

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**BỘ MÔN HỆ ĐIỀU HÀNH**

-----



**BÁO CÁO THỰC HÀNH**

**Họ tên: Vũ Ngọc Sơn**  
**Mã SV: B22DCKH104**  
**Lớp: D22CQKH02-B**

**GVHD: ThS.Nguyễn Đình Quân**

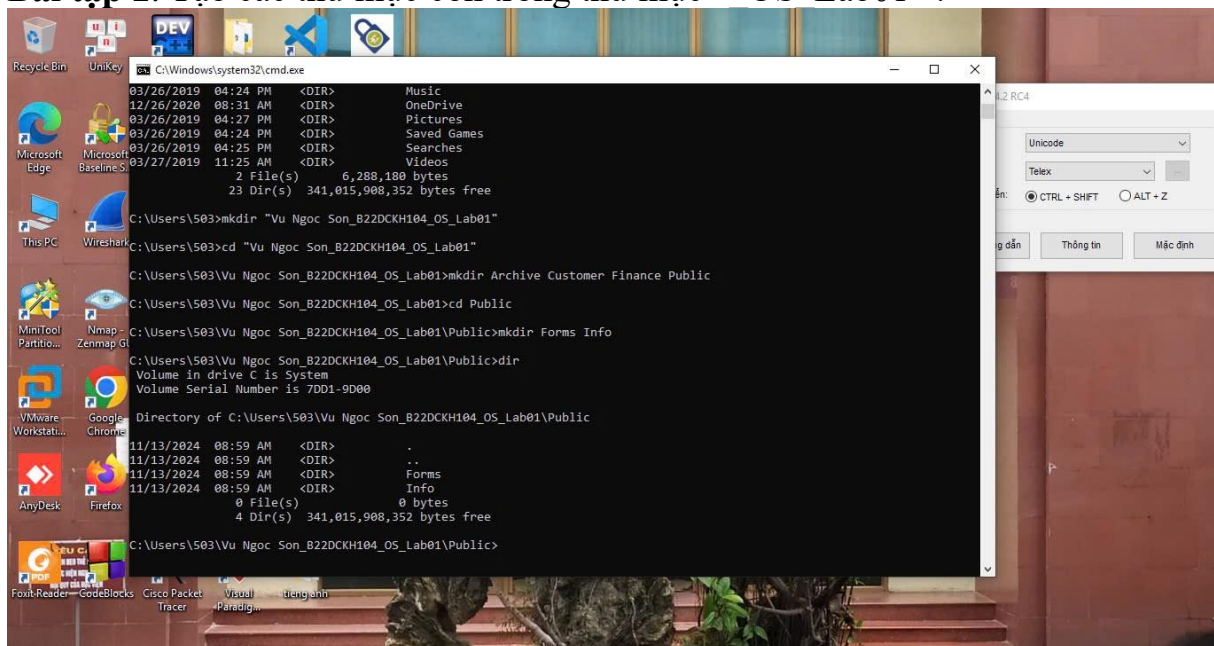
***Hà Nội, ngày 13 tháng 11 năm 2024***

## Mục lục

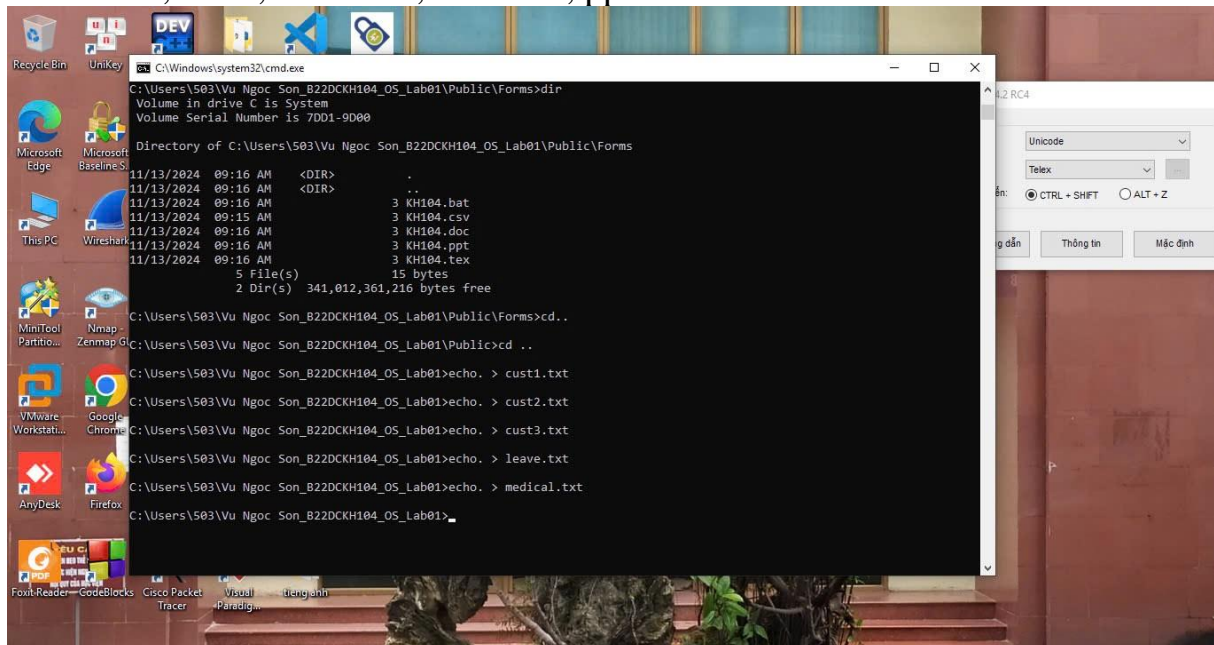
<b>I.Lab Exercise 1:</b>	1
<b>Bài tập 1:</b> Tạo các thư mục con trong thư mục “_OS_Lab01” :	1
<b>Bài tập 2:</b> Trong thư mục Forms của bài tập số 1, sử dụng câu lệnh tạo các file sau: Excel, word, batch file, latex file, pptx file.	2
<b>Bài tập 3:</b> Sử dụng thư mục trong bài tập 1 thực hiện các yêu cầu sau bằng dòng lệnh:	2
<b>II.Lab Exercise 3: Phần 3.2</b>	8
1.Đọc và in thông tin từ BOOT:	8
2.Đọc, phân tích, hiển thị nội dung bảng FAT:	8
3.Đọc, phân tích, hiển thị ROOT:	9
4.Duyệt số thứ tự hoặc nội dung các cluster của file cho trước:	10
5.Viết đoạn chương trình in ra nội dung giống như câu lệnh dir: Chương trình và kết quả được thực hiện như dưới đây:	11
6.Kết quả:	14

