**EXPERIMENT – 36**

36. With linked allocation, each file is a linked list of disk blocks; the disk blocks may be scattered

anywhere on the disk. The directory contains a pointer to the first and last blocks of the file. Each

block contains a pointer to the next block. Design a C program to simulate the file allocation

strategy.

#include <stdio.h>

#include <stdlib.h>

#define MAX\_BLOCKS 100

typedef struct Block {

int blockID;

struct Block\* next;

} Block;

typedef struct File {

char name[20];

Block\* start;

Block\* end;

int size;

} File;

Block\* freeBlocks[MAX\_BLOCKS];

int blockUsed[MAX\_BLOCKS] = {0};

void initFreeBlocks() {

for (int i = 0; i < MAX\_BLOCKS; i++) {

freeBlocks[i] = (Block\*)malloc(sizeof(Block));

freeBlocks[i]->blockID = i;

freeBlocks[i]->next = NULL;

}

}

Block\* getFreeBlock() {

for (int i = 0; i < MAX\_BLOCKS; i++) {

if (!blockUsed[i]) {

blockUsed[i] = 1;

return freeBlocks[i];

}

}

return NULL;

}

void allocateFile(File\* file) {

int n;

printf("Enter file name: ");

scanf("%s", file->name);

printf("Enter number of blocks to allocate: ");

scanf("%d", &n);

file->start = file->end = NULL;

file->size = 0;

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

Block\* newBlock = getFreeBlock();

if (!newBlock) {

printf("Disk is full! Allocation stopped at %d blocks.\n", i);

break;

}

if (file->start == NULL) {

file->start = file->end = newBlock;

} else {

file->end->next = newBlock;

file->end = newBlock;

}

file->size++;

}

printf("File '%s' allocated with %d blocks.\n", file->name, file->size);

printf("Start block: %d, End block: %d\n", file->start->blockID, file->end->blockID);

}

void displayFileBlocks(File file) {

Block\* current = file.start;

printf("Blocks for file '%s':\n", file.name);

while (current) {

printf("%d -> ", current->blockID);

current = current->next;

}

printf("NULL\n");

}

int main() {

File file;

int choice;

initFreeBlocks();

while (1) {

printf("\n--- Linked File Allocation Menu ---\n");

printf("1. Allocate File\n");

printf("2. Display File Blocks\n");

printf("3. Exit\n");

printf("Enter choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

allocateFile(&file);

break;

case 2:

displayFileBlocks(file);

break;

case 3:

printf("Exiting...\n");

exit(0);

default:

printf("Invalid choice!\n");

}

}

return 0;

}

SAMPLE OUTPUT:

--- Linked File Allocation Menu ---

1. Allocate File

2. Display File Blocks

3. Exit

Enter choice: 1

Enter file name: demo.txt

Enter number of blocks to allocate: 4

File 'demo.txt' allocated with 4 blocks.

Start block: 0, End block: 3

Enter choice: 2

Blocks for file 'demo.txt':

0 -> 1 -> 2 -> 3 -> NULL