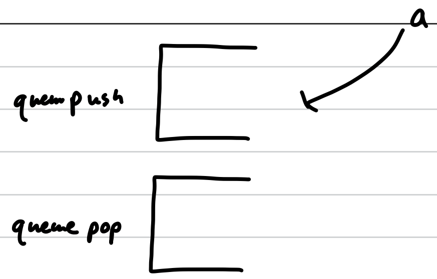
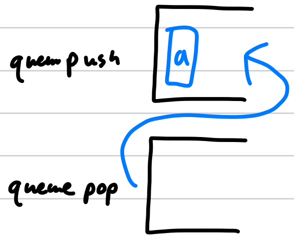
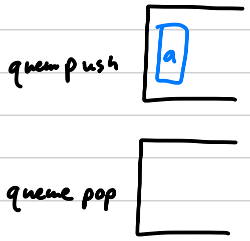
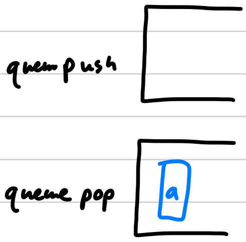
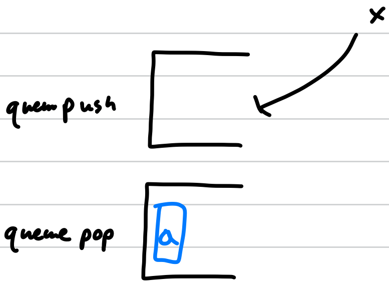
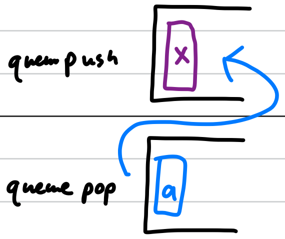

a
queue push
queue pop

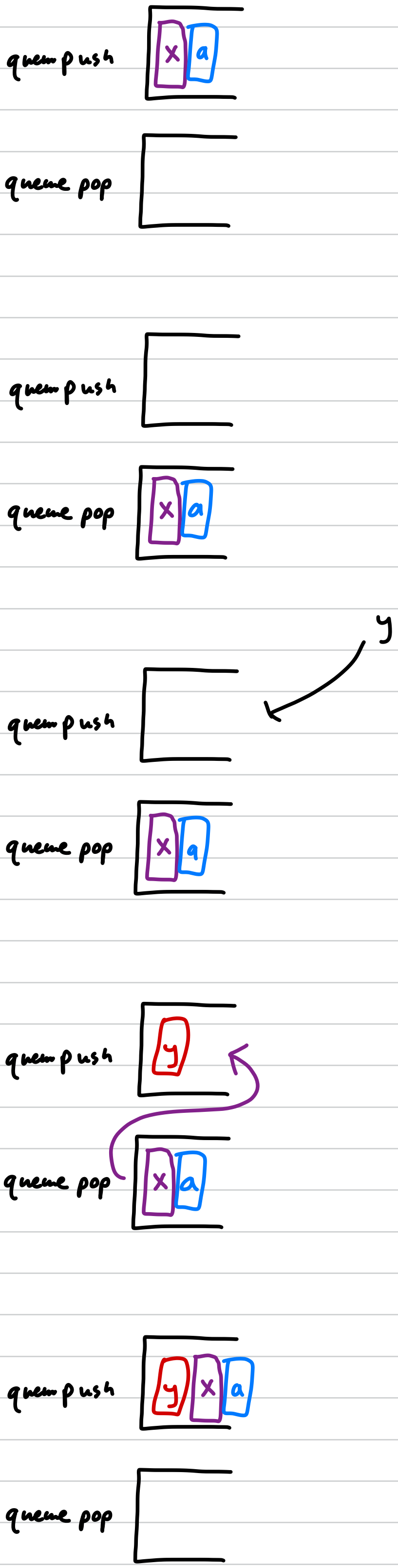

queue push
a
queue pop

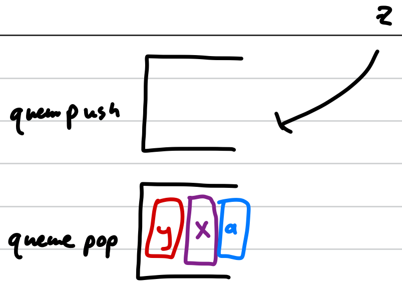

queue push
a
queue pop

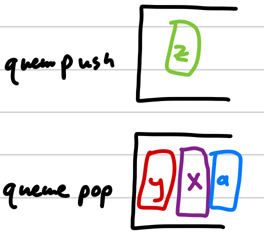

queue push
queue pop
a

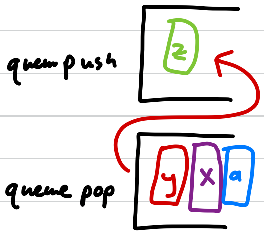

x
queue push
queue pop
a

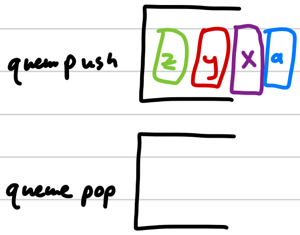

queue push
x
queue pop
a

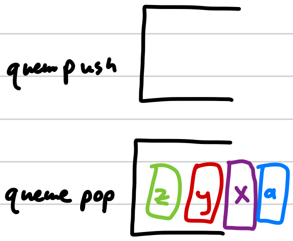

queue push
x
a
queue pop
queue push
queue pop
x
a
y
queue push
queue pop
x
a
queue push
y
queue pop
x
a
queue push
y
x
a
queue pop