## I.Lab Exercise 1:

### Bài tập 1: Tạo các thư mục con trong thư mục “ OS Lab01” :

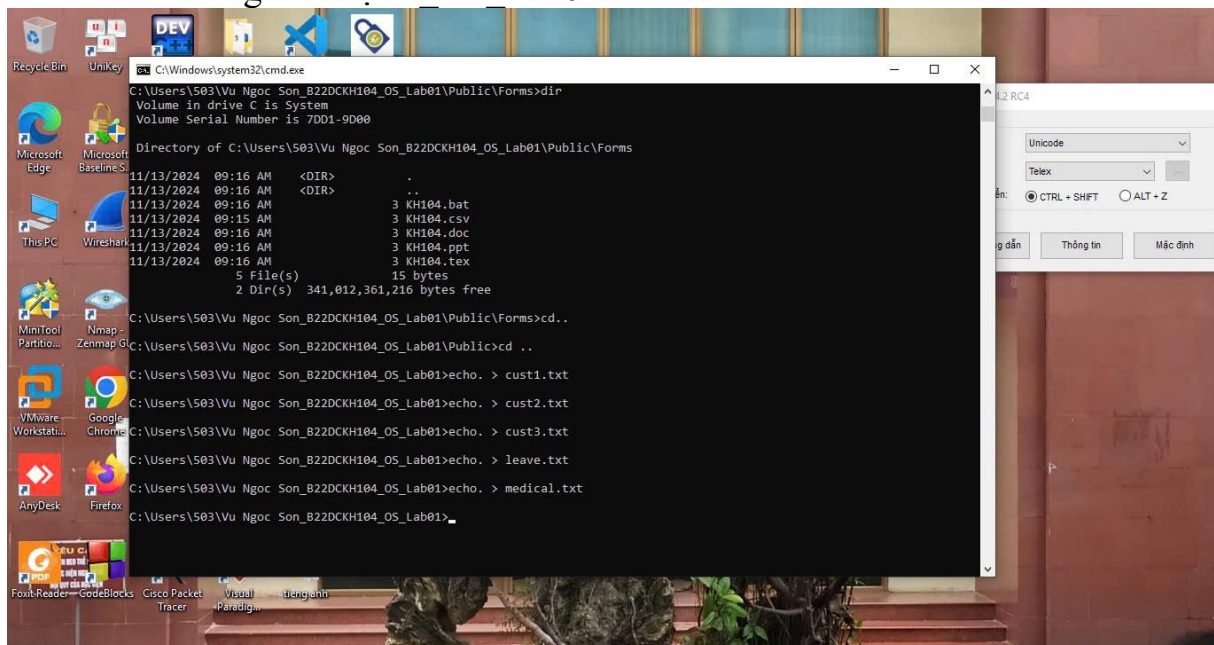


**Bài tập 2:** Trong thư mục Forms của bài tập số 1, sử dụng câu lệnh tạo các file sau: Excel, word, batch file, latex file, pptx file.

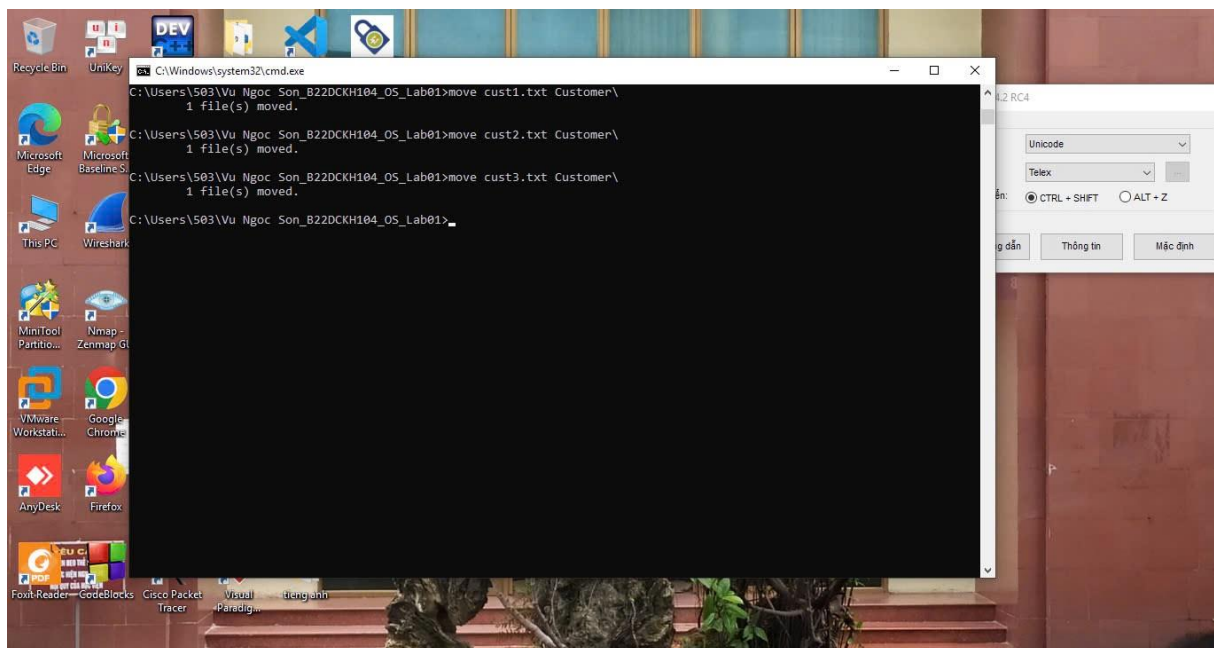


**Bài tập 3:** Sử dụng thư mục trong bài tập 1 thực hiện các yêu cầu sau bằng dòng lệnh:

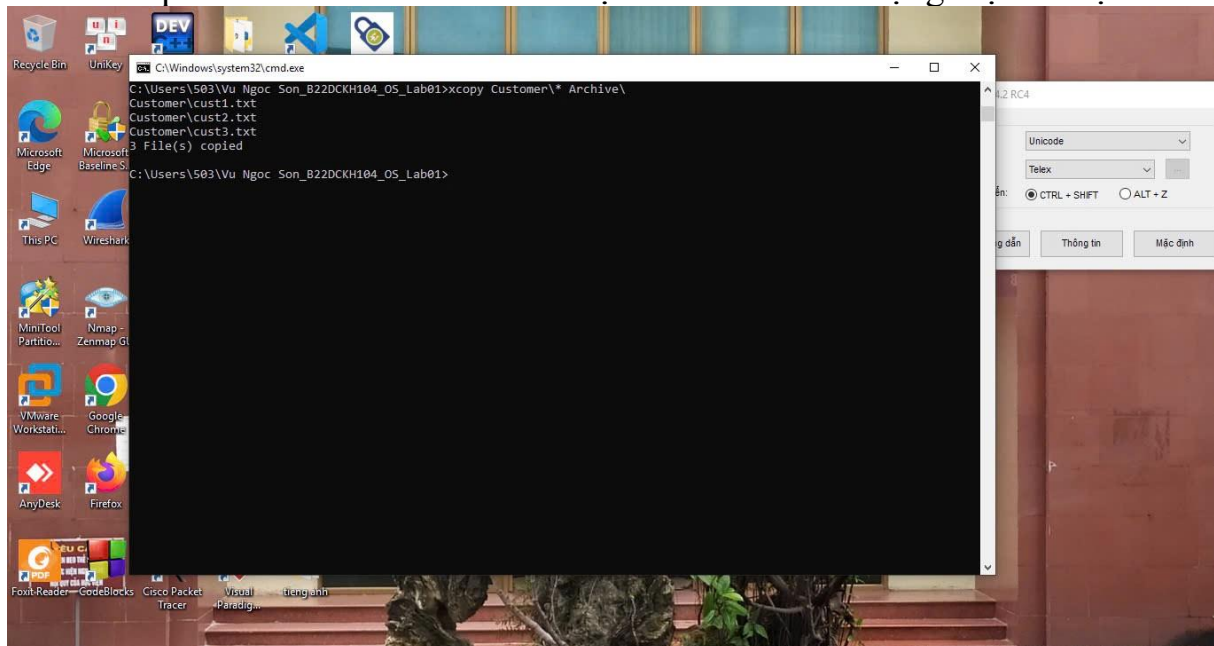
1. Tạo 3 tệp tin cust1.txt; cust2.txt and cust3.txt và hai tài liệu leave.txt và medical.txt trong thư mục “ OS Lab01”



