**Trent University: Operating Systems (COIS3320)**

**Dr. Bin Guo**

**Lab 9: Memory Mapped Files**

**Outline**

In this lab you are going to study Memory Mapping files in C.

You are to write a C program that opens a binary file named **numbers.bin** and maps it to a

memory region using the system call **mmap()**.The ‘numbers.bin’ contains **ten** 4-byte integers in binary format.

After memory mapping this file you are to read the contents from this memory mapped

region; that is, read one integer at a time, and copy it to an integer array (names

intArray) using the **memcpy()**function. Finally, you are to loop through the intArray

array to add all the numbers of the array and output the sum to the console.

In particular, your program needs to do the following:

1. To be able to use mmap() system call and the memcpy() function you need to add the

below header files in addition to your standard input output header file.

#include <sys/mman.h> /\*For mmap() function\*/

#include <string.h> /\*For memcpy function\*/

2. Other useful header files are listed below. They enable you to use the **open()**system call

which is used to open a new file and obtain its file descriptor.

#include <fcntl.h> /\*For file descriptors\*/

#include <stdlib.h> /\*For file descriptors\*/

3. Define global variables for the integer array and a signed character pointer to store the

starting address of the memory mapped file. E.g.:

int intArray[MEMORY\_SIZE]; int intArray[INT\_COUNT];

signed char \*mmapfptr;

You can define ‘MEMORY\_SIZE’ as a macro definition. It is the total number of bytes you

will be copying from numbers.bin file. Sample code:

#define INT\_SIZE 4 // Size of integer in bytes

#define INT\_COUNT 10 #define MEMORY\_SIZE INT\_COUNT \* INT\_SIZE

4. Open the file (**numbers.bin**) using the **open()** system call. Since you will be simply reading this file use the **O\_RDONLY** option. E.g.:

int mmapfile\_fd = open(“numbers.bin”, O\_RDONLY);

5. Use the **mmap()** system call to memory map this file. E.g.:

mmapfptr = mmap(0, MEMORY\_SIZE, PROT\_READ, MAP\_PRIVATE,

mmapfile\_fd, 0);

6. Retrieve the contents of the memory mapped file (using a loop) and store it in the integer

array using **memcpy()** function. Sample code to use memcpy() is:

memcpy(intArray + i, mmapfptr + 4\*i, INT\_SIZE);

INT\_SIZE = the size of the contents in bytes to be copied from the memory mapped file to

**intArray**. Since we are reading only 4 bytes (size of an integer) at a time, INT\_SIZE = 4.

7. Unmap the memory mapped file using the **unmap()** system call. E.g.:

munmap(mmapfptr, MEMORY\_SIZE);

8. Loop through intArray to add all numbers in the array and output this sum to the console.

9. Compile your program without errors.

10. Show the program output to your TA.

-------------------------------------------------------------------------------------------------------------------------

Sample Program output:

**./lab9**

Sum of numbers = 92