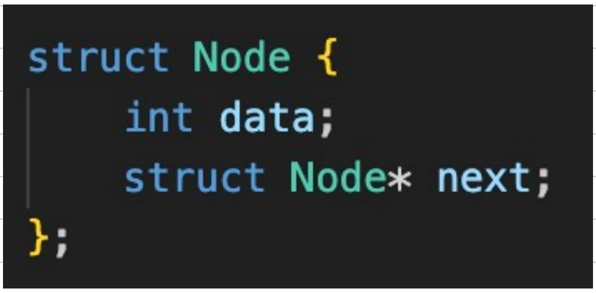

z
queue push
queue pop
y
x
a


queue push
z
queue pop
y
x
a

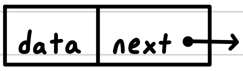

queue push
z
queue pop
y
x
a

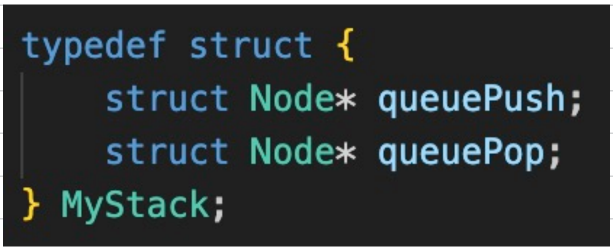

queue push
z
y
x
a
queue pop


queue push
queue pop
z
y
x
a

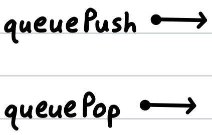


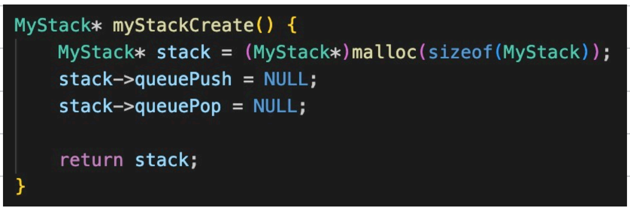
```c
struct Node {
    int data;
    struct Node* next;
};
```



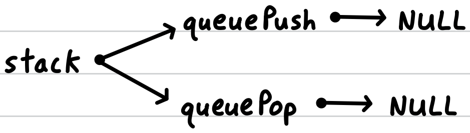


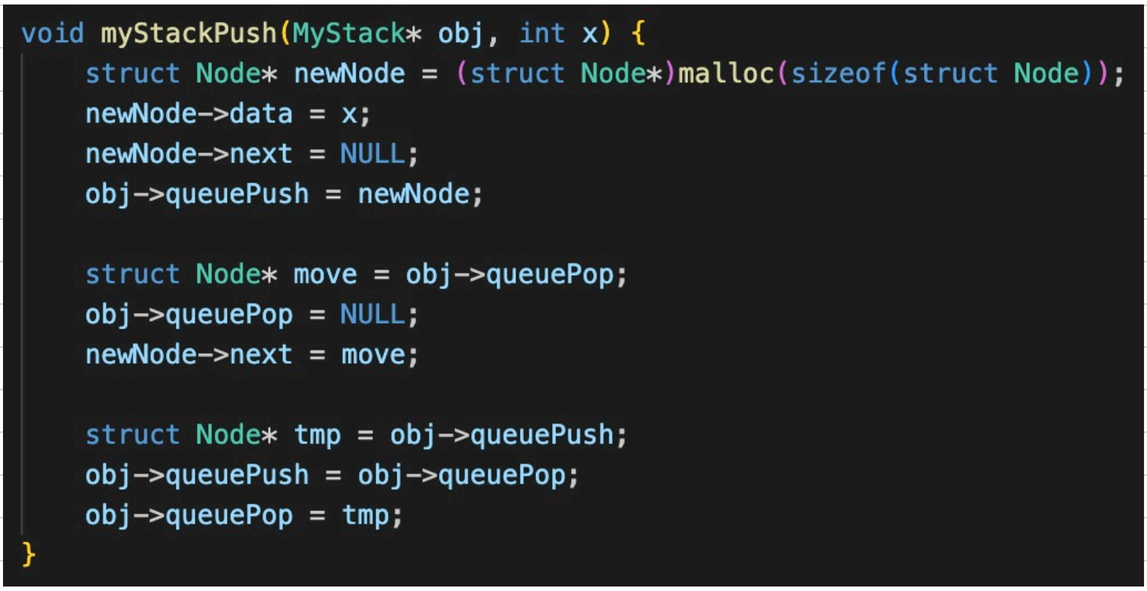
```c
typedef struct {
    struct Node* queuePush;
    struct Node* queuePop;
} MyStack;
```





