МІНІСТЕРСТВО ОСВІТИ і НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Факультет прикладної математики
Кафедра системного програмування і спеціалізованих комп'ютерних систем

Лабораторна робота №4

з дисципліни **"Введення до оперативних систем"**

Тема: **" Файлові системи"**

Варіант 9

Виконав:
студент IV-го курсу ФПМ
групи КВ-82
Іваненко О.А.

Київ 2021

Метою лабораторної роботи є ознайомлення з основними підходами до фізичної організації файлових систем та їх реалізацією у файлових системах

Завдання на виконання роботи

1. Написати програму, що моделює роботу складових заданої файлової системи згідно варіанта (перелік варіантів представлений нижче).
Вхідні дані студент задає самостійно з урахуванням особливостей індивідуального варіанта завдання.

2. Зробити візуалізацію роботи програми і кінцевих результатів на різних наборах вхідних даних.

Варіант 9

9. *Побудовати таблицю розміщення файлів відповідно до моделі файлової системи FAT-16.*

Імітувати процеси створення файлів заданого у кластерах розміру, збільшення файлів на задану кількість кластерів та знищення файлів. Передбачити пошук файлів за їх ідентифікатором та роздруківку номерів кластерів, які займає файл.

Лістинг програми

```
package com;

import java.util.Scanner;

class Cluster{
    final int id;
    int next;

    Cluster(int id){
        this.id = id;
        this.next = 0;
    }

    void setNext(int next){
        this.next = next;
    }
}

class File{
    String filename;
    int firstCluster;
    int size; // in clusters

    File(String filename, int firstCluster, int size){
        this.filename = filename;
        this.firstCluster = firstCluster;
        this.size = size;
    }
}

class FAT {
    Cluster[] table = new Cluster[256];
    File[] rootDirectory = new File[10];
    int rootDirectorySize = 0;

    FAT(){
        for(int i = 0; i < 256; i++){
            table[i] = new Cluster(i);
        }
    }

    void printClusters(){
        System.out.print("\n\n-------------------------------------------------------------
------------------------------");
        System.out.print("\n      | ");
        for(int i = 0; i < 16; i++) System.out.printf("0x%02X ", i);
        System.out.print("\n------|-----------------------------------------------------
----------------------------");
        for(int i = 0; i < table.length; i++){
            if(i % 16 == 0) {
                System.out.print("\n");
                System.out.printf("|0x%02X | ", i / 16);
            }
            System.out.printf("0x%02X ", table[i].next);
        }
    }

    void printRootTable(){
        System.out.println("\nRoot directory table:");
        for (int i = 0; i < rootDirectory.length; i++) {
            if (rootDirectory[i] != null) {
```

```
                System.out.printf("Index: %d, Filename: %s, First cluster: 0x%02X,
Size: %d clusters%n", i,
                        rootDirectory[i].filename, rootDirectory[i].firstCluster,
rootDirectory[i].size);
            }
        }
    }

    void createTest0(){
        rootDirectory[rootDirectorySize] = new File("foo.txt", 0x70, 9);
        rootDirectorySize++;
        table[0x70].setNext(0x71);
        table[0x71].setNext(0x72);
        table[0x72].setNext(0x55);

        table[0x55].setNext(0x56);
        table[0x56].setNext(0x03);

        table[0x03].setNext(0x04);
        table[0x04].setNext(0x9c);

        table[0x9c].setNext(0x9D);
        table[0x9D].setNext(0xff);
    }

    int addCluster() {
        for (int i = 1; i < table.length; i++) {
            if (table[i].next == 0x00) {
                for (int j = i + 1; j < table.length; j++) {
                    if (table[j].next == 0x00) {
                        table[i].next = j;
                        return i;
                    }
                }
            }
        }
        return -1;
    }

    void addLastCluster(){
        for(int i = 1; i < table.length; i++) {
            if (table[i].next == 0x00) {
                table[i].next = 0xFF;
                break;
            }
        }
    }

    void addClusterToExist(int index) {
        for(int i = 1; i < table.length; i++) {
            if (table[i].next == 0x00) {
                System.out.println(rootDirectory[index].firstCluster);
                System.out.println(i);
                table[i].next = rootDirectory[index].firstCluster;
                rootDirectory[index].firstCluster = i;
                rootDirectory[index].size++;
                break;
            }
        }
    }

    void addFile(String name, int size){
        if(size == 0) return;
```

```
        rootDirectory[rootDirectorySize] = new File(name, addCluster(), size);
        rootDirectorySize++;

        for(int i = 0; i < size - 2; i++) addCluster();

        addLastCluster();
    }

    void increaseFile(int index, int number){
        for(int i = 0; i < number; i++) { addClusterToExist(index); }
    }

    void deleteFile(int index){
        int next = rootDirectory[index].firstCluster;
        do{
            int tmp = table[next].next;
            table[next].next = 0x00;
            next = tmp;
            rootDirectory[index].size--;

        }while(rootDirectory[index].size != 0);

        rootDirectory[index] = null;
    }

    public void printArrayOfClusters(int index) {
        int curr = rootDirectory[index].firstCluster;
        System.out.print("Cluster array of file #" + index + ": ");
        do{
            System.out.printf("0x%02X ", curr);
            curr = table[curr].next;
        }while(curr != 0xFF);
    }
}



public class Main {

    private static void createFile(FAT fat){
        Scanner sc = new Scanner(System.in);
        System.out.print("Input name: ");
        String name = sc.nextLine();
        System.out.print("\nInput size: ");
        int size = sc.nextInt();
        fat.addFile(name, size);
        System.out.println("File created! Check FAT table.");
    }

    private static void removeFile(FAT fat){
        Scanner sc = new Scanner(System.in);
        System.out.print("Input index: ");
        int index = sc.nextInt();
        fat.deleteFile(index);
        System.out.println("File removed! Check FAT table.");
    }

    private static void increaseFile(FAT fat){
        Scanner sc = new Scanner(System.in);
        System.out.print("Input index: ");
        int index = sc.nextInt();
```

