(35)

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_RECORDS 100

#define RECORD\_SIZE 256

typedef struct {

char data[RECORD\_SIZE];

} Record;

typedef struct {

Record records[MAX\_RECORDS];

int count;

} FileSystem;

void initializeFileSystem(FileSystem \*fs) {

fs->count = 0;

}

int addRecord(FileSystem \*fs, const char \*data) {

if (fs->count >= MAX\_RECORDS) {

return -1; // File system is full

}

strncpy(fs->records[fs->count].data, data, RECORD\_SIZE);

fs->count++;

return 0; // Success

}

void readRecords(const FileSystem \*fs) {

for (int i = 0; i < fs->count; i++) {

printf("Record %d: %s\n", i + 1, fs->records[i].data);

}

}

int main() {

FileSystem fs;

initializeFileSystem(&fs);

addRecord(&fs, "First record");

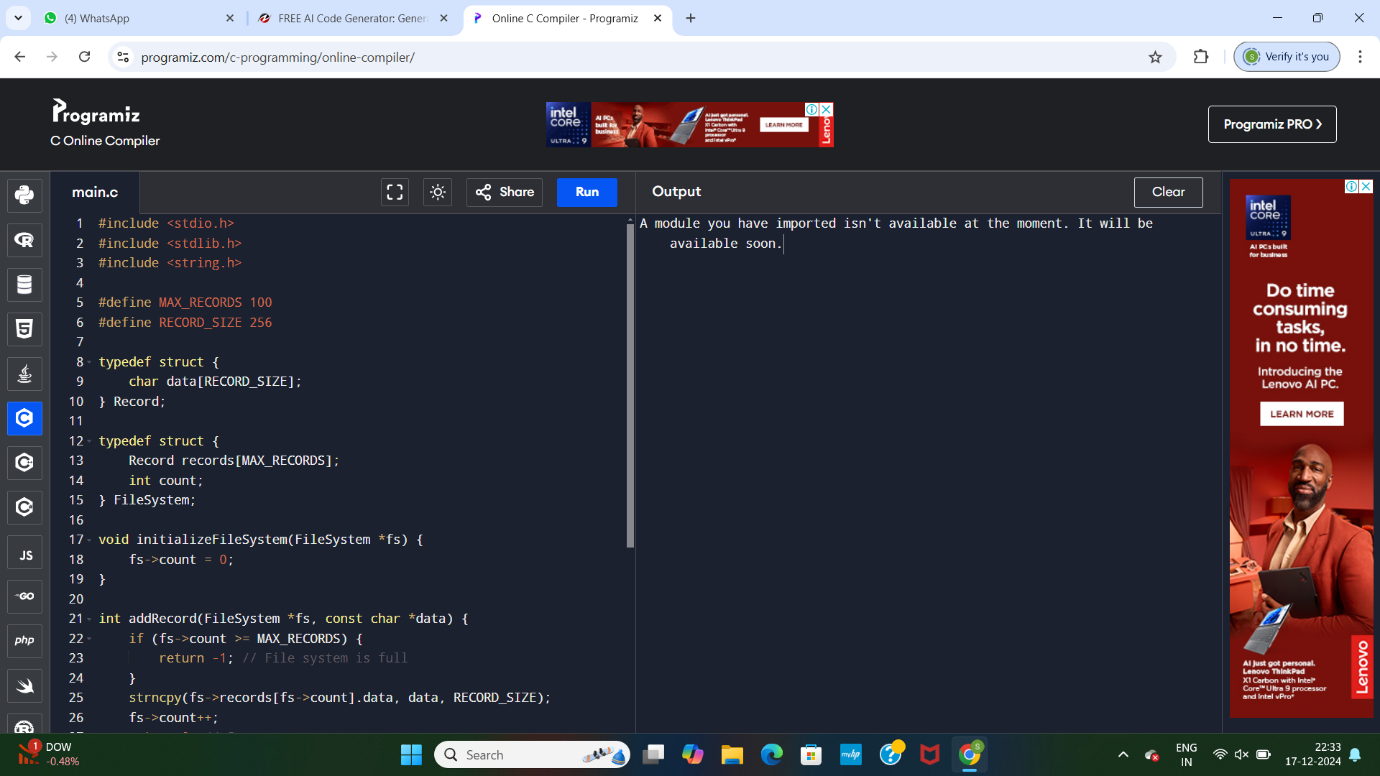
addRecord(&fs, "Second record");

addRecord(&fs, "Third record");

readRecords(&fs);

return 0;

}



(37)

#include <stdio.h>

void findWaitingTime(int processes[], int n, int bt[], int wt[]) {

wt[0] = 0;

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

wt[i] = bt[i - 1] + wt[i - 1];

}

void findTurnAroundTime(int processes[], int n, int bt[], int wt[], int tat[]) {

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

tat[i] = bt[i] + wt[i];

}

void findavgTime(int processes[], int n, int bt[]) {

int wt[n], tat[n];

findWaitingTime(processes, n, bt, wt);

findTurnAroundTime(processes, n, bt, wt, tat);

float total\_wt = 0, total\_tat = 0;

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

total\_wt += wt[i];

total\_tat += tat[i];

}

printf("Average waiting time: %.2f\n", total\_wt / n);

printf("Average turnaround time: %.2f\n", total\_tat / n);

}

int main() {

int processes[] = { 1, 2, 3};

int n = sizeof(processes) / sizeof(processes[0]);

int burst\_time[] = { 10, 5, 8 };

findavgTime(processes, n, burst\_time);

return 0;

}