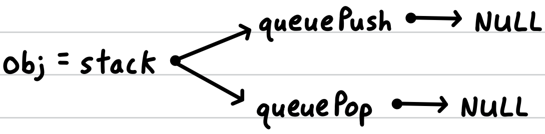
```c
MyStack* myStackCreate() {
    MyStack* stack = (MyStack*)malloc(sizeof(MyStack));
    stack->queuePush = NULL;
    stack->queuePop = NULL;

    return stack;
}
```

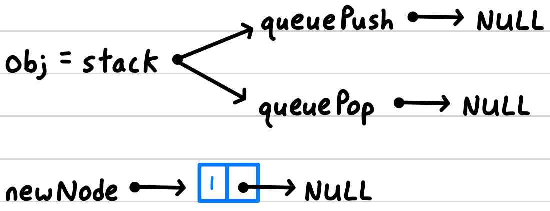


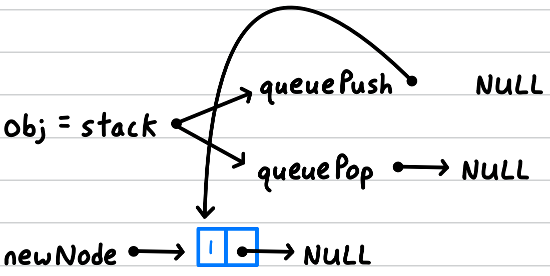


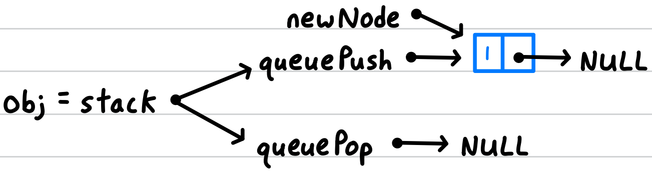
```c
void myStackPush(MyStack* obj, int x) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = x;
    newNode->next = NULL;
    obj->queuePush = newNode;

    struct Node* move = obj->queuePop;
    obj->queuePop = NULL;
    newNode->next = move;

    struct Node* tmp = obj->queuePush;
    obj->queuePush = obj->queuePop;
    obj->queuePop = tmp;
}
```

