Linked

#include <stdio.h>

// Structure to store file details

struct file {

char fname[10]; // File name

int start; // Starting block of the file

int size; // Number of blocks used by the file

int block[10]; // Blocks allocated to the file (linked)

} f[10];

int main() {

int i, j, n;

// Input: number of files

printf("Enter number of files: ");

scanf("%d", &n);

// Input details for each file

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

printf("\nEnter file name: ");

scanf("%s", f[i].fname);

printf("Enter starting block: ");

scanf("%d", &f[i].start);

// Store starting block in block[0]

f[i].block[0] = f[i].start;

printf("Enter number of blocks: ");

scanf("%d", &f[i].size);

printf("Enter block numbers (linked order): ");

for (j = 1; j <= f[i].size; j++) {

scanf("%d", &f[i].block[j]); // block[1] to block[size]

}

}

// Display file allocation table

printf("\nFile\tStart\tSize\tBlocks (Linked)\n");

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

printf("%s\t%d\t%d\t", f[i].fname, f[i].start, f[i].size);

for (j = 0; j < f[i].size; j++) {

printf("%d--->", f[i].block[j]);

}

// Print last block without arrow

printf("%d\n", f[i].block[j]);

}

return 0;

}

START

1. Input number of files → n

2. Repeat for each file i = 0 to n-1:

a. Input file name → fname

b. Input starting block → start

c. Save starting block to block[0]

d. Input number of blocks → size

e. Input block numbers from 1 to size (linked structure)

3. Print table header

4. For each file:

a. Print file name, start, size

b. Print all blocks in linked format:

block[0] ---> block[1] ---> ... ---> block[size]

END