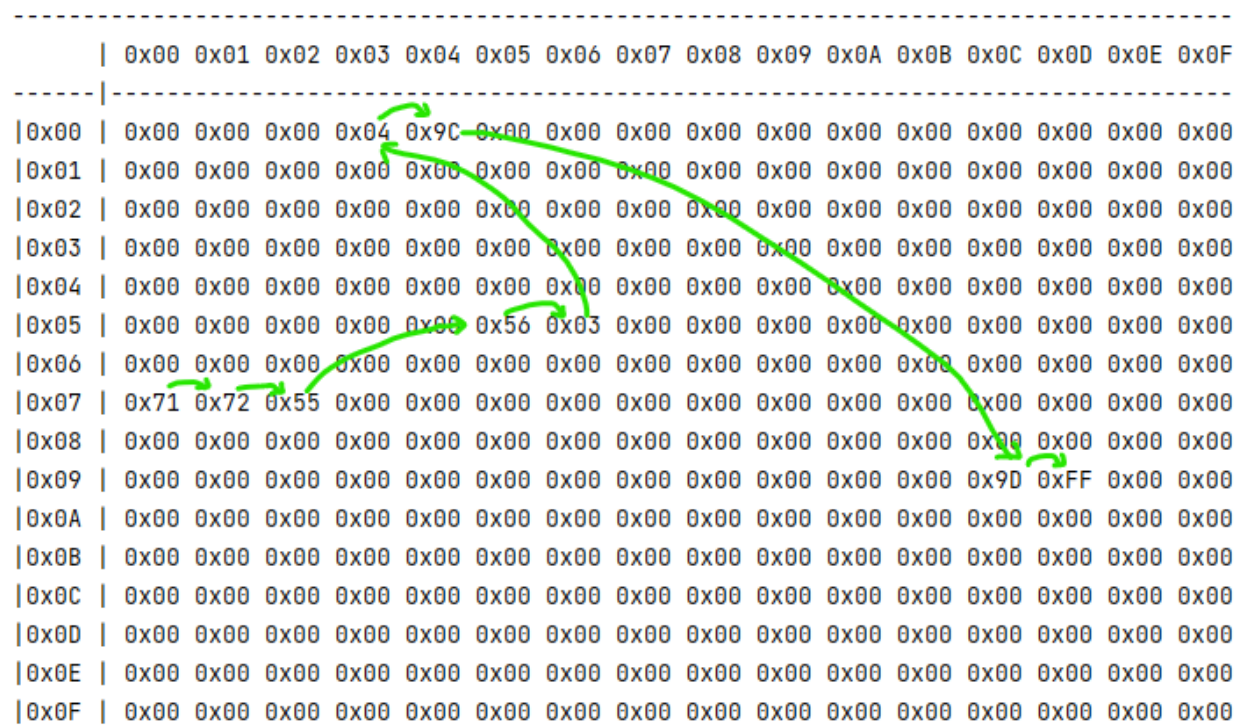
```
        System.out.print("\nInput amount of increasing:");
        int amount = sc.nextInt();
        fat.increaseFile(index, amount);
        System.out.println("File increased! Check FAT table.");
    }

    private static void printArrayOfClusters(FAT fat){
        Scanner sc = new Scanner(System.in);
        System.out.print("Input index: ");
        int index = sc.nextInt();
        fat.printArrayOfClusters(index);
    }

    public static void main(String[] args) {
        FAT fat = new FAT();
        fat.createTest0();
        Scanner sc = new Scanner(System.in);
        int n = 1;
        while (n != 0){
            System.out.println("\n\n\nMENU:");
            System.out.println("1. Print FAT table");
            System.out.println("2. Print directory");
            System.out.println("3. Create file");
            System.out.println("4. Increase file size");
            System.out.println("5. Remove file");
            System.out.println("6. Print array of clusters");
            System.out.println("0. Exit");
            n = sc.nextInt();

            switch(n){
                case 1: fat.printClusters(); break;
                case 2: fat.printRootTable(); break;
                case 3: createFile(fat); break;
                case 4: increaseFile(fat); break;
                case 5: removeFile(fat); break;
                case 6: printArrayOfClusters(fat); break;
                default: break;
            }
        }
    }
}
```

