Assignment Number

Problem Statement

Program in C to copy the contents on one file to another specified by the user at command line

Theory

To copy all the contents of one file to the other, each byte of the existing file have to be copied to the new file. The smallest unit of data that can be transferred in C is one byte, and to represent one byte of data, the character datatype of C is used. A variable of character datatype holds 8 bit or 1 byte of data. At first, the exisiting file is opened in read only mode and the new file is created in write mode. Then, at every iteration until the existing file is exhausted, one byte is read from the existing file and is written to the new file.

Algorithm

Input :

1. The path of the file to read the content from, say OldFile
2. The path of the file to write the content to, say NewFile

Output : The content of OldFile successfully copied to the NewFile, or suitable unsuccessful message

Steps :

Step 1 : Set FileRead = FileOpen(OldFile, Write)
// FileOpen is a function that opens the file at the argument
// path in the specified mode
Step 2 : If(FileRead = Null)
Then
a) Print “Unable to open the file for reading”
b) Exit
[End of if structure]
Step 3 : Set FileWrite = FileOpen(NewFile, Read)
Step 4 : If(FileWrite = Null)
Then
a) Print “Unable to open the file for writing”
b) Exit
[End of if structure]
Step 5 : Repeat through step 5.a to 5.b while FileRead is not exhausted
a) Set ChRead = ReadCharacter(FileRead)
// ReadCharacer is a procedure that reads the next character
// from the specified file
b) WriteCharacter(FileWrite, ChRead)
// WriteCharacter is a procedure that writes one character
// to the specified file
Step 6 : FileClose(FileRead)
// FileClose is a procedure that releases all handles to
// the given argument file
Step 7 : FileClose(FileWrite)

Source Code

```
#include <stdio.h>

int main(int argc, char *argv[]){
   FILE *fileRead, *fileWrite;
   char ch;
   if(argc != 3){
      printf("\nWrong number of arguments!\n");
      return 1;
   }
   fileRead = fopen(argv[1], "r");
   if(fileRead == NULL){
      printf("\nUnable to open file %s for reading!\n", argv[1]);
      return 2;
   }
   fileWrite = fopen(argv[2], "w");
   if(fileWrite == NULL){
      printf("\nUnable to open file %s for writing!\n", argv[2]);
      fclose(fileRead);
      return 2;
   }
   printf("\nCopy started..");
   while(!feof(fileRead)){
      fscanf(fileRead, "%c", &ch);
      fprintf(fileWrite, "%c", ch);
   }
   fclose(fileRead);
   fclose(fileWrite);
   printf("\nCopied contents of file '%s' to file '%s'..\n", argv[1], argv[2]);
   return 0;
}
```

Input and Output

Contents of input file (argument 1)	Invocation of the program	Output of the program	Content of output file (argument 2)
This is the file to be copied	./a.out copy1 copy2	Copy started.. Copied contents of file 'copy1' to file 'copy2'..	This is the file to be copied
A quick brown fox jumps over the lazy dog	./a.out readfile.txt writefile.txt	Copy started.. Copied contents of file ‘readfile.txt’ to ‘writefile.txt’	A quick brown fox jumps over the lazy dog

Discussion

1. There is no test to see if there is enough space in the disk where a new file will be created and the existing file will be copied.
2. Even if the existing file can be opened, the copy operation might not finish successfully because of an IO error.
3. This copy operation is pretty inefficient for large files as only one byte is copied at a time. A more effiecient solution will include copying large chunks of data at once between the files.