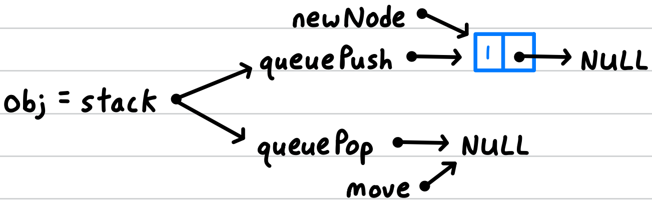
Push 1,2,3,4,5

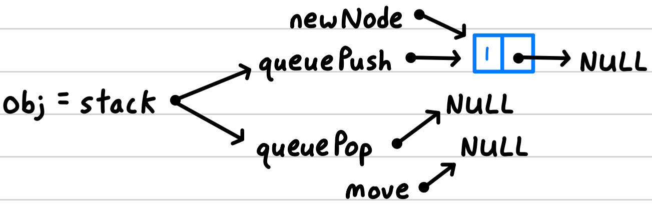


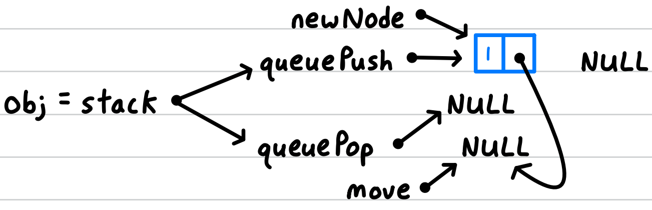

queuePush
NULL
obj = stack
queuePop
NULL
newNode
1
NULL

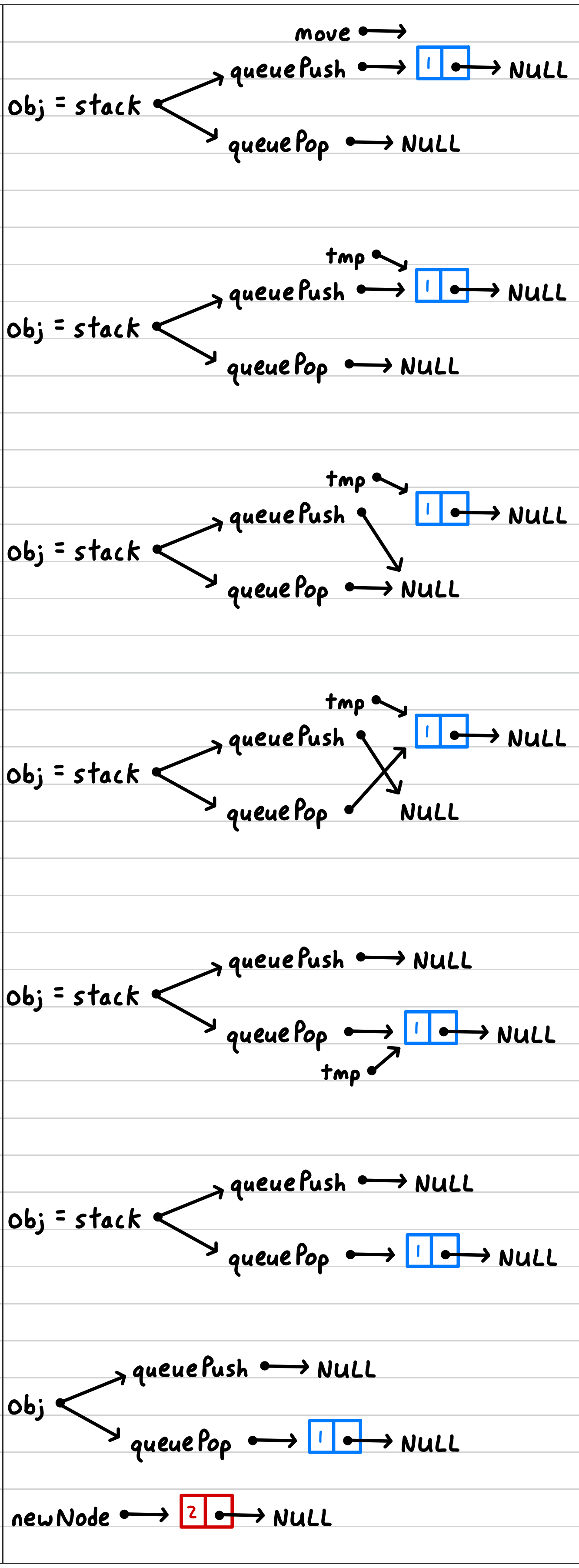

queuePush
NULL
obj = stack
queuePop
NULL
newNode
1
NULL

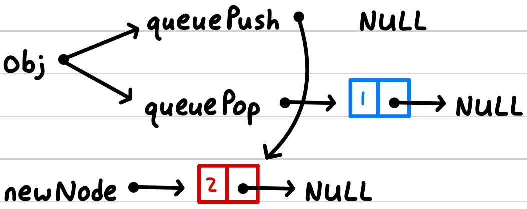

newNode
queuePush
1
NULL
obj = stack
queuePop
NULL

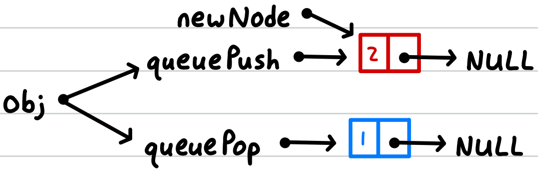

newNode
queuePush
1
NULL
obj = stack
queuePop
NULL
move

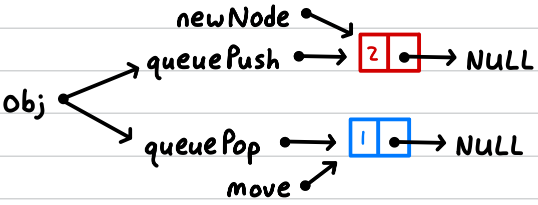

newNode
queuePush
1
NULL
obj = stack
NULL
queuePop
NULL
move

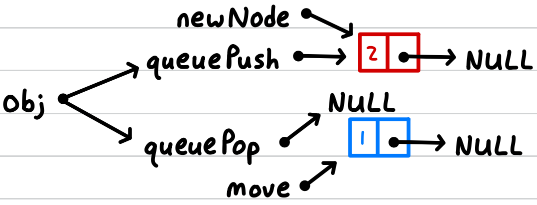

newNode
queuePush
1
NULL
obj = stack
NULL
queuePop
NULL
move

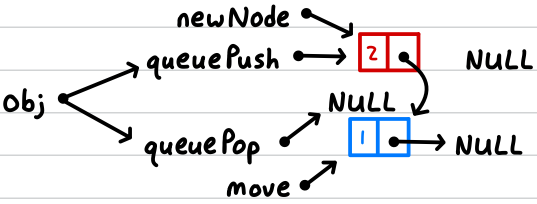

move
queuePush
1
NULL
obj = stack
queuePop
NULL
tmp
queuePush
1
NULL
obj = stack
queuePop
NULL
tmp
queuePush
1
NULL
obj = stack
queuePop
NULL
tmp
queuePush
1
NULL
obj = stack
queuePop
NULL
queuePush
NULL
obj = stack
queuePop
1
NULL
tmp
queuePush
NULL
obj = stack
queuePop
1
NULL
queuePush
NULL
obj
queuePop
1
NULL
newNode
2
NULL