Тестування програми

Пункт меню:

```
MENU:
1. Print FAT table
2. Print directory
3. Create file
4. Increase file size
5. Remove file
6. Print array of clusters
0. Exit
```

Початкова FAT таблиця. Для спрощення візуалізації було обрано виводити 256 кластерів. Для тестування програми ручним способом занесено файл в таблицю каталогів та таблицю FAT.



```
------------------------------------------------------------------------------------------
      | 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
------|-----------------------------------------------------------------------------------
|0x00 | 0x00 0x00 0x00 0x04 0x9C 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x01 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x02 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x03 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x04 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x05 | 0x00 0x00 0x00 0x00 0x00 0x56 0x03 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x06 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x07 | 0x71 0x72 0x55 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x08 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x09 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x9D 0xFF 0x00 0x00
|0x0A | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0B | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0C | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0D | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0E | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0F | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Root directory table:
Index: 0, Filename: foo.txt, First cluster: 0x70, Size: 9 clusters
```

Можна проаналізувати таблицю FAT. Як ми бачимо – першим кластером э кластер 0x70. З нього, аналогічно до зв'язного списку йдуть інші байти.

Спробуємо додати новий файл до системи:

```
Input name: bar.txt

Input size: 8
File created! Check FAT table.
```

	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
0x00	0x00	0x02	0x05	0x04	0x9C	0x06	0x07	0x08	0x09	0x0A	0xFF	0x00	0x00	0x00	0x00	0x00
0x01	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x02	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x03	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x04	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x05	0x00	0x00	0x00	0x00	0x00	0x56	0x03	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x06	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x07	0x71	0x72	0x55	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x08	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x09	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x9D	0xFF	0x00	0x00
0x0A	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x0B	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x0C	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x0D	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x0E	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x0F	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00

Як можна побачити, при додаванні нового файлу до програми, ми заповнюємо кластери послідовністю, пропускаючи вже зайняті байти.