2. Di chuyển 3 file Cust1.txt, Cust2.txt, Cust3.txt tới thư mục Customer

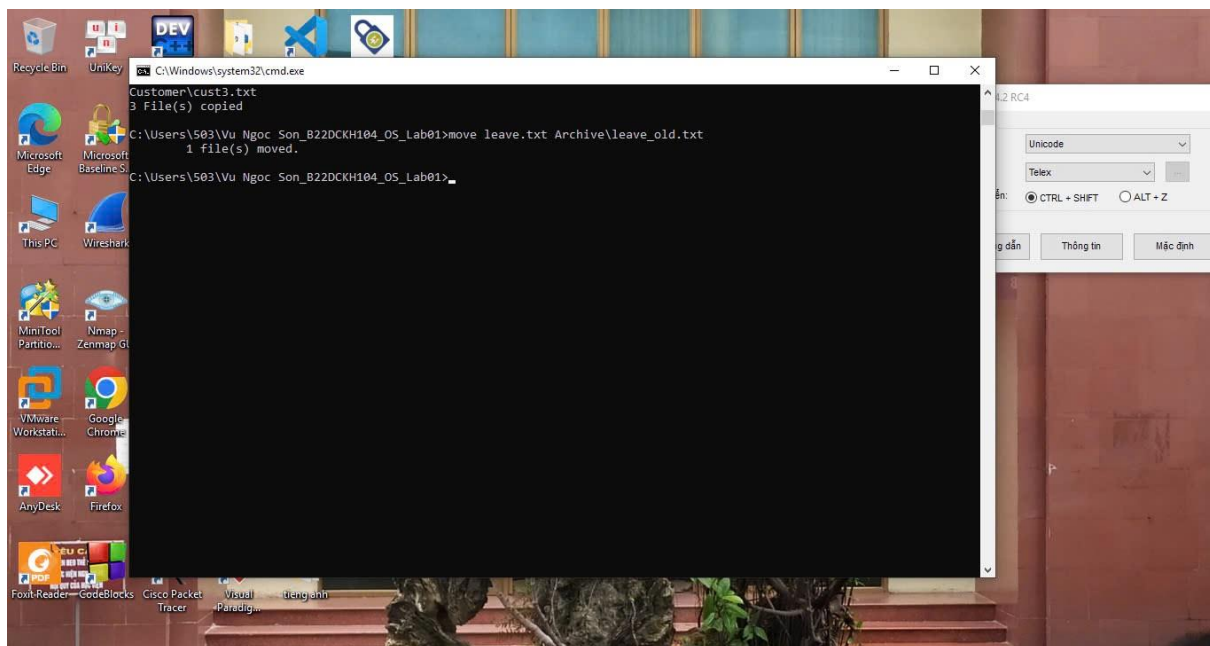


3. Sao chép 3 Customer files đến thư mục Archive chỉ sử dụng một câu lệnh

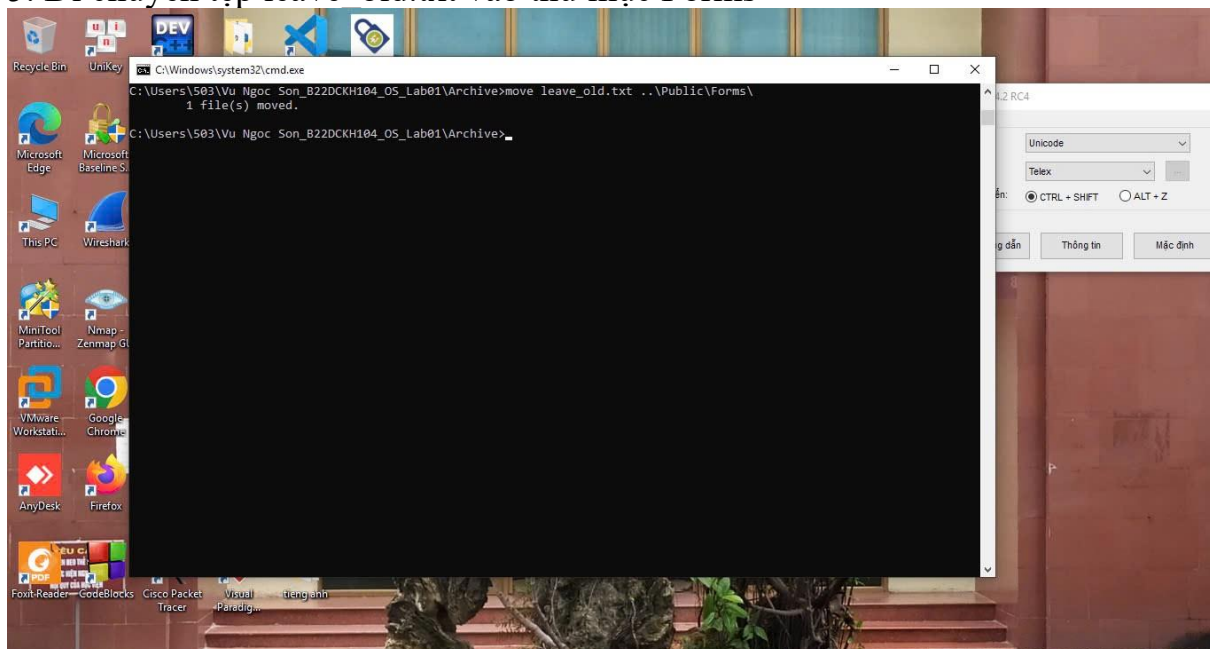


4. Di chuyển tệp leave.txt vào thư mục Archive và đổi tên nó thành leave\_old.txt

