Homework #9 **M1522.000800 System Programming**

Sample Solution

Question 1

I/O Redirection

Assumed that we are supposed to implement I/O redirection program like `$ ls > foo.txt`. Write a C program '`redirector.c`' that switch the standard output file descriptor of some program to the given output file. Use `execl()` to execute *`hello`* in *`redirector`*.

```
/*
 * redirector.c
 */
#include <fcntl.h>
#include <unistd.h>

int main(int argc, char *argv[]) {
    /* Write Your Own Code */
    int fd;
    char *pname = argv[1];
    char *ofname = argv[2];

    fd = open(ofname, O_CREAT|O_TRUNC|O_WRONLY, S_IRUSR|S_IWUSR);
    dup2(fd, 1);
    execl(pname, pname, NULL);

    return 0;
}
```

```
/*
 * hello.c
 */
#include <stdio.h>

int main() {
    printf("hello world");
}
```

< Expected Result >

```
$ gcc hello.c -o hello
$ gcc redirector.c -o redirector
$ ./redirector hello output.txt
$ cat output.txt
hello world
```

Question 2

How Processes Share Files

What would the following program print for result?

```
/*
 * question2.c
 */
#include <fcntl.h>
#include <unistd.h>

int main(int argc, char *argv[]) {
    int fd1;
    char c1, c2;
    char *fname = argv[1];
    fd1 = open(fname, O_RDONLY, 0);
    read(fd1, &c1, 1);
    if (fork()) { /* Parent */
        read(fd1, &c2, 1);
        printf("Parent: c1 = %c, c2 = %c\n", c1, c2);
    } else { /* Child */
        sleep(1);
        read(fd1, &c2, 1);
        printf("Child: c1 = %c, c2 = %c\n", c1, c2);
    }
    return 0;
}
```

< Result >

```
$ echo "abcde" > input.txt
$ gcc question2.c -o question2
$ ./question2 input.txt
Parent: c1 = a, c2 = b
Child: c1 = a, c2 = c
```