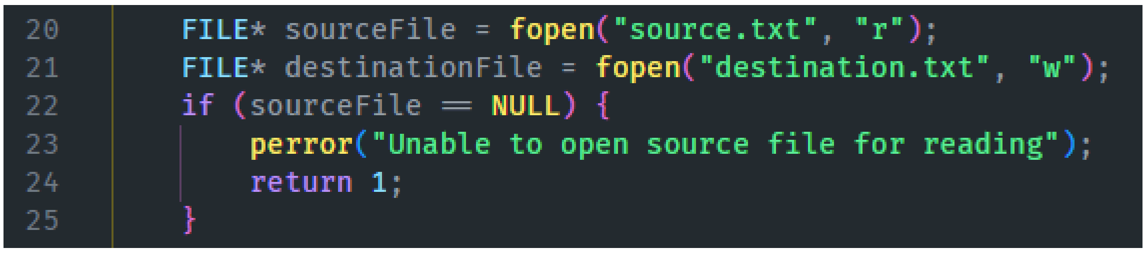
Project 9 File System Operations

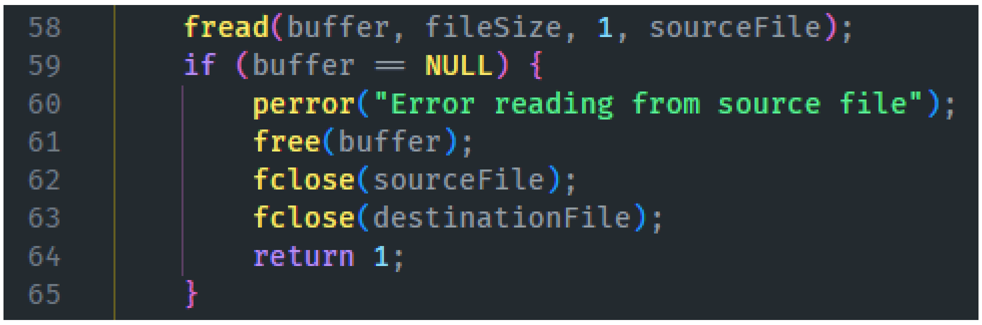
110511010 楊育陞

Part 1



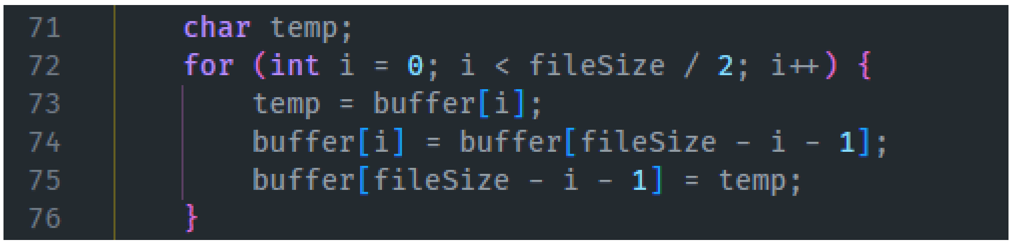
```c
    FILE* sourceFile = fopen("source.txt", "r");
    FILE* destinationFile = fopen("destination.txt", "w");
    if (sourceFile == NULL) {
        perror("Unable to open source file for reading");
        return 1;
    }
```

First I open the source file for reading and destination file for writing by specifying the file name and the mode. And I check if the file is opened successfully by checking if the file pointer is NULL.