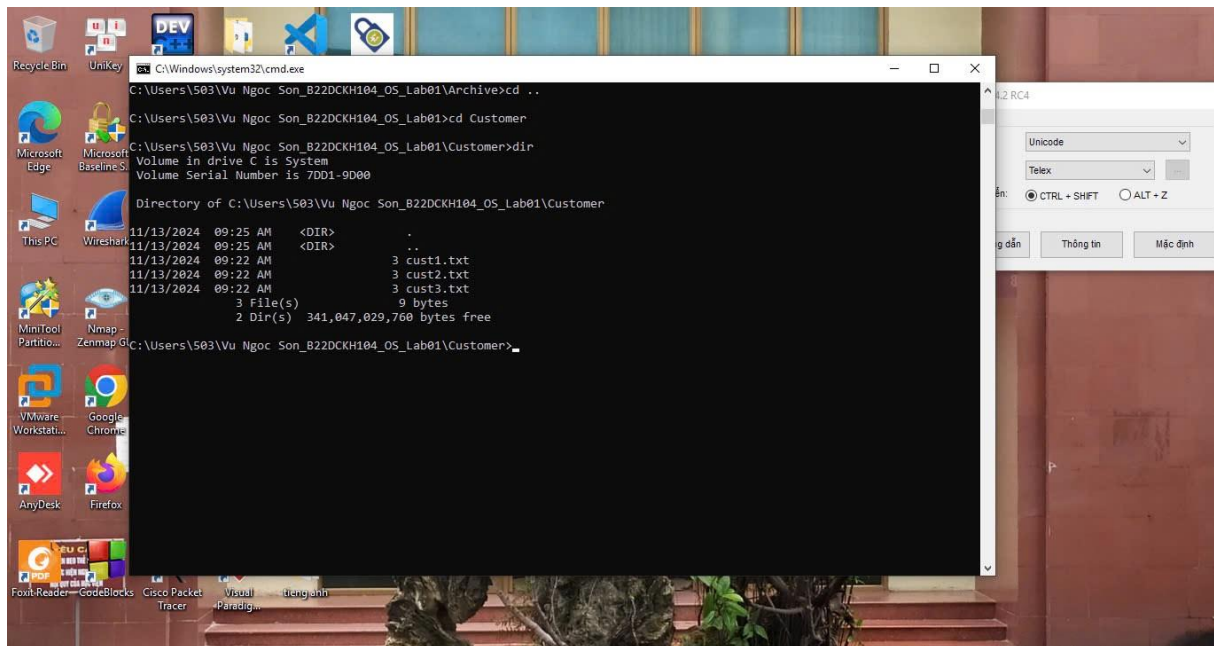




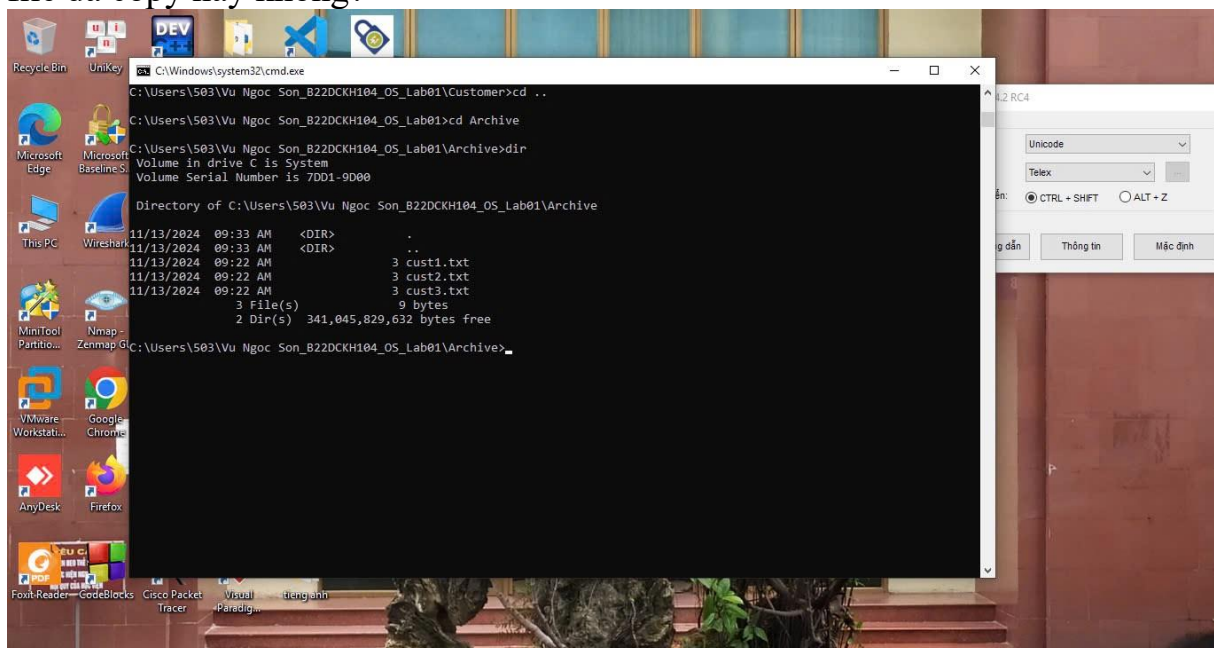
## 5. Di chuyển tệp leave\_old.txt vào thư mục Forms



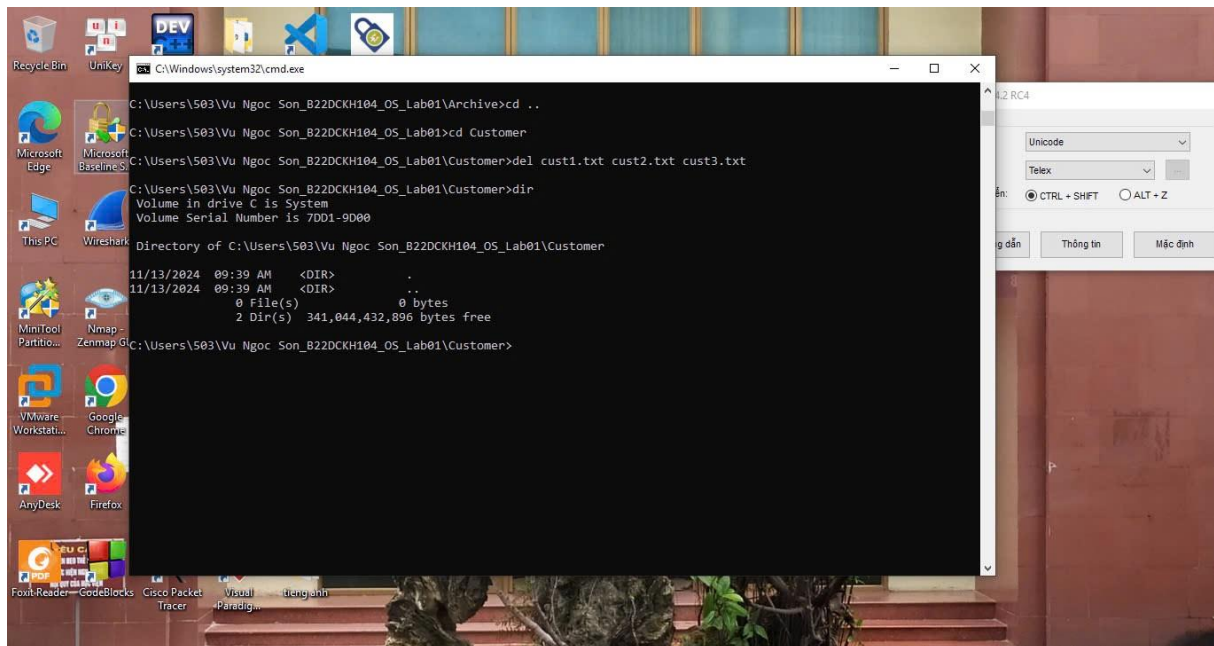
## 6. Thay đổi đường dẫn sang thư mục Customer và kiểm tra xem nó còn chưa 3 tệp hay không?