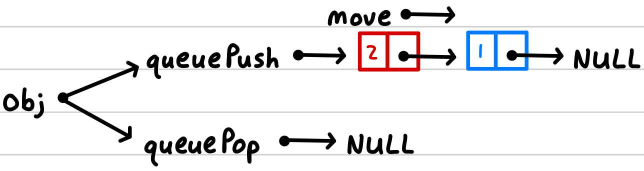

obj
queuePush
NULL
queuePop
1
NULL
newNode
2
NULL

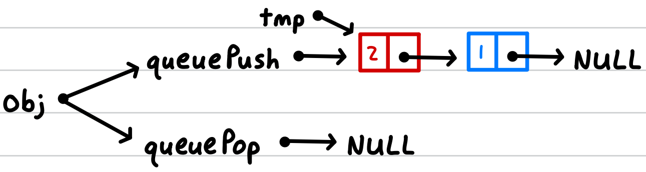

newNode
queuePush
2
NULL
obj
queuePop
1
NULL

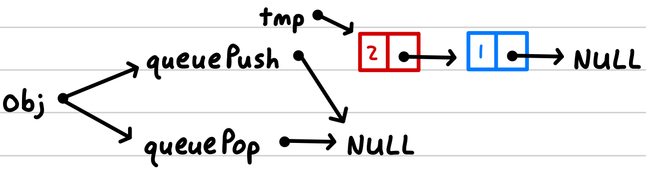

newNode
queuePush
2
NULL
obj
queuePop
1
NULL
move

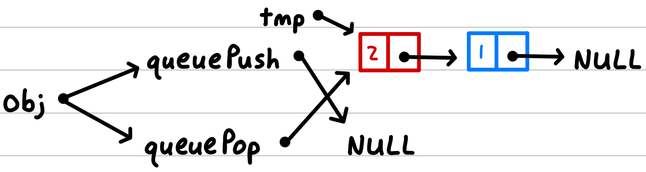

newNode
queuePush
2
NULL
obj
NULL
queuePop
1
NULL
move

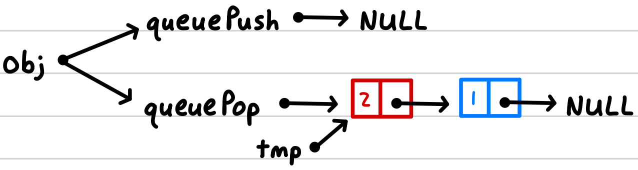

newNode
queuePush
2
NULL
obj
NULL
queuePop
1
NULL
move

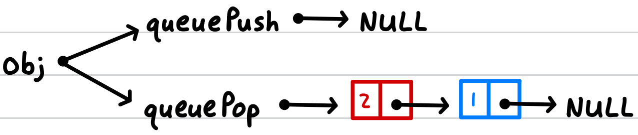

move
queuePush
2
1
NULL
obj
queuePop
NULL

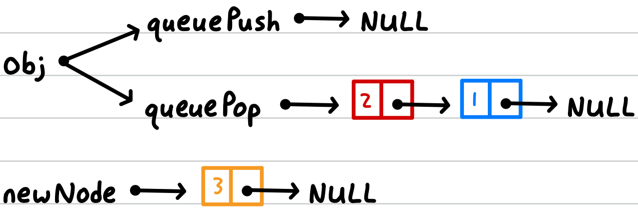

tmp
obj
queuePush
queuePop
2
1
NULL
NULL

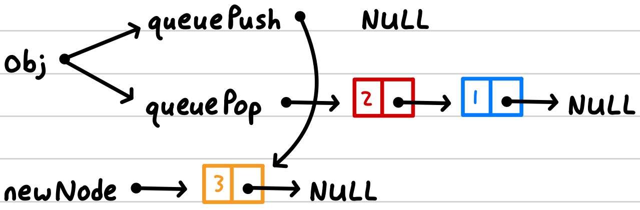

tmp
obj
queuePush
queuePop
2
1
NULL
NULL

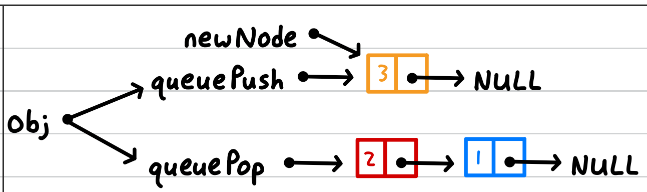