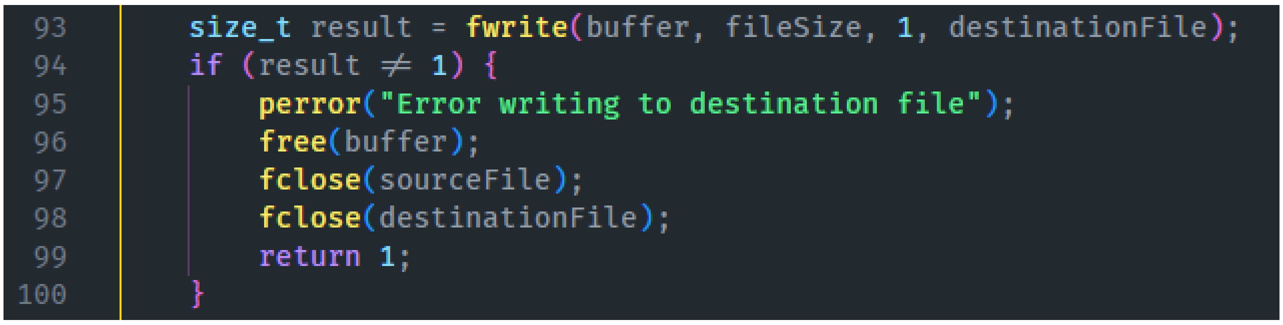
```c
    fread(buffer, fileSize, 1, sourceFile);
    if (buffer == NULL) {
        perror("Error reading from source file");
        free(buffer);
        fclose(sourceFile);
        fclose(destinationFile);
        return 1;
    }
```

Then I read the contents of the source file into the buffer by using `fread()` function. And I check if the file is read successfully by checking if buffer pointer is NULL.



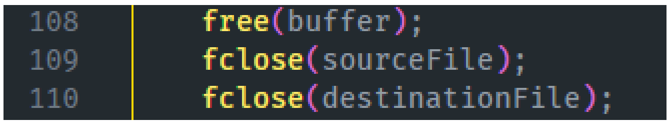
```c
    char temp;
    for (int i = 0; i < fileSize / 2; i++) {
        temp = buffer[i];
        buffer[i] = buffer[fileSize - i - 1];
        buffer[fileSize - i - 1] = temp;
    }
```

I use a for loop to swap to reverse the contents of the buffer.



```c
    size_t result = fwrite(buffer, fileSize, 1, destinationFile);
    if (result != 1) {
        perror("Error writing to destination file");
        free(buffer);
        fclose(sourceFile);
        fclose(destinationFile);
        return 1;
    }
```

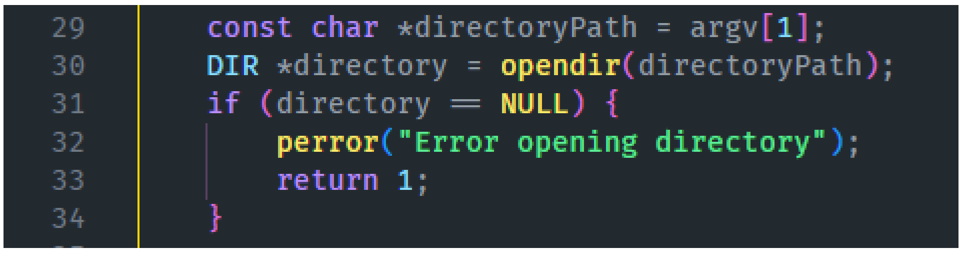
After reversing the buffer, I use `fwrite()` function to write the reversed buffer to the destination file. And I check if the file is written successfully by checking if the return value of `fwrite()` is equal to the size of the buffer.



```c
    free(buffer);
    fclose(sourceFile);
    fclose(destinationFile);
```

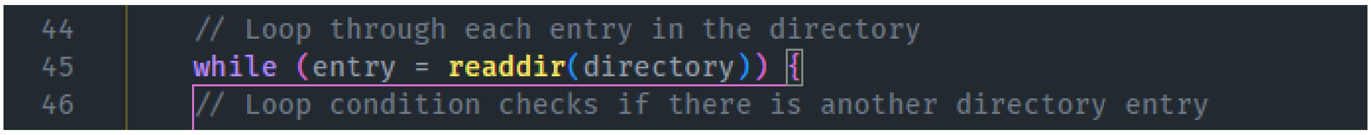
I use `free()` function to free the memory allocated to the buffer. And I use `fclose()` function to close the source file and destination file.