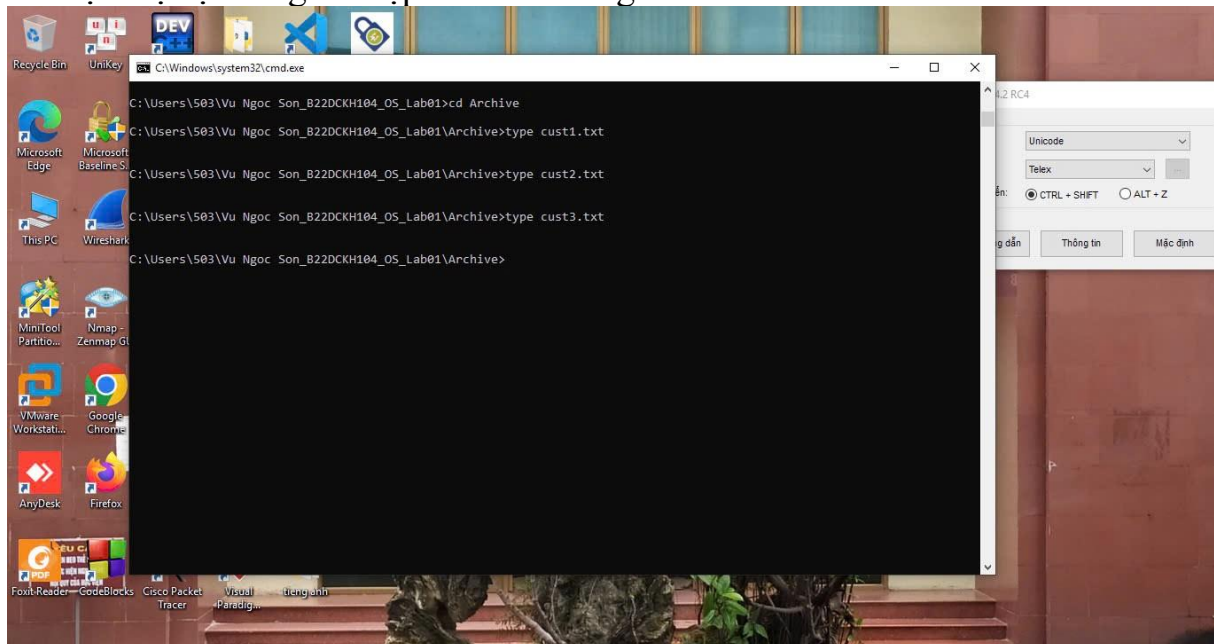
7. Thay đổi đường dẫn sang thư mục Archive và kiểm tra xem nó còn chứa 3 file đã copy hay không?



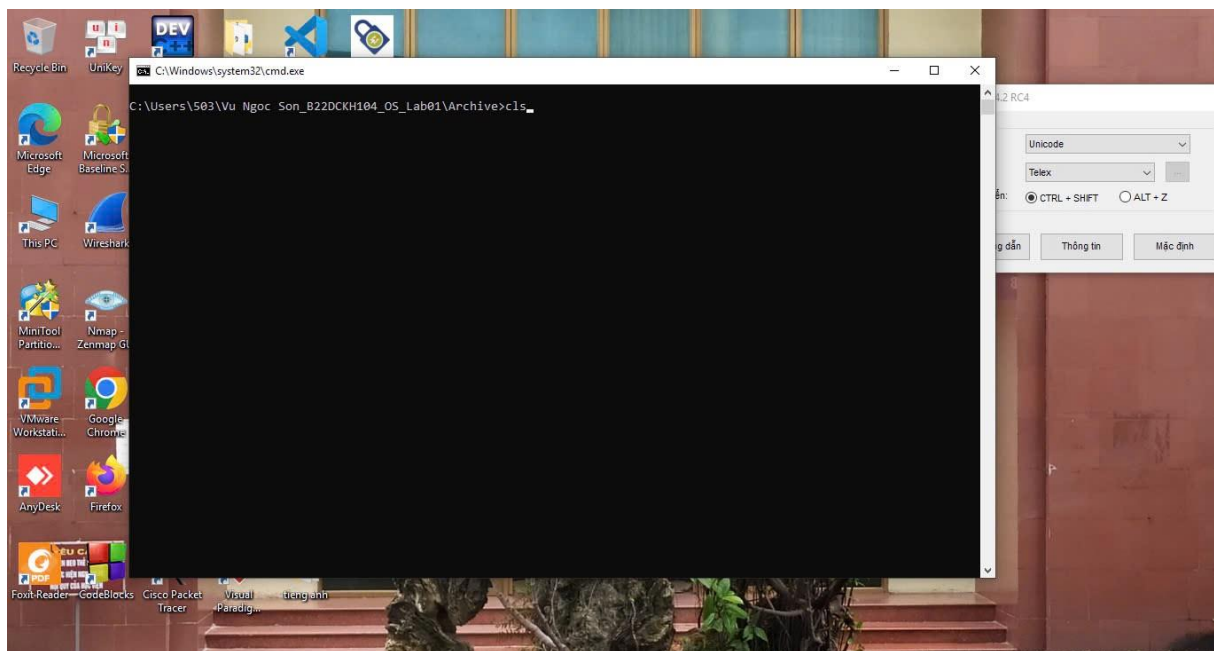
8. Thay đổi đường dẫn sang thư mục Customer và xóa 3 tệp Customer.



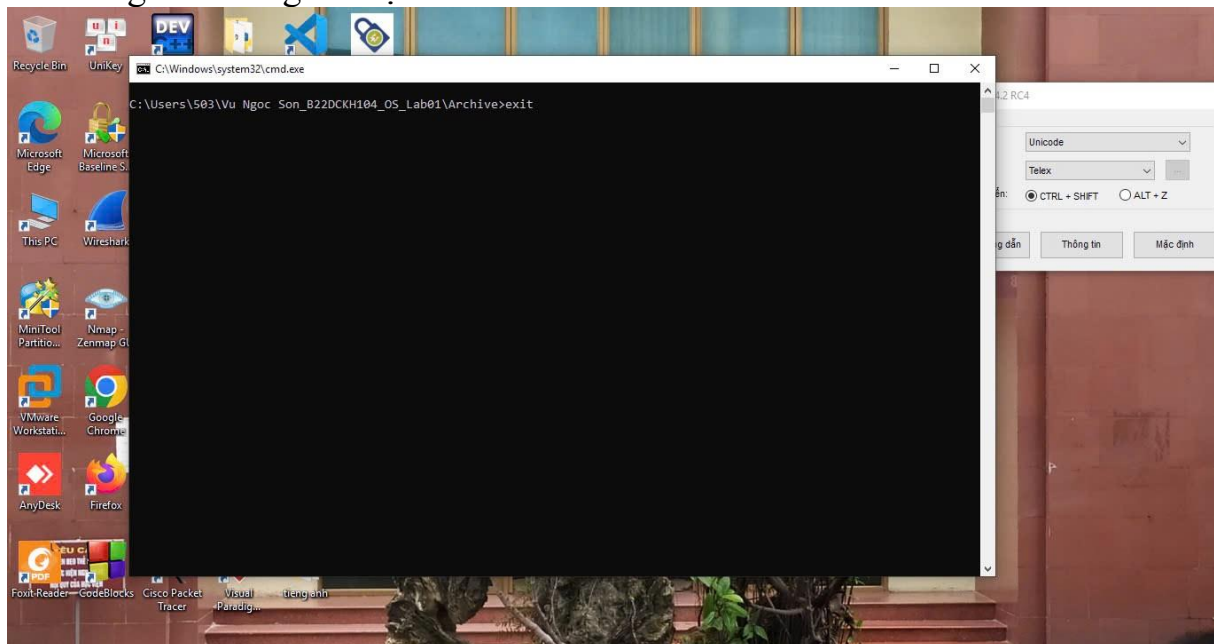
9. Hiện thị nội dung các tệp văn bản trong cửa sổ cmd của window.



