Operating System (CT-353) LAB 04 & 05

1) Implement the above code and paste the screen shot of the output.

CODE

```c
#include <stdio.h>

int main() {
    int buffer[10], bufsize, in, out, produce, consume, choice = 0;
    in = 0;
    out = 0;
    bufsize = 10;

    while (choice != 3) {
        printf("\n1. Produce \t 2. Consume \t3. Exit");
        printf("\nEnter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                if ((in + 1) % bufsize == out)
                    printf("\nBuffer is Full");
                else {
                    printf("\nEnter the value: ");
                    scanf("%d", &produce);
                    buffer[in] = produce;
                    in = (in + 1) % bufsize;
                }
                break;

            case 2:
                if (in == out)
                    printf("\nBuffer is Empty");
                else {
                    consume = buffer[out];
                    printf("\nThe consumed value is %d", consume);
                    out = (out + 1) % bufsize;
                }
                break;
        }
    }
}
```

MUHAMMAD AAMIR
DT-22047

Operating System (CT-353) LAB 04 & 05

OUTPUT

```
1    #include <stdio.h>
2
3    int main() {
```

```
C:\Users\admin\Downloads\DM lab 04.exe

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

Buffer is Empty
1. Produce      2. Consume     3. Exit
Enter your choice: 1

Enter the value: 5

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

The consumed value is 5
1. Produce      2. Consume     3. Exit
Enter your choice: 1

Enter the value: 54

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

Enter the value: 2

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

The consumed value is 54
1. Produce      2. Consume     3. Exit
Enter your choice: 50
```

- Warnings: 0
- Output Filename: C:\Users\admin\Downloads\DM lab 04.exe

```
Enter your choice: 1

Enter the value: 2

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

The consumed value is 54
1. Produce      2. Consume     3. Exit
Enter your choice: 50

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

Enter the value: 20

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

The consumed value is 2
1. Produce      2. Consume     3. Exit
Enter your choice: 3

------------------------------
Process exited after 59.81 seconds with return value 0
Press any key to continue . . .
```

MUHAMMAD AAMIR
DT-22047

2) Solve the producer-consumer problem using linked list. Note: Keep the buffer size to 10 places.