tmp
obj
queuePush
queuePop
2
1
NULL
NULL

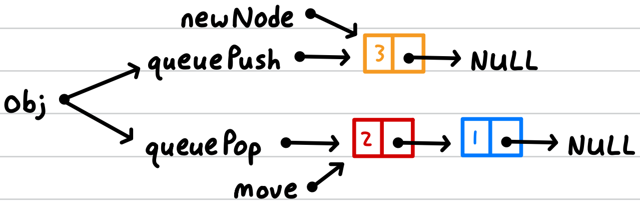

obj
queuePush
NULL
queuePop
2
1
NULL
tmp

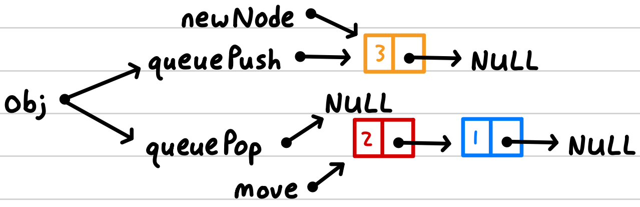

obj
queuePush
NULL
queuePop
2
1
NULL

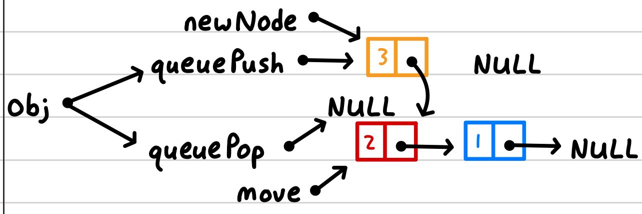

obj
queuePush
NULL
queuePop
2
1
NULL
newNode
3
NULL

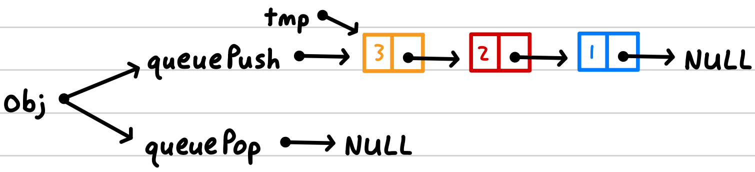

obj
queuePush
NULL
queuePop
2
1
NULL
newNode
3
NULL

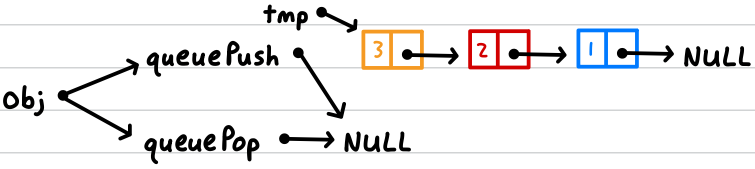

newNode
queuePush
obj
queuePop
3
NULL
2
1
NULL

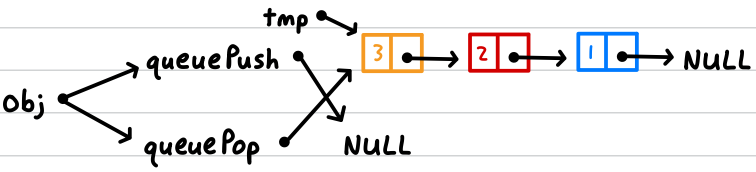

newNode
queuePush
obj
queuePop
move
3
NULL
2
1
NULL

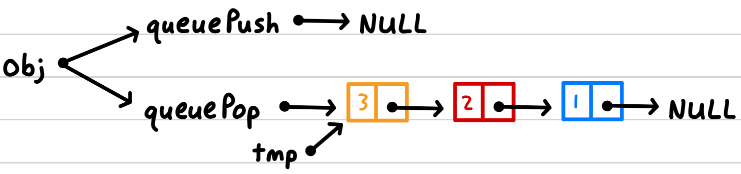

newNode
queuePush
obj
queuePop
move
3
NULL
NULL
2
1
NULL

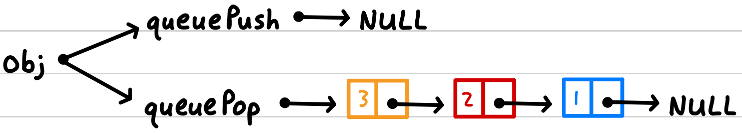

newNode
queuePush
obj