Part 2



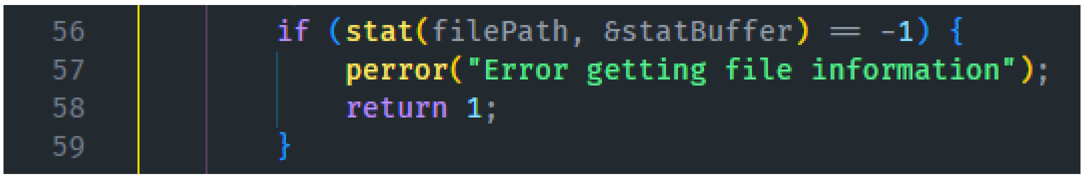
```c
    const char *directoryPath = argv[1];
    DIR *directory = opendir(directoryPath);
    if (directory == NULL) {
        perror("Error opening directory");
        return 1;
    }
```

First I open the directory specified by the command line argument by using `opendir()` function. And I check if the directory is opened successfully by checking if the directory pointer is NULL.



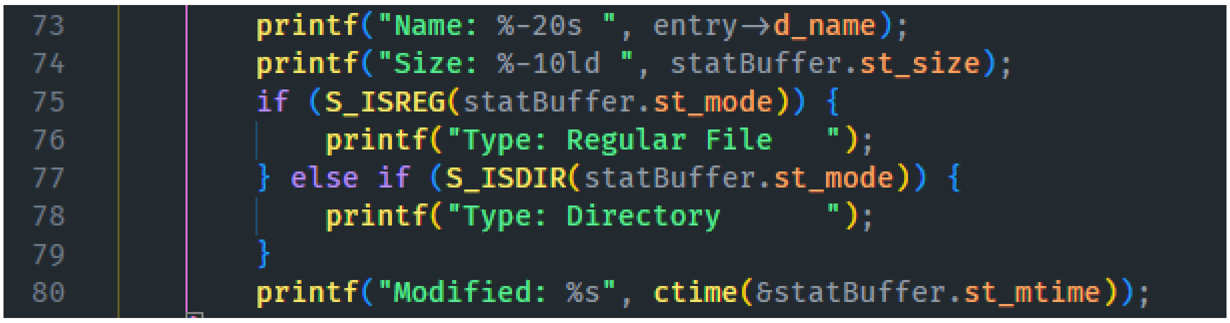
```c
    // Loop through each entry in the directory
    while (entry = readdir(directory)) {
    // Loop condition checks if there is another directory entry
```

Then I loop through each entry in the directory by using `readdir()` function in a while loop.



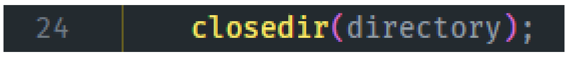
```c
        if (stat(filePath, &statBuffer) == -1) {
            perror("Error getting file information");
            return 1;
        }
```

Then I store the information of the entry in the `statBuffer` by using `stat()` function. And I check if the information is stored successfully by checking if the return value of `stat()` is equal to -1.



```c
    printf("Name: %-20s ", entry->d_name);
    printf("Size: %-10ld ", statBuffer.st_size);
    if (S_ISREG(statBuffer.st_mode)) {
        printf("Type: Regular File   ");
    } else if (S_ISDIR(statBuffer.st_mode)) {
        printf("Type: Directory      ");
    }
    printf("Modified: %s", ctime(&statBuffer.st_mtime));
```