CODE

```
#include <stdio.h>
#define BUFFER_SIZE 10

typedef struct Node {
    int data;
    struct Node* next;
} Node;

Node* head = NULL;
Node* tail = NULL;
int count = 0;

pthread_mutex_t mutex;
sem_t empty, full;

void insert(int item) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->data = item;
    newNode->next = NULL;

    if (tail == NULL) {
        head = tail = newNode;
    } else {
        tail->next = newNode;
        tail = newNode;
    }
    count++;
}

int remove_item() {
    if (head == NULL) return -1;

    Node* temp = head;
    int item = temp->data;
    head = head->next;

    if (head == NULL) tail = NULL;

    free(temp);
    count--;
    return item;
}

void* producer(void* arg) {
```

```c
    int item;
    while (1) {
        item = rand() % 100;
        sem_wait(&empty);
        pthread_mutex_lock(&mutex);

        insert(item);
        printf("Produced: %d\n", item);

        pthread_mutex_unlock(&mutex);
        sem_post(&full);
        sleep(1);
    }
}

void* consumer(void* arg) {
    int item;
    while (1) {
        sem_wait(&full);
        pthread_mutex_lock(&mutex);

        item = remove_item();
        printf("Consumed: %d\n", item);

        pthread_mutex_unlock(&mutex);
        sem_post(&empty);
        sleep(1);
    }
}

int main() {
    pthread_t prod, cons;

    pthread_mutex_init(&mutex, NULL);
    sem_init(&empty, 0, BUFFER_SIZE);
    sem_init(&full, 0, 0);

    pthread_create(&prod, NULL, producer, NULL);
    pthread_create(&cons, NULL, consumer, NULL);

    pthread_join(prod, NULL);
    pthread_join(cons, NULL);

    pthread_mutex_destroy(&mutex);
    sem_destroy(&empty);
    sem_destroy(&full);
    return 0;
}
```
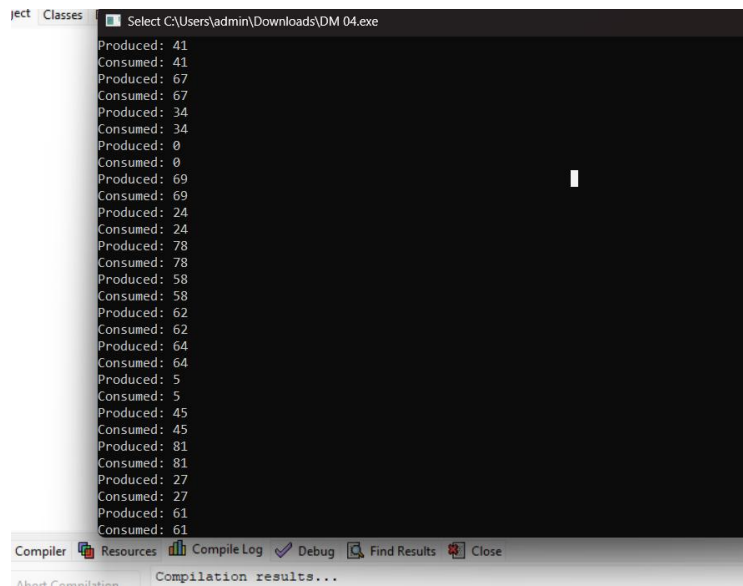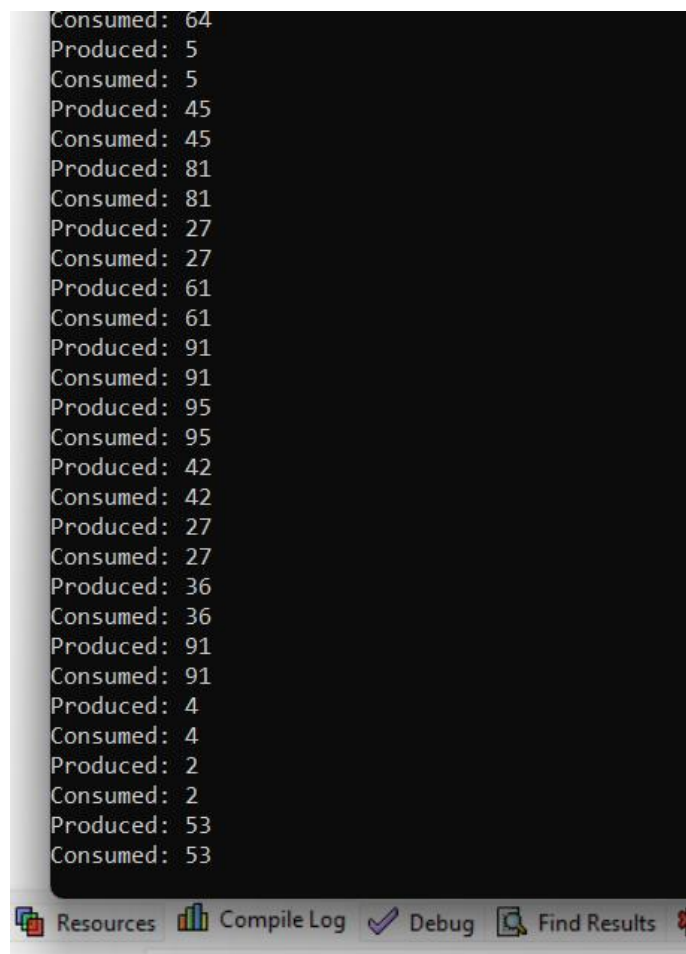
Operating System (CT-353) LAB 04 & 05

Output

Select C:\Users\admin\Downloads\DM 04.exe

```
Produced: 41
Consumed: 41
Produced: 67
Consumed: 67
Produced: 34
Consumed: 34
Produced: 0
Consumed: 0
Produced: 69
Consumed: 69
Produced: 24
Consumed: 24
Produced: 78
Consumed: 78
Produced: 58
Consumed: 58
Produced: 62
Consumed: 62
Produced: 64
Consumed: 64
Produced: 5
Consumed: 5
Produced: 45
Consumed: 45
Produced: 81
Consumed: 81
Produced: 27
Consumed: 27
Produced: 61
Consumed: 61
```