Тепер спробуємо збільшити перший файл на певну кількість кластерів.

```
Input index: 0

Input amount of increasing:8

-------------------------------------------------------------------------------
      | 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
------|------------------------------------------------------------------------
|0x00 | 0x00 0x02 0x05 0x04 0x9C 0x06 0x07 0x08 0x09 0x0A 0xFF 0x70 0x0B 0x0C 0x0D 0x0E
|0x01 | 0x0F 0x10 0x11 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x02 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x03 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x04 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x05 | 0x00 0x00 0x00 0x00 0x00 0x56 0x03 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x06 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x07 | 0x71 0x72 0x55 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x08 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x09 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x9D 0xFF 0x00 0x00
|0x0A | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0B | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0C | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0D | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0E | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0F | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Root directory table:
Index: 0, Filename: foo.txt, First cluster: 0x12, Size: 17 clusters
Index: 1, Filename: bar.txt, First cluster: 0x01, Size: 8 clusters
```

Темно зеленим кольором позначено нові кластери, які відносяться до першого файлу. Для запобігання великої зовнішньої фрагментації, було обрано додавати по одному кластеру за раз. Кожний новий кластер починає вказувати на попередній.

Тепер протестуємо видалення файлу. Видалимо другий файл(позначенний жовтим кольором)

```
-------------------------------------------------------------------------------
      | 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
------|------------------------------------------------------------------------
|0x00 | 0x00 0x00 0x00 0x04 0x9C 0x00 0x00 0x00 0x00 0x00 0x00 0x70 0x0B 0x0C 0x0D 0x0E
|0x01 | 0x0F 0x10 0x11 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x02 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x03 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x04 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x05 | 0x00 0x00 0x00 0x00 0x00 0x56 0x03 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x06 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x07 | 0x71 0x72 0x55 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x08 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x09 | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x9D 0xFF 0x00 0x00
|0x0A | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0B | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0C | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0D | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0E | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
|0x0F | 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00


Root directory table:
Index: 0, Filename: foo.txt, First cluster: 0x12, Size: 17 clusters
```

Як ми можемо бачити, ми звільнили ці кластери від файлу, та залишився лише перший файл.

Продемонструємо роботу роздруківки кластерів:

```
Input index: 0
Cluster array of file #0: 0x12 0x11 0x10 0x0F 0x0E 0x0D 0x0C 0x0B 0x70 0x71 0x72 0x55 0x56 0x03 0x04 0x9C 0x9D 0xFF
```

Якщо порівняти з таблицею, можна зробити висновок, що кластери друкуються правильно.

Висновок

В цій лабораторній було досліджено файлову систему FAT16. Система досить проста, через що має плюси та мінуси.

До плюсів можна віднести просте створення нових файлів, оскільки потрібно лише оновити таблицю FAT та таблицю каталогу. Збільшувати файл також просто, потрібно лише налаштовувати наступне поле в таблиці FAT, що і було продемонстровано в програмі.

Щодо використання пам’яті, також все добре, оскільки якщо блок пустий, його можна використовувати, та неважливо, зайняті сусідні, чи ні, через що, зовнішня фрагментація майже відсутня. Простота доступу, оскільки FAT система веде себе, як зв’язаний список, може бути як плюсом, так і мінусом, для послідовного доступу все буде добре, але якщо в нас буде випадковий доступ, нам потрібно проходити багато непотрібних кластерів.

Загалом, FAT є поганим вибором для ситуацій, коли необхідний випадковий доступ до великих файлів, однак це хороший спосіб для змінного сховища, де переважно копіюються файли з однієї машини на іншу.