```c
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>

typedef uint8_t BYTE;

int main(int argc, char *argv[])
{
    // Verifica se foi digitado 2 argumentos na execução do programa;
    if(argc != 2)
    {
        printf("Usage: ./recover image\n");
        return 1;
    }

    // Salva o argumento em uma variável "FILE" para leitura "r";
    FILE *input = fopen(argv[1], "r");

    // Se o arquivo não existir, retorna erro;
    if (input == NULL)
    {
        printf("File not found\n");
        return 2;
    }

    BYTE buffer[512]; // Variável para armazenar os dados lidos;
    char *filename = malloc(sizeof(BYTE)); // Cria um ponteiro para o
espaço liberado pela função malloc;
    FILE *output = NULL; // Cria um ponteiro para um arquivo ainda não
existente;
    int countfiles = 0; // Variável para controlar qual arquivo estamos;

    // Enquanto estiver lendo o arquivo, continue;
    // fread("variável de armazenamento", "tamanho do que estamos lendo",
"Quantas vezes vamos repetir","Arquivo");
    while(fread(buffer, sizeof(char), 512, input))
    {
        // Se os 4 primeiros digitos forem 0xff, 0xd8, 0xff e >= 0xe0 e
<= 0xef;
        if(buffer[0] == 0xff && buffer[1] == 0xd8 && buffer[2] == 0xff &&
(buffer[3] & 0xf0) == 0xe0)
        {
            // Preencha a string no espaço alocado por malloc com um nome
###.jpg\0 (8 dígitos) sendo ### o número do arquivo;
            sprintf(filename, "%03i.jpg", countfiles);+

            // Abra um arquivo com o nome "filename", para escrever "w";
            output = fopen(filename, "w");
```

```
            // Adiciona +1 ao contador de arquivos;
            countfiles++;
        }

        // Se Ponteiro estiver ligado a algo, ou seja, for diferente de
nulo;
        if(output != NULL)
        {
            // Escreva nesse arquivo;
            // fwrite("Variável de armazenamento", "Tamanho do que vamos
escrever", "Quantas vezes vamos repetir", "Arquivo");
            fwrite(buffer, sizeof(BYTE), 512, output);
        }
    }

    // Libera o espaço alocado pelo malloc;
    free(filename);

    // Fecha os programas abertos.
    fclose(output);
    fclose(input);

    return 0;
}
```