main.c

Share Run

Output Clear

```c
#include <stdio.h>

int main() {
    FILE *fp;
    char str[100];

    fp = fopen("file.txt", "w");
    fprintf(fp, "Operating Systems File Handling\n");
    fclose(fp);

    fp = fopen("file.txt", "r");
    fgets(str, 100, fp);
    printf("Read from file: %s", str);
    fclose(fp);

    remove("file.txt"); // Deletes the file

    return 0;
}
```

```
Segmentation fault


=== Code Exited With Errors ===
```

main.c

Share Run

```c
#include <stdio.h>
#include <pthread.h>

void* threadFunc(void* arg) {
    printf("Thread is running with ID: %lu\n", pthread_self());
    pthread_exit(NULL);
}

int main() {
    pthread_t tid1, tid2;

    pthread_create(&tid1, NULL, threadFunc, NULL);
    pthread_create(&tid2, NULL, threadFunc, NULL);

    pthread_join(tid1, NULL);
    pthread_join(tid2, NULL);

    if (pthread_equal(tid1, tid2))
        printf("Threads are equal\n");
```

Output Clear

```
Thread is running with ID: 136832521828032
Thread is running with ID: 136832513435328
Threads are not equal
Main thread exiting.


=== Code Execution Successful ===
```

main.c

Share Run

```c
#include <stdio.h>
#include <string.h>

int main() {
    FILE *fp = fopen("sample.txt", "r");
    char word[] = "hello";
    char line[100];

    if (!fp) {
        perror("fopen");
        return 1;
    }

    while (fgets(line, sizeof(line), fp)) {
        if (strstr(line, word))
            printf("%s", line);
    }

    fclose(fp);
```

Output Clear

```
fopen: No such file or directory


=== Code Exited With Errors ===
```

main.c

Share Run

Output Clear

```c
#include <dirent.h>

int main() {
    DIR *dir;
    struct dirent *entry;

    dir = opendir(".");
    if (dir == NULL) {
        perror("opendir");
        return 1;
    }

    printf("Files in current directory:\n");
    while ((entry = readdir(dir)) != NULL)
        printf("%s\n", entry->d_name);

    closedir(dir);
    return 0;
}
```

```
Files in current directory:
.
..
.bash_logout
.bashrc
.profile


=== Code Execution Successful ===
```

main.c

Share Run

Output Clear

```c
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h>

#define SIZE 5
int buffer[SIZE];
int in = 0, out = 0;

sem_t empty, full;
pthread_mutex_t mutex;

void* producer(void* arg) {
    int item = 1;
    for (int i = 0; i < 5; i++) {
        sem_wait(&empty);
        pthread_mutex_lock(&mutex);
```

```
Produced: 1
Consumed: 1
Produced: 2
Consumed: 2
Produced: 3
Consumed: 3
Produced: 4
Consumed: 4
Produced: 5
Consumed: 5


=== Code Execution Successful ===
```