Instructions : Clone your GitHub Classroom repository for this assignment. Follow all steps below to complete the programming assignment portion of Homework 4. Push your changes to GitHub and check that all tests are passing in Actions.

After completing each portion of the assignment, compile your code with the command *sh compile.sh* and check correctness with *make test* from within the *build* directory. Note, if your local computer uses an alternative to *make* you will need to compile manually. **Do not change compile.sh as this could cause GitHub Actions to fail.**

If you get confused with GitHub/CMake logistics, refer back to Homework 0 here.

1 Write to Journal

For this portion of the homework assignment, you will complete the method `write_to_journal()`. This method is passed the following:

Listing 1: Arguments

```
int txb_bytes; // number of bytes in variable txb
char* txb;     // transaction begin block of size txb_bytes
int ibytes;    // number of bytes in variable i
char* i;       // inode block of size ibytes
int bbytes;    // number of bytes in variable b
char* b;       // bitmap block of size bbytes
int Db_bytes;  // number of bytes in variable Db
char* Db;      // data block of size Db_bytes
int txe_bytes; // number of bytes in variable txe
char* txe;     // transaction end block of size txe bytes
```

Edit this method to write all data to the file 'journal.txt'. The data should be written to this file in the following order. Make sure to close the file before the method returns.

1. char* txb
2. char* i
3. char* b
4. char* Db
5. char* txe

Hints :

- The method open(...) should be used to open the file, returning a file descriptor
- When creating a file, you will need to pass permissions as the third argument (i.e. 0600)
- The method write(...) takes the file descriptor, size, and data to write to the file
- After writing to the file, be sure to call fsync() to force write to disk
- The method close(...) closes the file descriptor

-

2 Checkpointing

For this portion of the homework assignment, you will complete the method `checkpoint(...)`, which is passed the following parameters:

Listing 2: Arguments

```
int txe_bytes; // number of bytes in variable txe
char* txe;     // transaction end block of size txe bytes
```

Assume all data has been previously written to the journal. Your task is to now copy this data to its final location. This can be done with the following steps.

1. Open the file journal.txt and make sure that all data has been written to the journal
2. If the journal is incomplete, close the file and return -1
3. Otherwise, create and open the file data.txt. Copy all data from journal.txt to data.txt.
4. Return 0 if checkpointing completes.

Hints:

- The method open(...) will open a file and return a file descriptor.
- When creating a file, you will need to pass permissions as the third argument (i.e. 0600)
- The method lseek(...) can be used to seek to x bytes before the file with lseek(fd, -x, SEEK END)
- The method lseek(...) can also seek to the beginning of the file with SEEK SET
- After writing the the file, be sure to call fsync() to force write to disk
- Make sure to close any open file descriptors before returning from the method.

3 Check for Correctness

To check that your code is working, do the following:

1. Make sure all tests are passing locally

```
sh compile.sh
cd build
make test
```

2. Push all changes to GitHub
3. Check that your GitHub actions are passing

The End.