QUESTION # 1

In this bounded buffer problem , our goal is to put producer to sleep when buffer is full and consumer to sleep when there is no item in buffer. Does the given code fragment fulfill the desired objective ? If no, then modify the given code fragment to achieve the desired objective.

```
Semaphore mutex=1;
Semaphore empty=N;

producer
{
while (1)
{

wait(empty);
wait(mutex);

//producer code

signal(mutex);
}
}

consumer
{
while (1)
{
wait(mutex)

//consumer code


signal(mutex)
signal(empty);
}
}
```

QUESTION # 2

Consider the bounded buffer problem again. A serious problem might be encountered by reversing the order of mutex and empty semaphore. You are required to identify the problem in the given code fragment along with brief explanation.

```
producer
{
while (1)
{

wait(mutex);
wait(empty);
```

```
//producer code

signal(mutex);
}
}
```

QUESTION # 3

Does the given two process solution for critical section problem satisfy the mutual exclusion criteria? Elaborate**.**

PROCESS P0

```
flag[0] = false;
flag[1] = false;
P0:
while (true)
{
 flag[0] = true;
while (flag[1]) {
flag[0] = false;
while (flag[1]) {
no-op;
}
flag[0] = true;
}

critical section
flag[0] = false;
remainder section
}
```

PROCESS P1

```
while (true) {
flag[1] = true;
while (flag[0])
{
flag[1] = false;
while (flag[0])
{
no-op;
}
flag[1] = true;
}
critical section
flag[1] = false;
remainder section
}
```

QUESTION # 4
Consider the given two process solution for resolving the critical section problem. Initially variable V1 and V2 are assigned different values.

PROCESS P0

While (V1==V2);
critical section
V1=V2;

PROCESS P1
While (V1!=V2);
critical section
V2 = not(V1);

1) Does the above code fragment satisfy the mutual exclusion criteria? Explain.
2) Does the above code fragment make a progress? Explain.