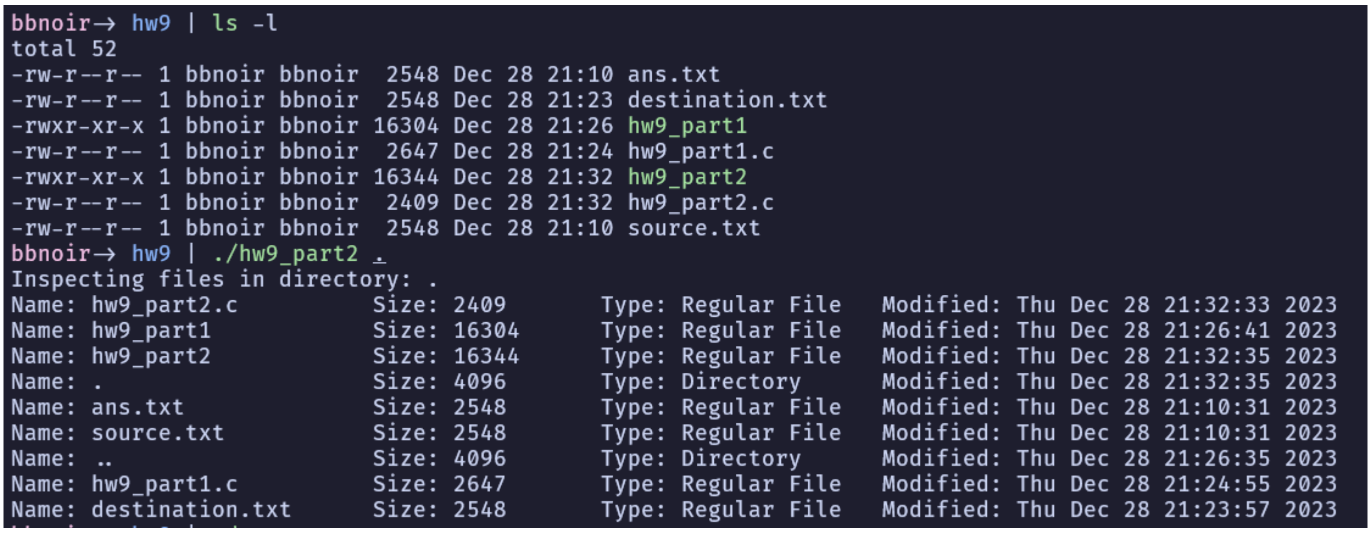
Then I print the information of the entry. To print the file name, I use `d_name` in the `entry`. To print the file size, I use `st_size` in the `statBuffer`. To print the file type, I use `S_ISREG()` function to check if the file is a regular file and use `S_ISDIR()` function to check if the file is a directory. To print the modification time, I use `st_mtime` in the `statBuffer` and use `ctime()` function to convert the modification time to a string.



```c
    closedir(directory);
```

Finally, I use `closedir()` function to close the directory.

screenshot of test result:



```
bbnoir-> hw9 | ls -l
total 52
-rw-r--r-- 1 bbnoir bbnoir  2548 Dec 28 21:10 ans.txt
-rw-r--r-- 1 bbnoir bbnoir  2548 Dec 28 21:23 destination.txt
-rwxr-xr-x 1 bbnoir bbnoir 16304 Dec 28 21:26 hw9_part1
-rw-r--r-- 1 bbnoir bbnoir  2647 Dec 28 21:24 hw9_part1.c
-rwxr-xr-x 1 bbnoir bbnoir 16344 Dec 28 21:32 hw9_part2
-rw-r--r-- 1 bbnoir bbnoir  2409 Dec 28 21:32 hw9_part2.c
-rw-r--r-- 1 bbnoir bbnoir  2548 Dec 28 21:10 source.txt
bbnoir-> hw9 | ./hw9_part2 .
Inspecting files in directory: .
Name: hw9_part2.c          Size: 2409       Type: Regular File   Modified: Thu Dec 28 21:32:33 2023
Name: hw9_part1            Size: 16304      Type: Regular File   Modified: Thu Dec 28 21:26:41 2023
Name: hw9_part2            Size: 16344      Type: Regular File   Modified: Thu Dec 28 21:32:35 2023
Name: .                    Size: 4096       Type: Directory      Modified: Thu Dec 28 21:32:35 2023
Name: ans.txt              Size: 2548       Type: Regular File   Modified: Thu Dec 28 21:10:31 2023
Name: source.txt           Size: 2548       Type: Regular File   Modified: Thu Dec 28 21:10:31 2023
Name: ..                   Size: 4096       Type: Directory      Modified: Thu Dec 28 21:26:35 2023
Name: hw9_part1.c          Size: 2647       Type: Regular File   Modified: Thu Dec 28 21:24:55 2023
Name: destination.txt      Size: 2548       Type: Regular File   Modified: Thu Dec 28 21:23:57 2023
```