Compiler Resources Compile Log Debug Find Results Close

Compilation results...

```
Consumed: 64
Produced: 5
Consumed: 5
Produced: 45
Consumed: 45
Produced: 81
Consumed: 81
Produced: 27
Consumed: 27
Produced: 61
Consumed: 61
Produced: 91
Consumed: 91
Produced: 95
Consumed: 95
Produced: 42
Consumed: 42
Produced: 27
Consumed: 27
Produced: 36
Consumed: 36
Produced: 91
Consumed: 91
Produced: 4
Consumed: 4
Produced: 2
Consumed: 2
Produced: 53
Consumed: 53
```

Resources Compile Log Debug Find Results

MUHAMMAD AAMIR
DT-22047

3) In producer-consumer problem what difference will it make if we utilize stack for the buffer rather

than an array?

Using a stack instead of a queue in the producer-consumer problem fundamentally changes the processing order from FIFO to LIFO, which may not be suitable for many traditional producer-consumer use cases.

1) Implement the above code and paste the screen shot of the output.

CODE

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

sem_t x, y;
pthread_t tid;
pthread_t writerthreads[100], readerthreads[100];
int readercount = 0;

void *reader(void *param) {
    sem_wait(&x);
    readercount++;

    if (readercount == 1) {
        sem_wait(&y);
    }
    sem_post(&x);

    printf("%d reader is inside\n", readercount);
    usleep(3);

    sem_wait(&x);
    readercount--;
    if (readercount == 0) {
        sem_post(&y);
    }
    sem_post(&x);
    printf("%d Reader is leaving\n", readercount + 1);
    return NULL;
}

void *writer(void *param) {
    printf("Writer is trying to enter\n");
    sem_wait(&y);
    printf("Writer has entered\n");
```

MUHAMMAD AAMIR
DT-22047

```
    sem_post(&y);
    printf("Writer is leaving\n");
    return NULL;
}

int main() {
    int n2, i;
    printf("Enter the number of readers:");
    scanf("%d", &n2);

    int n1[n2];
    sem_init(&x, 0, 1);
    sem_init(&y, 0, 1);

    for (i = 0; i < n2; i++) {
        pthread_create(&writerthreads[i], NULL, reader, NULL);
        pthread_create(&readerthreads[i], NULL, writer, NULL);
    }

    for (i = 0; i < n2; i++) {
        pthread_join(writerthreads[i], NULL);
        pthread_join(readerthreads[i], NULL);
    }

    return 0;
}
```
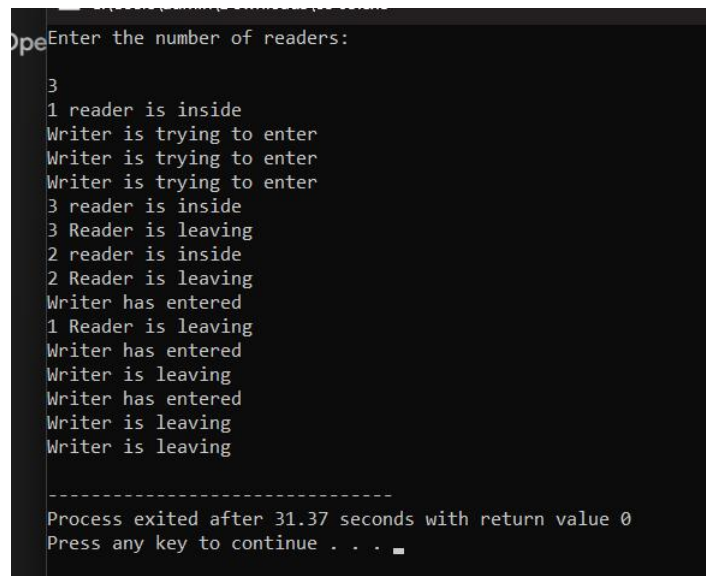
Output

MUHAMMAD AAMIR
DT-22047