CL 2006 Operating Systems Lab

Time: 20 minutes **Quiz 2** **Marks: 15**

Name: ______________________________ **Roll #:** __________________

Question 1 [4 Marks]

List all legal outputs the below program may produce when executed on a Ubuntu system. The output consists of strings made up of multiple letters.

```
#include<unistd.h>
#include<sys/wait.h>

//W(A) means write(1,"A”,sizeof("A"))
//W(x) means write(1,#x,sizeof #x)

int main() {
    W(A);
    pid_t child = fork();
    wait(NULL);
    W(X);
}
```

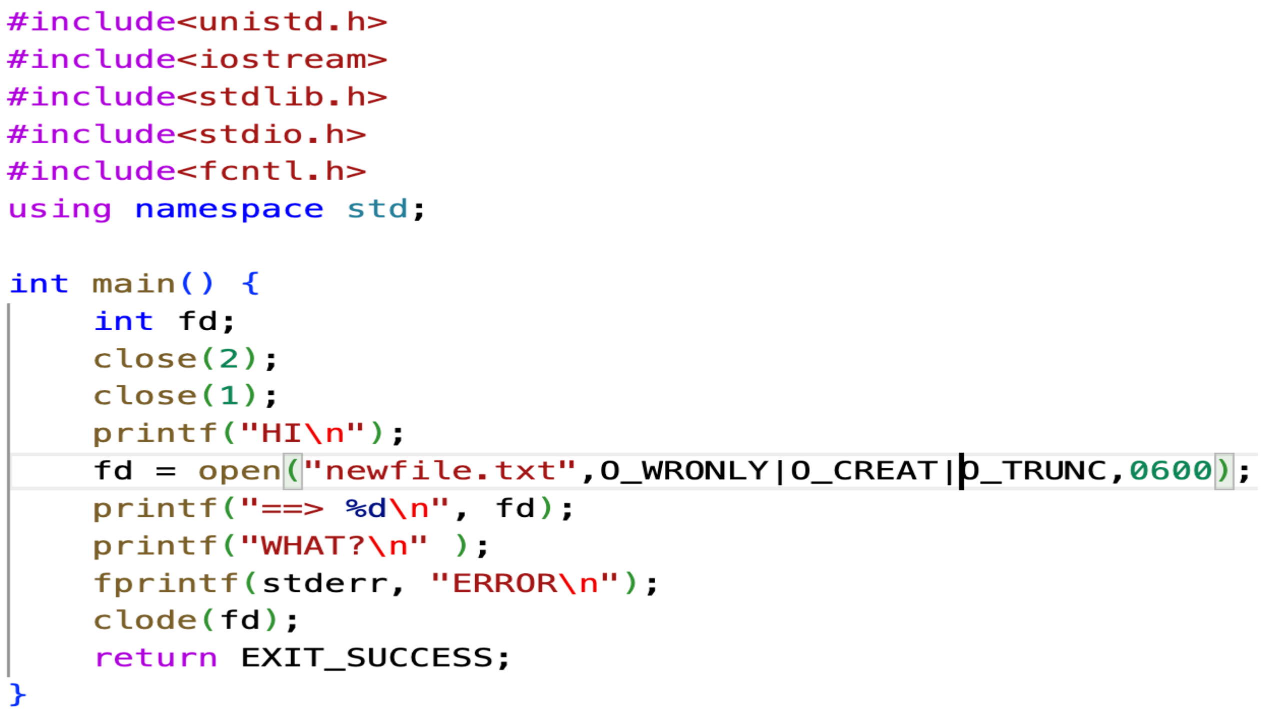
Question 2 [4 Marks]

What is the exact output that this program will produce? If multiple outputs are possible, succinctly describe all possibilities. Assume all the system calls complete successfully.

```
#include<unistd.h>
#include<iostream>
#include<stdlib.h>
#include<stdio.h>
using namespace std;

int main() {
    printf("ONE\n" );
    fprintf(stderr, "ERROR: ONE \n" );
    int rc = close(1);
    printf("TWO \n");
    fprintf(stderr, "ERROR: TWO\n" );
    rc = close(2);
    printf("THREE \n");
    fprintf(stderr, "ERROR: THREE\n");
    return EXIT_SUCCESS;
}
```

Question 3 [3 + 3 = 6 Marks]

Consider the below code.



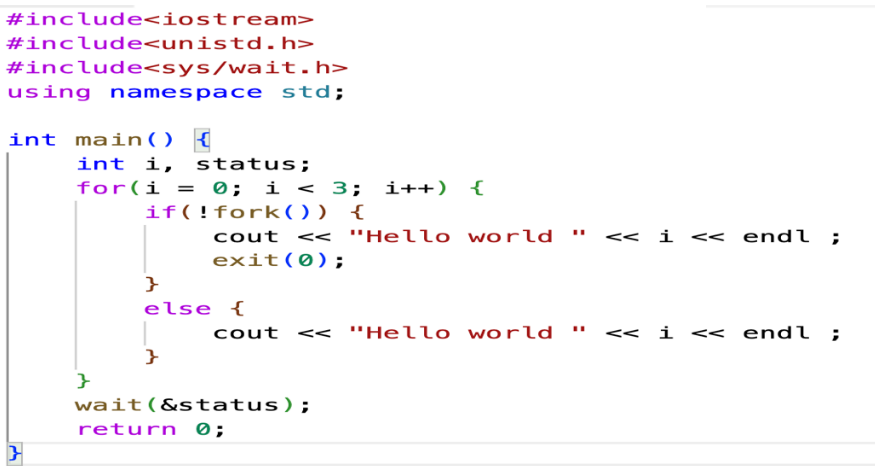
```
#include<unistd.h>
#include<iostream>
#include<stdlib.h>
#include<stdio.h>
#include<fcntl.h>
using namespace std;

int main() {
    int fd;
    close(2);
    close(1);
    printf("HI\n");
    fd = open("newfile.txt",O_WRONLY|O_CREAT|O_TRUNC,0600);
    printf("==> %d\n", fd);
    printf("WHAT?\n" );
    fprintf(stderr, "ERROR\n");
    clode(fd);
    return EXIT_SUCCESS;
}
```

□ What is the exact output? Further, what is the exact contents of the nwfile.txt file? If multiple outputs are possible, succinctly describe all possibilities. Assume that all system calls are completely successful.

□ How do the output and the file content change if we uncomment close (1) system call?

Question 4 [2 + 1 + 3 = 6 Marks]

Consider the below code:



```
#include<iostream>
#include<unistd.h>
#include<sys/wait.h>
using namespace std;

int main() {
    int i, status;
    for(i = 0; i < 3; i++) {
        if(!fork()) {
            cout << "Hello world " << i << endl ;
            exit(0);
        }
        else {
            cout << "Hello world " << i << endl ;
        }
    }
    wait(&status);
    return 0;
}
```

□ How many total processes are created, including the initial parent process?

□ How many total processes are running after the loop?

□ At the end of the program, are there any zombie processes? If so, why are there zombies and what happens to them when the parent terminates? If not, why aren't there any zombie processes?