {

if (allocation[i] != -1)

printf("Process %d (Size %d) -> Block %d\n", i+1, processSize[i], allocation[i]+1);

else

printf("Process %d (Size %d) -> Not Allocated\n", i+1, processSize[i]);

}

}

int main() {

int blockSize[BLOCKS] = {100, 500, 200, 300, 600};

int processSize[PROCESSES] = {212, 417, 112, 426};

int b1[BLOCKS], b2[BLOCKS], b3[BLOCKS];

for (int i = 0; i < BLOCKS; i++) {

b1[i] = b2[i] = b3[i] = blockSize[i];

}

firstFit(b1, processSize);

bestFit(b2, processSize);

worstFit(b3, processSize);

return 0;

}

**OUTPUT:**

A screenshot of a computer program

AI-generated content may be incorrect.