CS5243 Advanced UNIX Programming
Assignment 2 (4 pts)
Group 4

Screenshot of codes:

assignment3.c

```c
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>

struct CustomStream
{
    FILE *file;      // output file data
    char *buffer;    // Data buffer
    size_t size;     // Total size of the data buffer
    size_t position; // Current position in the buffer
};

int read_from_memory(void *cookie, char *buf, int size)
{
    // printf("Read\n");
    // printf("buf: %s\n", buf);
    // printf("size: %d\n", size);
    // printf("cookie: %s\n", cookie);
    // int tmp = fread(&buf, 1, 12, cookie);
    // printf("c: %c\n", fgetc(cookie));
    // printf("tmp: %d\n", tmp);
    return fread(buf, 1, size, cookie);
}

int write_to_memory(void *cookie, const char *buf, int size)
{
    // printf("Write\n"); // for checking
    // printf("%s\n", buf); // for checking
    // printf("%d\n", size); // for checking
    return fwrite(buf, 1, size, cookie);
}

int seek_memory(void *cookie, long offset, int whence)
{
    // printf("Seek\n");
    // printf("%ld\n", offset);
    FILE *memfile = (FILE *)cookie;
    return fseek(memfile, (long)offset, whence);
}

int close_memory(void *cookie)
{
    // Implement closing your memory buffer here.
    // printf("Close\n");
    fclose(cookie); // close the file
    return 0;
}
```

```c

int main()
{
    size_t numwritten;
    char input_data[12] = "hello, world"; // len = 12
    char new_buffer[1024];

    FILE *file = fmemopen(new_buffer, 1024, "rw");
    FILE *customStream = funopen(file, read_from_memory, write_to_memory, seek_memory, close_memory);

    size_t bytesWrite = write_to_memory(file, input_data , 12);
    // printf("bytesWrite: %zu\n", bytesWrite);

    char buffer[1024] = "";

    // Print "world"
    int seekPos = seek_memory(file, 7, SEEK_SET);
    // printf("seekPos: %d\n", seekPos);

    size_t bytesRead = read_from_memory(file, buffer, 5);
    // printf("%zu\n", bytesRead);
    printf("%s\n", buffer);

    // Print "hello, world"
    seekPos = seek_memory(file, 0, SEEK_SET);
    // printf("seekPos: %d\n", seekPos);

    bytesRead = read_from_memory(file, buffer, 12);
    // printf("%zu\n", bytesRead);
    printf("%s\n", buffer);

    fclose(customStream); // trigger the close_to_memory function
    return 0;
}
```

NORMAL ⚠ 1 assignment3.c

:37

Screenshot of result:

```
freebsd@generic:~/Advanced-UNIX-Programming_Student/assignment3 % make clean && make
rm -f assignment3.o assignment3
gcc  -O2 -pipe -c assignment3.c -o assignment3.o
assignment3.c: In function 'main':
assignment3.c:59:75: warning: passing argument 4 of 'funopen' from incompatible pointer type [-Wincompat
ible-pointer-types]
   59 |     FILE *customStream = funopen(file, read_from_memory, write_to_memory, seek_memory, close_mem
ory);
      |                                                                           ^~~~~~~~~~
      |                                                                           |
      |                                                                           int (*)(void *, long i
nt,  int)
In file included from assignment3.c:5:
/usr/include/stdio.h:414:13: note: expected 'fpos_t (*)(void *, fpos_t,  int)' {aka 'long int (*)(void *
, long int,  int)'} but argument is of type 'int (*)(void *, long int,  int)'
  414 |             fpos_t (* _Nullable)(void *, fpos_t, int),
      |             ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
gcc -std=c11 -O2 -Wall -o assignment3 assignment3.o
freebsd@generic:~/Advanced-UNIX-Programming_Student/assignment3 % ./assignment3
world
hello, world
```

1. We first declared a struct called CustomStream to initialize a file stream.

2. We then declare functions and call **fread, fwrite, fseek,** and **fclose** functions to implement the read, write, seek, and close functions.

3. In main function, we use the **fmemopen** function to create a memory stream associated with **new_buffer**. Function **funopen** to create a custom stream (**customStream**) associated with the **file** stream.

4. **size_t bytesWrite** writes 12 characters from **input_data** to the custom memory stream.

5. **int seekPos** moves the file position indicator to the 7^{th} byte from the beginning of the custom stream using **seek_memory** function.

6. size_t bytesRead reads 5 characters from the custom stream into **buffer**, and we are able to print *world*.

7. After printing *world*, **seekPos** moves the file position indicator back to the beginning of the custom stream, and we use **bytesRead** to read characters again and print *hello, world*.

8. Finally, we use **fclose** to close the memory stream.