CS 333-LAB QUIZ 4
140050048

Part-A

1.

Not much difference is observed.

Only that fgetattr is called after reading once more in case of mmap.c as it is required for unmapping the file.

Also the flags in the bb_open and bb_read calls are different. 0X00008000 in **disk.c** and 0x00008002 in **mmap.c**.

Using **disk.c:**

```
bb_getattr(path="/one-kb-file.txt", statbuf=0xd6c8cc20)
    bb_fullpath:  rootdir = "/opt/labuser/fuse-tutorial-2016-03-25/example/rootdir", path = "/one-kb-file.txt", fpath = "/opt/labuser/fuse-tutorial-2016-03-25/example/rootdir/one-kb-file.txt"
    lstat returned 0
    si:
    st_dev = 64512
    st_ino = 1453032
    st_mode = 0100664
    st_nlink = 1
    st_uid = 999
    st_gid = 999
    st_rdev = 0
    st_size = 1024
    st_blksize = 4096
    st_blocks = 8
    st_atime = 0x5811bf0c
    st_mtime = 0x5811bf0b
    st_ctime = 0x5811bf0b

bb_open(path"/one-kb-file.txt", fi=0xd748dd10)
    bb_fullpath:  rootdir = "/opt/labuser/fuse-tutorial-2016-03-25/example/rootdir", path = "/one-kb-file.txt", fpath = "/opt/labuser/fuse-tutorial-2016-03-25/example/rootdir/one-kb-file.txt"
    open returned 6
    fi:
    flags = 0x00008000
    fh_old = 0x00000000
    writepage = 0
    direct_io = 0
    keep_cache = 0
    fh = 0x0000000000000006
    lock_owner = 0x0000000000000000

bb_read(path="/one-kb-file.txt", buf=0xcc000b70, size=4096, offset=0, fi=0xd648bd10)
    fi:
    flags = 0x00008000
    fh_old = 0x00000006
    writepage = 0
    direct_io = 0
    keep_cache = 0
    fh = 0x0000000000000006
    lock_owner = 0x0000000000000000
    pread returned 1024

bb_fgetattr(path="/one-kb-file.txt", statbuf=0xd6c8cbf0, fi=0xd6c8cd10)
    fi:
    flags = 0x00000000
    fh_old = 0x00000006
    writepage = 0
    direct_io = 0
    keep_cache = 0
    fh = 0x0000000000000006
    lock_owner = 0x0000000000000000
    si:
    st_dev = 64512
    st_ino = 1453032
    st_mode = 0100664
    st_nlink = 1
    st_uid = 999
    st_gid = 999
    st_rdev = 0
    st_size = 1024
    st_blksize = 4096
    st_blocks = 8
    st_atime = 0x5811bf0c
    st_mtime = 0x5811bf0b
    st_ctime = 0x5811bf0b

bb_flush(path="/one-kb-file.txt", fi=0xd748dd10)
    fi:
    flags = 0x00000000
    fh_old = 0x00000006
    writepage = 0
    direct_io = 0
    keep_cache = 0
    fh = 0x0000000000000006
    lock_owner = 0x0f17a42b6fb7b5ea

bb_release(path="/one-kb-file.txt", fi=0xd648bd10)
    fi:
    flags = 0x00008000
    fh_old = 0x00000006
    writepage = 0
    direct_io = 0
    keep_cache = 0
    fh = 0x0000000000000006
    lock_owner = 0x0000000000000000
    close returned 0
```

Using **mmap.c:**

```
bb_getattr(path="/one-kb-file.txt", statbuf=0xd648bc20)
    bb_fullpath:  rootdir = "/opt/labuser/fuse-tutorial-2016-03-25/example/rootdir", path = "/one-kb-file.txt", fpath = "/opt/labuser/fuse-tutorial-2016-03-25/example/rootdir/one-kb-file.txt"
    lstat returned 0
    si:
    st_dev = 64512
    st_ino = 1453032
    st_mode = 0100664
    st_nlink = 1
    st_uid = 999
    st_gid = 999
    st_rdev = 0
    st_size = 1024
    st_blksize = 4096
    st_blocks = 8
    st_atime = 0x5811bf0c
    st_mtime = 0x5811bf0b
    st_ctime = 0x5811bf0b
```

```
bb_open(path"/one-kb-file.txt", fi=0xd6c8cd10)
    bb_fullpath:  rootdir = "/opt/labuser/fuse-tutorial-2016-03-25/example/rootdir", path = "/one-kb-file.txt", fpath = "/opt/labuser/fuse-tutorial-2016-03-25/example/rootdir/one-kb-file.txt"
    open returned 6
    fi:
    flags = 0x00008002
    fh_old = 0x00000000
    writepage = 0
    direct_io = 0
    keep_cache = 0
    fh = 0x0000000000000006
    lock_owner = 0x0000000000000000

bb_read(path="/one-kb-file.txt", buf=0xd0000c90, size=4096, offset=0, fi=0xd748dd10)
    fi:
    flags = 0x00008002
    fh_old = 0x00000006
    writepage = 0
    direct_io = 0
    keep_cache = 0
    fh = 0x0000000000000006
    lock_owner = 0x0000000000000000
    pread returned 1024

bb_flush(path="/one-kb-file.txt", fi=0xd648bd10)
    fi:
    flags = 0x00000000
    fh_old = 0x00000006
    writepage = 0
    direct_io = 0
    keep_cache = 0
    fh = 0x0000000000000006
    lock_owner = 0x63638e6974267985

bb_release(path="/one-kb-file.txt", fi=0xd6c8cd10)
    fi:
    flags = 0x00008002
    fh_old = 0x00000006
    writepage = 0
    direct_io = 0
    keep_cache = 0
    fh = 0x0000000000000006
    lock_owner = 0x0000000000000000
    close returned 0
```

2.
When the file size is large significant difference is observed.
The differences in the above case are also observed.
In case of reading from disk, bb_read is repeatedly called to read from each block where the read requests gradually increase from 4kb to 128 kb.
Where as in case of reading form mmapped file, the requests are consistently of size 128 kb which corresponds to memory mapping.
Moreover mmapped file took more time as in both cases almost same number of fetches are required from memory and memory mapping has overhead.

3.Not much difference except for the flags as the size is less than 4 KB.

4.
In disk.c write is called for every 4KB block where as in mmapped case write is called on larger

sizes of 128 kb fewer times and after the mapped area is exhausted again read is done to bring new data into memory and then write continues.
Memory mapped case is faster by 25% time.

PART-B:

The file server.c will be compiled as

```
gcc server.c -o server
```

and will be run as

```
./server 5000
```

5000 is fixed.

bbfs.c is compiled normally using the makefile.