Prog.c code

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

#define BUFFER_SIZE 100

int main() {
    int fd1[2];  /* Pipe for communication from parent to child */
    int fd2[2];  /* Pipe for communication from child to parent */
    pid_t pid;

    char read_msg[BUFFER_SIZE];
    char write_msg[BUFFER_SIZE];

    /* Initialize two pipes */
    if (pipe(fd1) == -1 || pipe(fd2) == -1) {
        fprintf(stderr, "Pipe failed");
        return 1;
    }

    /* Fork a child process */
    pid = fork();

    if (pid < 0) {
        fprintf(stderr, "Fork failed");
        return 1;
    }

    if (pid > 0) {  /* Parent process */
        close(fd1[0]);  /* Close the unused read end of parent to child pipe, dad will use that for writing and son to read what he wrote */
        close(fd2[1]);  /* Close the unused write end of child to parent pipe, son will use that for writing and dad to read what he wrote */

        strcpy(write_msg, "Greetings son.");
        write(fd1[1], write_msg, strlen(write_msg) + 1);
```

```
        read(fd2[0], read_msg, BUFFER_SIZE);
        printf("%s\n", read_msg);
        sleep(2);


        strcpy(write_msg, "How are you son.");
        write(fd1[1], write_msg, strlen(write_msg) + 1);

        read(fd2[0], read_msg, BUFFER_SIZE);
        printf("%s\n", read_msg);
        sleep(2);

        close(fd1[1]); /* Close the write end after sending
messages */
        close(fd2[0]); /* Close the read end after reading
responses */

    } else {  /* Child process */
        close(fd1[1]);  /* Close the unused write end of parent
to child pipe */
        close(fd2[0]);  /* Close the unused read end of child to
parent pipe */

        read(fd1[0], read_msg, BUFFER_SIZE);
        printf("%s\n", read_msg);
        sleep(2);

        strcpy(write_msg, "Hello dad.");
        write(fd2[1], write_msg, strlen(write_msg) + 1);

        read(fd1[0], read_msg, BUFFER_SIZE);
        printf("%s\n", read_msg);
        sleep(2);

        strcpy(write_msg, "Fine dad, thanks.");
        write(fd2[1], write_msg, strlen(write_msg) + 1);

        close(fd1[0]); /* Close the read end after reading
messages */
```

```
        close(fd2[1]); /* Close the write end after sending
responses */
    }

    return 0;
}
```

RESULTATS:

```
cd "/Users/keliane/devoir1_csi3531/" && gcc prog.c -o prog && "/Users/keliane/devoir1_cs
keliane@Kelianes-Mac devoir1_csi3531 % cd "/Users/keliane/devoir1_csi3531/" && gcc prog.
Greetings son.
Hello dad.
How are you son.
Fine dad, thanks.
keliane@Kelianes-Mac devoir1_csi3531 % cd "/Users/keliane/devoir1_csi3531/" && gcc prog.
Greetings son.
Hello dad.
How are you son.
Fine dad, thanks.
keliane@Kelianes-Mac devoir1_csi3531 %
```