29. Write a C program to simulate the solution of Classical Process Synchronization Problem.

```
#include <stdio.h>
#include <stdlib.h>
int mutex = 1, full = 0, empty = 3, x = 0; // buffer size = 3
void wait(int *s) { (*s)--; }
void signal(int *s) { (*s)++; }
void producer() {
   wait(&mutex);
   wait(&empty);
   x++;
   printf("Producer produces item %d\n", x);
   signal(&full);
   signal(&mutex);
}
void consumer() {
   wait(&mutex);
   wait(&full);
   printf("Consumer consumes item %d\n", x);
   x--;
   signal(&empty);
   signal(&mutex);
}
int main() {
   int choice;
   printf("--- Classical Process Synchronization (Producer-Consumer) ---\n");
```

```
    while (1) {
        printf("\n1. Produce 2. Consume  3. Exit\nEnter choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                if ((mutex == 1) && (empty != 0))
                    producer();
                else
                    printf("Buffer is Full!\n");
                break;
            case 2:
                if ((mutex == 1) && (full != 0))
                    consumer();
                else
                    printf("Buffer is Empty!\n");
                break;
            case 3:
                exit(0);
            default:
                printf("Invalid choice!\n");
        }
    }
    return 0;
}
```

OUTPUT-

```c
void consumer() {
    wait(&mutex);
    wait(&full);
    printf("Consumer consumes item %d\n", x);
    x--;
    signal(&empty);
    signal(&mutex);
}
int main() {
    int choice;
    printf("--- Classical Process Synchronization (Producer-Consumer) ---\n");
    while (1) {
        printf("\n1. Produce 2. Consume  3. Exit\nEnter choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                if ((mutex == 1) && (empty != 0))
                    producer();
                else
                    printf("Buffer is Full!\n");
                break;
            case 2:
                if ((mutex == 1) && (full != 0))
                    consumer();
                else
                    printf("Buffer is Empty!\n");
                break;
            case 3:
                exit(0);
            default:
                printf("Invalid choice!\n");
        }
    }
    return 0;
}
```

```
--- Classical Process Synchronization (Producer-C

1. Produce 2. Consume  3. Exit
Enter choice: 1
Producer produces item 1

1. Produce 2. Consume  3. Exit
Enter choice: 2
Consumer consumes item 1

1. Produce 2. Consume  3. Exit
Enter choice: 3

--------------------------------
Process exited after 13.93 seconds with return va
Press any key to continue . . .
```

.