10. Xóa màn hình cmd



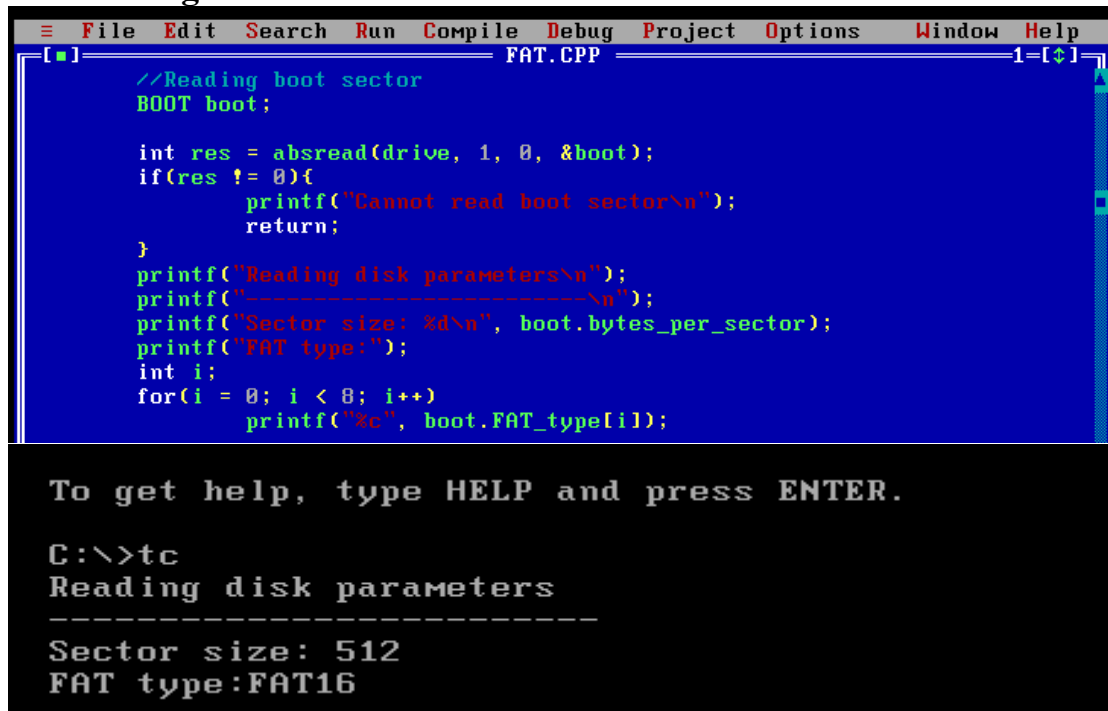
## 11. Đóng cmd bằng câu lệnh





## II.Lab Excercise 3: Phần 3.2

### 1.Đọc và in thông tin từ BOOT:



```
File Edit Search Run Compile Debug Project Options Window Help
FAT.CPP 1=[+]
```

```
//Reading boot sector
BOOT boot;

int res = absread(drive, 1, 0, &boot);
if(res != 0){
    printf("Cannot read boot sector\n");
    return;
}
printf("Reading disk parameters\n");
printf("-----\n");
printf("Sector size: %d\n", boot.bytes_per_sector);
printf("FAT type:");
int i;
for(i = 0; i < 8; i++)
    printf("%c", boot.FAT_type[i]);
```

To get help, type HELP and press ENTER.

C:\>tc

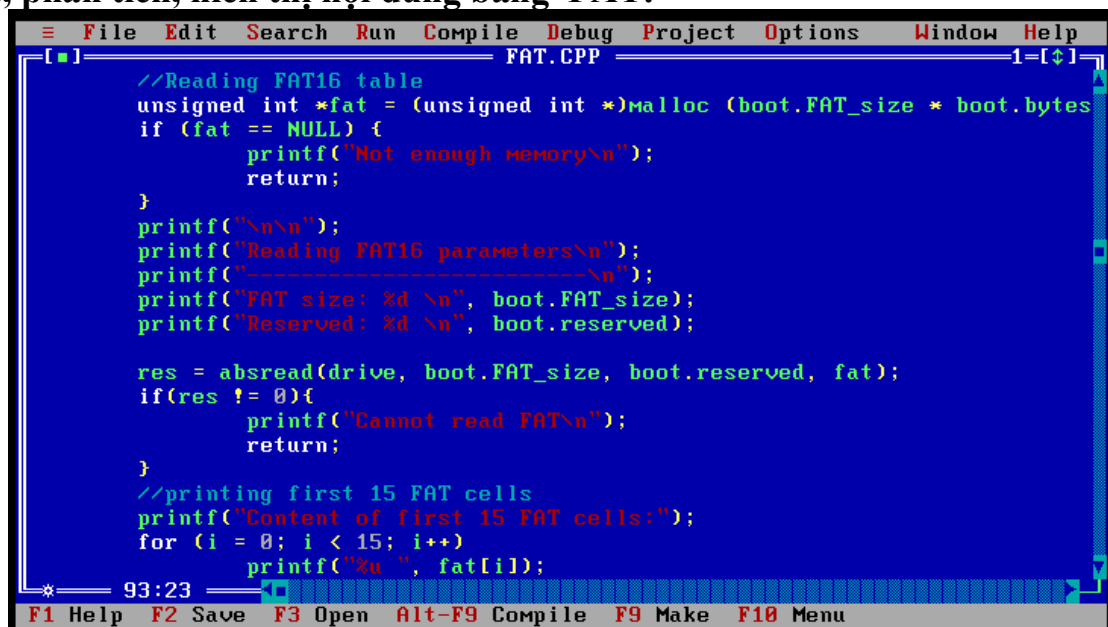
Reading disk parameters

-----

Sector size: 512

FAT type:FAT16

### 2.Đọc, phân tích, hiển thị nội dung bảng FAT:



```
File Edit Search Run Compile Debug Project Options Window Help
FAT.CPP 1=[+]
```

```
//Reading FAT16 table
unsigned int *fat = (unsigned int *)MALLOC(boot.FAT_size * boot.bytes_per_sector);
if (fat == NULL) {
    printf("Not enough memory\n");
    return;
}
printf("\n\n");
printf("Reading FAT16 parameters\n");
printf("-----\n");
printf("FAT size: %d\n", boot.FAT_size);
printf("Reserved: %d\n", boot.reserved);

res = absread(drive, boot.FAT_size, boot.reserved, fat);
if(res != 0){
    printf("Cannot read FAT\n");
    return;
}
//printing first 15 FAT cells
printf("Content of first 15 FAT cells:");
for (i = 0; i < 15; i++)
    printf("%u ", fat[i]);
```

\* 93:23

F1 Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu

```

File Edit Search Run Compile Debug Project Options Window Help
FAT.CPP

//Counting number of free clusters from first 100 clusters
int free_count = 0;
for (i = 2; i < 100; i++){ //first 2 clusters are not used
    if(fat[i] == 0) free_count++;
}
printf("\n");
printf("Number of free clusters from first 100 clusters:");
printf("%d\n", free_count);

//Printing clusters of a file from cluster n
unsigned int n = 5;
unsigned int cur = n;
printf("Clusters of a file from %u: ", n);
while(cur < 0xFFF8){
    printf("->%u", cur);
    cur = fat[cur];
}

Reading FAT16 parameters
-----
FAT size: 33
Reserved: 6
Content of first 15 FAT cells:65528 65535 3 4 5 6 7 8 9 65535 11 12 13 14 15
Number of free clusters from first 100 clusters:61
Clusters of a file from 5: ->5->6->7->8->9