queuePop
move
3
NULL
NULL
2
1
NULL

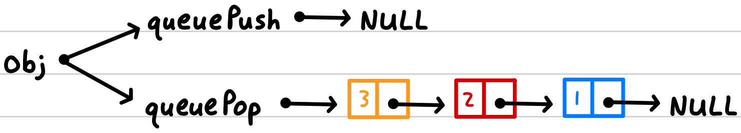

tmp
queuePush
obj
queuePop
3
2
1
NULL
NULL


tmp
queuePush
obj
queuePop
3
2
1
NULL
NULL





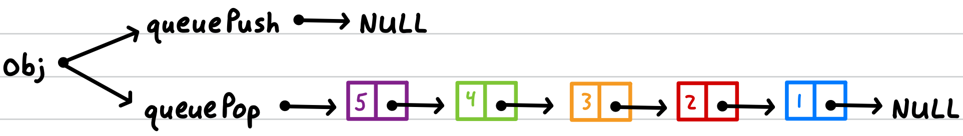


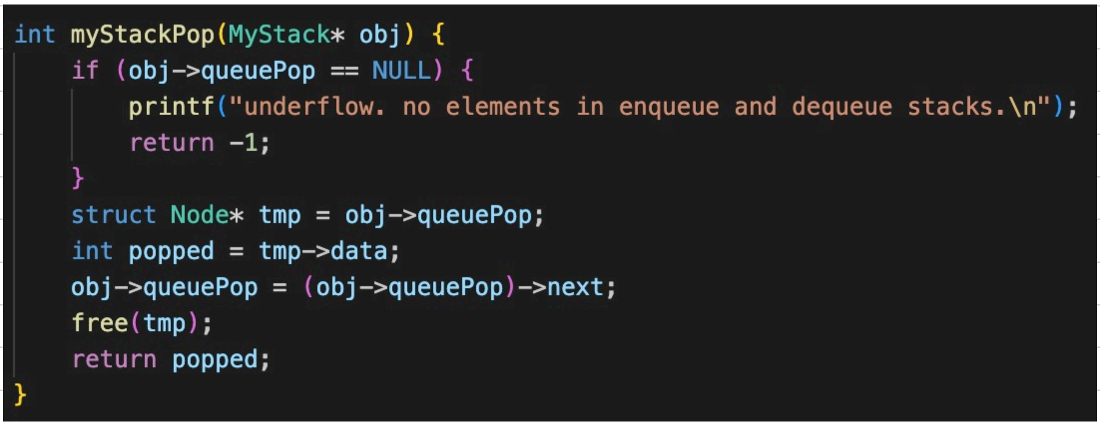


Skip

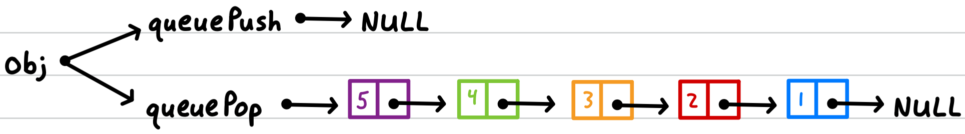
↓

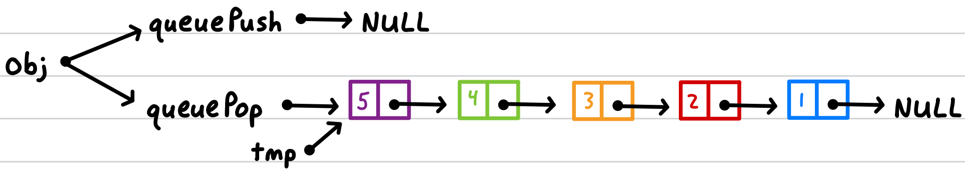
Final

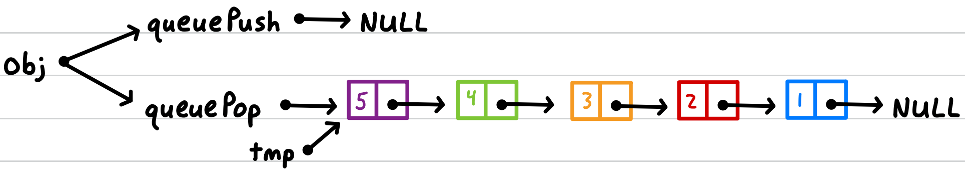




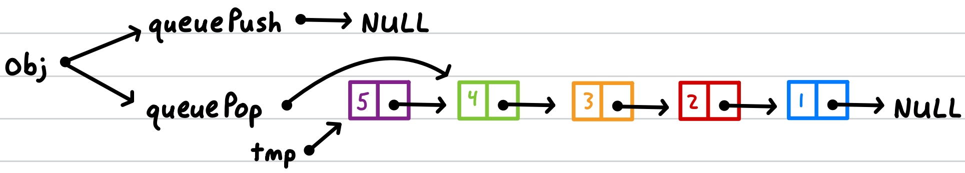
```
int myStackPop(MyStack* obj) {
    if (obj->queuePop == NULL) {
        printf("underflow. no elements in enqueue and dequeue stacks.\n");
        return -1;
    }
    struct Node* tmp = obj->queuePop;
    int popped = tmp->data;
    obj->queuePop = (obj->queuePop)->next;
    free(tmp);
    return popped;
}
```



