

main.c

↺ ↻ 🔗 Share

Run

```
1 #include <stdio.h>
2 #include<unistd.h>
3 int main()
4 {
5     pid_t pid=fork();
6     if (pid<0)
7     {
8         perror("fork failed");
9         return 1;
10    }
11    else if (pid==0)
12    {
13        printf("childprocess:\n");
14        printf("PID:%d\n",getpid());
15        printf("parent PID:%d\n",getpid());
16    }
17    else
18    {
19        printf("parentprocess:\n");
20        printf("PID:%d\n",getpid());
21        printf("child PID:%d\n",pid);
22    }
23    return 0;
24 }
```

Output

Clean

parentprocess:
PID:33862
child PID:33863
childprocess:
PID:33863
parent PID:33863

=== Code Execution Successful ===

main.c



Run

Output

```
2 #include <fcntl.h>
3 #include <unistd.h>
4 #include <stdlib.h>
5 #define BUFFER_SIZE 1024
6 int main(int argc, char *argv[]) {
7     if (argc != 3) {
8         fprintf(stderr, "Usage: %s <source_file> <destination_file>\n", argv[0]);
9         exit(1);
10    }
11    int src_fd = open(argv[1], O_RDONLY);
12    if (src_fd < 0) {
13        perror("Error opening source file");
14        exit(1);
15    }
16    int dest_fd = open(argv[2], O_WRONLY | O_CREAT | O_TRUNC, 0644);
17    if (dest_fd < 0) {
18        perror("Error opening/creating destination file");
19        close(src_fd);
20        exit(1);
21    }
22    char buffer[BUFFER_SIZE];
23    ssize_t bytes_read, bytes_written;
24    while ((bytes_read = read(src_fd, buffer, BUFFER_SIZE)) > 0) {
25        bytes_written = write(dest_fd, buffer, bytes_read);
26        if (bytes_written != bytes_read) {
27            perror("Error writing to destination file");
28            close(src_fd);
29            close(dest_fd);
30            exit(1);
31        }
32    }
33    if (bytes_read < 0) {
34        perror("Error reading source file");
35    }
36    close(src_fd);
37    close(dest_fd);
38    printf("File copied successfully.\n");
39    return 0;
40 }
```

Usage: /tmp/hf8wJDPm09/main.o <source_file> <destination_file>

=== Code Exited With Errors ===