```

### 3. Đọc, phân tích, hiển thị ROOT:

```

File Edit Search Run Compile Debug Project Options Window Help
FAT.CPP

//Reading ROOT
printf("\n\n");
printf("Reading ROOT information:\n");
printf("-----\n");

int num_byte = boot.ROOT_size * 32; //sizeof(ROOT)

ROOT *root = (ROOT *)malloc(num_byte);
if(root == NULL) return;

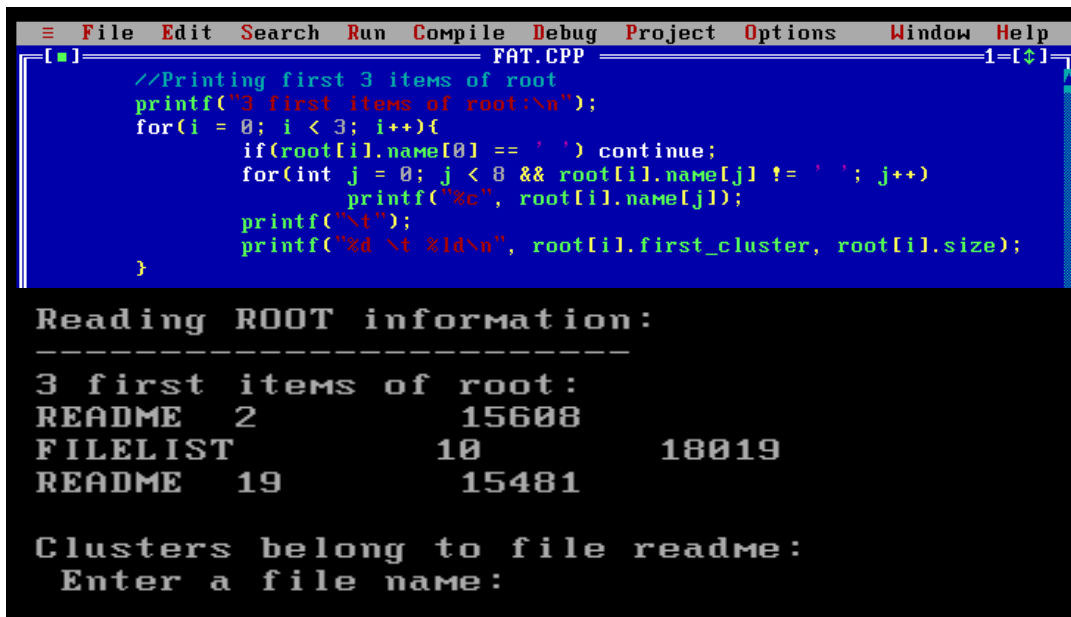
int num_sector = num_byte / boot.bytes_per_sector;
int root_begin = boot.reserved + boot.FAT_size * boot.FAT_cnt;

res = absread(drive, num_sector, root_begin, (void *)root);
if(res != 0){
    printf("\n Cannot read ROOT\n");
    return;
}

//Printing first 3 items of root

133:23
F1 Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu

```



```
File Edit Search Run Compile Debug Project Options Window Help
FAT.CPP
//Printing first 3 items of root
printf("3 first items of root:\n");
for(i = 0; i < 3; i++){
    if(root[i].name[0] == ' ') continue;
    for(int j = 0; j < 8 && root[i].name[j] != ' '; j++)
        printf("%c", root[i].name[j]);
    printf("\t");
    printf("%d \t %d\n", root[i].first_cluster, root[i].size);
}
```

Reading ROOT information:

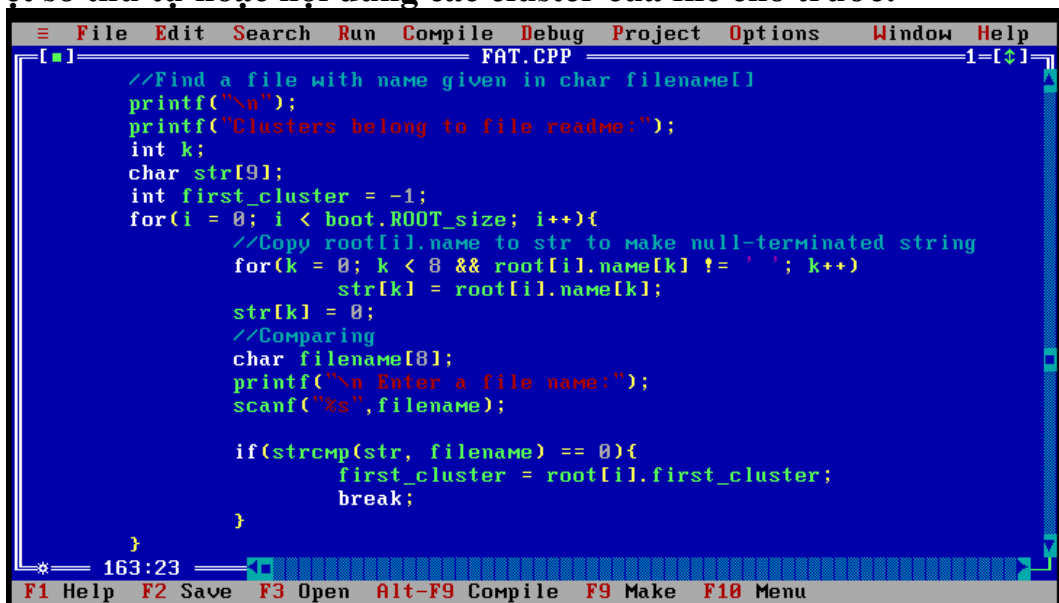
-----

3 first items of root:

README	2	15608
FILELIST	10	18019
README	19	15481

Clusters belong to file readme:  
Enter a file name:

#### 4. Duyệt số thứ tự hoặc nội dung các cluster của file cho trước:



```
File Edit Search Run Compile Debug Project Options Window Help
FAT.CPP
//Find a file with name given in char filename[]
printf("\n");
printf("Clusters belong to file readme:");
int k;
char str[9];
int first_cluster = -1;
for(i = 0; i < boot.ROOT_size; i++){
    //Copy root[i].name to str to make null-terminated string
    for(k = 0; k < 8 && root[i].name[k] != ' '; k++)
        str[k] = root[i].name[k];
    str[k] = 0;
    //Comparing
    char filename[8];
    printf("\n Enter a file name:");
    scanf("%s", filename);

    if(strcmp(str, filename) == 0){
        first_cluster = root[i].first_cluster;
        break;
    }
}
```

\* 163:23

F1 Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu

```

File Edit Search Run Compile Debug Project Options Window Help
FAT.CPP
//Printing clusters belonging to the file
if(first_cluster >= 0){
    cur = first_cluster;
    while(cur < 0xFFF8){
        printf("%a ", cur);
        cur = fat[cur];
    }
}

Clusters belong to file readme:
Enter a file name:README
2 3 4 5 6 7 8 9

```

5.Viết đoạn chương trình in ra nội dung giống như câu lệnh dir: Chương trình và kết quả được thực hiện như dưới đây:

```

File Edit Search Run Compile Debug Project Options Window Help
FAT.CPP
#include <stdio.h>
#include <dos.h>
#include <memory.h>
#include <stdlib.h>
#include <string.h>

struct BOOT { //for FAT16
    char jmp[3];
    char OEM[8];
    int bytes_per_sector;
    char sectors_per_cluster;
    int reserved;
    char FAT_cnt;
    int ROOT_size;
};

File Edit Search Run Compile Debug Project Options Window Help
FAT.CPP
char FAT_cnt;
int ROOT_size;
int total_sectors;
char media;
int FAT_size;
int sectors_per_track;
int head_cnt;
long hidden_sectors;
long total_sectors_long;
char unknown[3];
long serial;
char volume[11];
char FAT_type[8];
char loader[448];

File Edit Search Run Compile Debug Project Options Window Help
FAT.CPP
char loader[448];
char mark[2];
};

struct ROOT {
    char name[8];
    char ext[3];
    char attr;
    char reserved[10];
    char time[2];
    char date[2];
    int first_cluster;
    long size;
};

