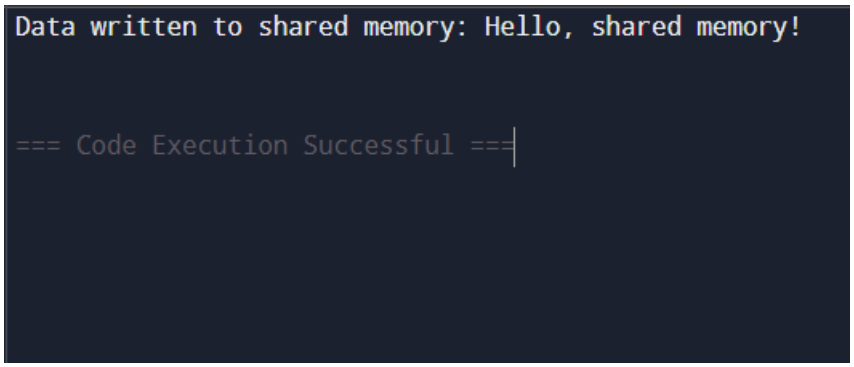
9.Illustrate the concept of inter-process communication using shared memory with a C program.

PROGRAM :

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/ipc.h>
#include <sys/shm.h>

#define SHM_SIZE 1024

int main() {
    // Use /tmp directory for creating a temporary file
    const char *filename = "/tmp/shmfile";

    // Ensure the file exists
    FILE *file = fopen(filename, "w");
    if (file == NULL) {
        perror("fopen");
        exit(EXIT_FAILURE);
    }
    fclose(file);

    // Generate a unique key for shared memory
    key_t key = ftok(filename, 65);
    if (key == -1) {
        perror("ftok");
        exit(EXIT_FAILURE);
    }
```

```
// Create shared memory segment
int shmid = shmget(key, SHM_SIZE, IPC_CREAT | 0666);
if (shmid == -1) {
    perror("shmget");
    exit(EXIT_FAILURE);
}

// Attach to the shared memory
char *shm_ptr = (char *)shmat(shmid, NULL, 0);
if (shm_ptr == (char *)(-1)) {
    perror("shmat");
    exit(EXIT_FAILURE);
}

// Write data to shared memory
strcpy(shm_ptr, "Hello, shared memory!");

// Print the data written
printf("Data written to shared memory: %s\n", shm_ptr);

// Detach from shared memory
if (shmdt(shm_ptr) == -1) {
    perror("shmdt");
    exit(EXIT_FAILURE);
}

// Remove the shared memory segment
if (shmctl(shmid, IPC_RMID, NULL) == -1) {
    perror("shmctl");
    exit(EXIT_FAILURE);
```

```
    }

    return 0;
}
```


Data written to shared memory: Hello, shared memory!
=== Code Execution Successful ===