```



```
File Edit Search Run Compile Debug Project Options Window Help
FAT.CPP 1-[↑]
void main()
{
    int drive = 3; //A=0, B=1, C=2, D=3 ...

    //Reading boot sector from disk D
    BOOT boot;

    int res = absread(drive, 1, 0, &boot);
    if(res != 0){
        printf("Cannot read boot sector\n");
        return;
    }
    printf("Reading disk parameters\n");

File Edit Search Run Compile Debug Project Options Window Help
C:FAT.CPP 2-[↑]
    //Reading FAT16 table from disk D
    unsigned int *fat = (unsigned int *)malloc (boot.FAT_size * boot.bytes
    if (fat == NULL) {
        printf("Not enough memory\n");
        return;
    }
    printf("\n\n");
    printf("Reading FAT16 parameters\n");
    printf("-----\n");
    printf("FAT size: %d \n", boot.FAT_size);
    printf("Reserved: %d \n", boot.reserved);

    res = absread(drive, boot.FAT_size, boot.reserved, fat);
    if(res != 0){

File Edit Search Run Compile Debug Project Options Window Help
C:FAT.CPP 2-[↑]
        printf("Cannot read FAT\n");
        return;
    }
    //printing first 15 FAT cells
    printf("Content of first 15 FAT cells:");
    for (i = 0; i < 15; i++)
        printf("%u ", fat[i]);

    //Counting number of free clusters from first 100 clusters
    int free_count = 0;
    for (i = 2; i < 100; i++){ //first 2 clusters are not used
        if(fat[i] == 0) free_count++;
    }
    printf("\n");

File Edit Search Run Compile Debug Project Options Window Help
C:FAT.CPP 2-[↑]
    printf("\n");
    printf("Number of free clusters from first 100 clusters:");
    printf("%d\n", free_count);

    //Printing clusters of a file from cluster n
    unsigned int n = 5;
    unsigned int cur = n;
    printf("Clusters of a file from %u: ", n);
    while(cur < 0xFFFF){
        printf("%u ", cur);
        cur = fat[cur];
    }

    //Reading ROOT from disk D
```

```

File Edit Search Run Compile Debug Project Options Window Help
C:FAT.CPP 2-[↑]
res = absread(drive, num_sector, root_begin, (void *)root);
if(res != 0){
    printf("\n Cannot read ROOT\n");
    return;
}

//Printing first 3 items of root
printf("3 First items of root:\n");
for(i = 0; i < 3; i++){
    if(root[i].name[0] == ' ') continue;
    for(int j = 0; j < 8 && root[i].name[j] != ' '; j++)
        printf("%c", root[i].name[j]);
    printf("\n");
}

File Edit Search Run Compile Debug Project Options Window Help
C:FAT.CPP 2-[↑]
printf("\n");
printf("%d \t %d\n", root[i].first_cluster, root[i].size);
}

//Find a file with name given in char filename[]
printf("\n");
printf("Clusters belong to file readme:");
int k;
char str[9];
int first_cluster = -1;
for(i = 0; i < boot.ROOT_size; i++){
    //Copy root[i].name to str to make null-terminated string
    for(k = 0; k < 8 && root[i].name[k] != ' '; k++)
        str[k] = root[i].name[k];

File Edit Search Run Compile Debug Project Options Window Help
C:FAT.CPP 2-[↑]
        str[k] = root[i].name[k];
        str[k] = 0;
        //Comparing
        char filename[8];
        printf("\n Enter a file name:");
        scanf("%s",filename);

        if(strcmp(str, filename) == 0){
            first_cluster = root[i].first_cluster;
            break;
        }
}

File Edit Search Run Compile Debug Project Options Window Help
C:FAT.CPP 2-[↑]
//Printing clusters belonging to the file
if(first_cluster >= 0){
    cur = first_cluster;
    while(cur < 0xFFF8){
        printf("%u ", cur);
        cur = fat[cur];
    }

    free(root);
    free(fat);

    getchar();
}

File Edit Search Run Compile Debug Project Options Window Help
C:FAT.CPP 2-[↑]
}
printf("-----DIR-----\n");
int j;
for(i=0;i<10;i++) {
    for(j=0;j<8;j++) printf("%c",root[i].name[j]);printf("\n");
    for(j=0;j<3;j++) printf("%c",root[i].ext[j]);printf("\n");
    char n[20];
    sprintf(n,"%d",root[i].size);
    if(strlen(n)%3==0) {
        for(j=0;j<strlen(n);j++)
            if(j!=0&&j!=strlen(n)-1&&j%3==2) printf("%c",n[j]);
            else printf("%c",n[j]);
    }
    else if(strlen(n)%3==1){

```

```

File Edit Search Run Compile Debug Project Options Window Help
C:FAT.CPP
else if(strlen(n)%3==1){
    for(j=0;j<strlen(n);j++)
        if(j!=strlen(n)-1&&j%3==0) printf("%c",n[j]);
        else printf("%c",n[j]);
}
else for(j=0;j<strlen(n);j++)
    if(j!=strlen(n)&&j%3==1) printf("%c",n[j]);
    else printf("%c",n[j]);
printf("\n");
char s[16],s1[8],s2[8];
getbit(root[i].date[0],s1);
getbit(root[i].date[1],s2);
for(j=0;j<8;j++) s[j]=s1[j],s[j+8]=s2[j];
int ngay=convert(s,0,4);

File Edit Search Run Compile Debug Project Options Window Help
C:FAT.CPP
int ngay=convert(s,0,4);
int thang=convert(s,5,8);
int nam=convert(s,9,15); nam+=1980;
if(thang<10) printf("0%d-",thang); else printf("%d-",thang);
if(ngay<10) printf("0%d",ngay); else printf("%d",ngay);
printf("-%d\t",nam%100);

getbit(root[i].time[0],s1);
getbit(root[i].time[1],s2);
for(j=0;j<8;j++) s[j]=s1[j],s[j+8]=s2[j];
int gio=convert(s,11,15);
int phut=convert(s,5,10);
if(gio<12) printf("%d:",gio);else printf("%d:",gio-12);
if(phut<10) printf("0%d:",phut); else printf("%d:",phut);

```

## 6.Kết quả:

```

-----
FAT size: 33
Reserved: 6
Content of first 15 FAT cells:65528 65535 3 4 5 6 7 8 9 65535 11 12 13 14 15
Number of free clusters from first 100 clusters:61
Clusters of a file from 5: ->5->6->7->8->9

Reading ROOT information:
-----
3 first items of root:
README 2      15608
FILELIST 10    18019
README 19     15481
-----DIR:-----
README      TXT      15,608, 05-11-98      4:33a
FILELIST    DOC      18,019, 02-01-10      7:37a
README      COM      15,481, 02-01-10      7:37a
README      COM      4,217, 02-01-10      7:37a
FAT          OBJ      3,743, 10-09-10      7:47a
FAT          EXE      12,803, 10-09-10      7:47a
              0        00-00-00      0:00a
              0        00-00-00      0:00a
              0        00-00-00      0:00a
              0        00-00-00      0:00a

```