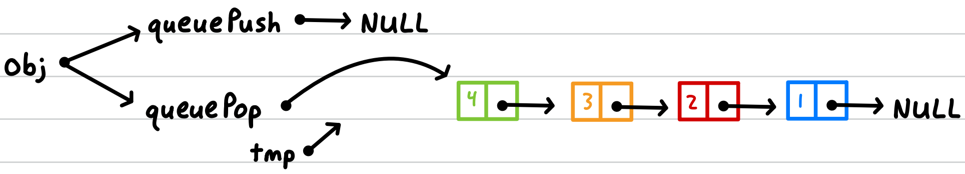




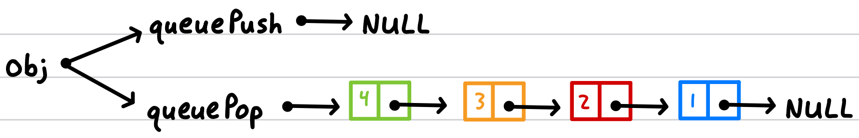
popped = 5



popped = 5

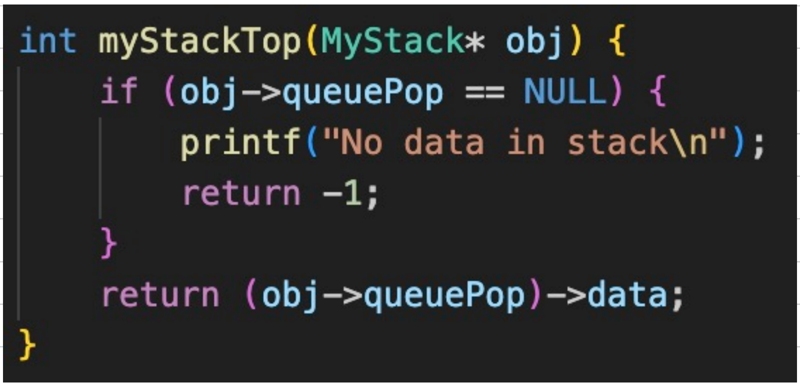


popped = 5

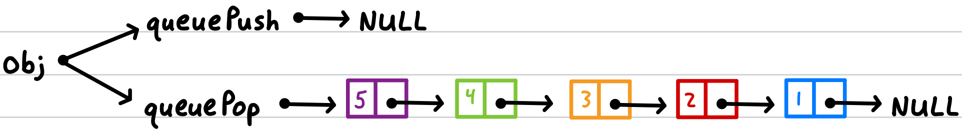


popped = 5

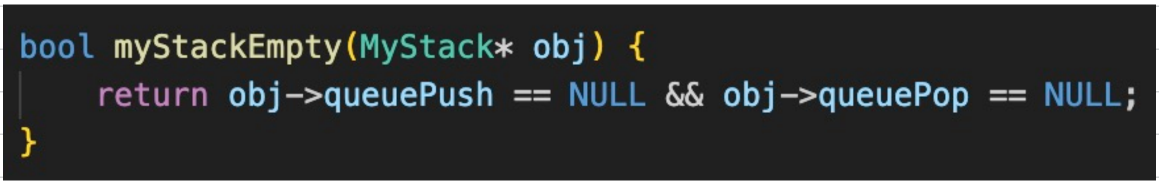
return popped



```c
int myStackTop(MyStack* obj) {
    if (obj->queuePop == NULL) {
        printf("No data in stack\n");
        return -1;
    }
    return (obj->queuePop)->data;
}
```

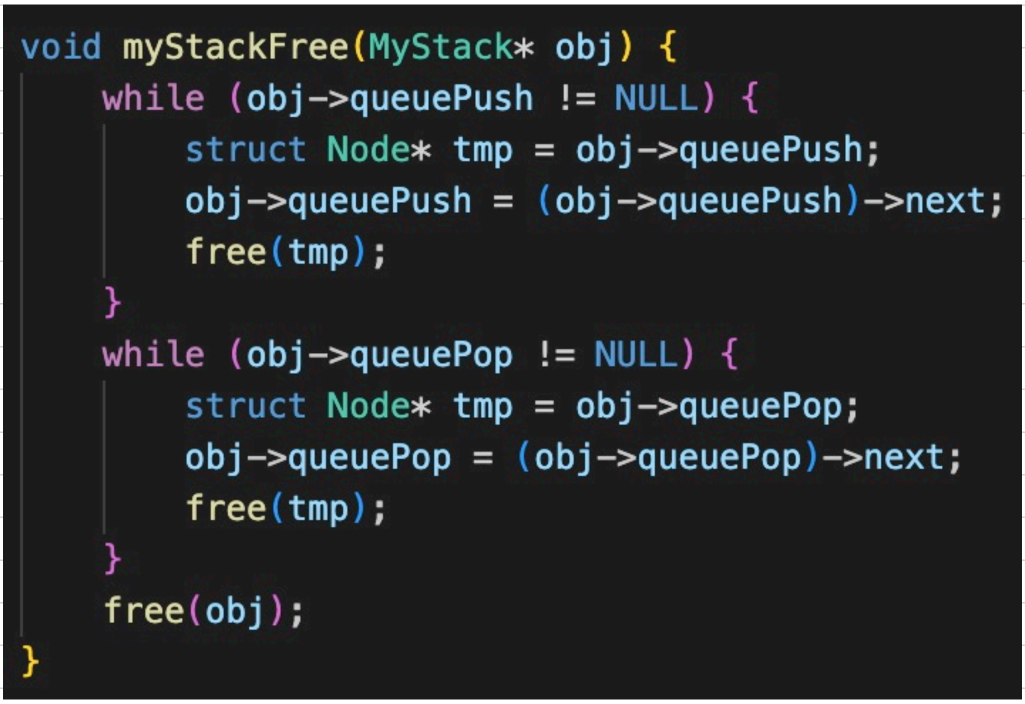


return 5

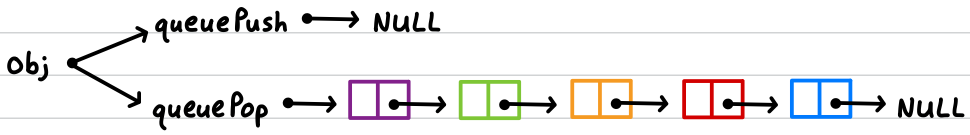


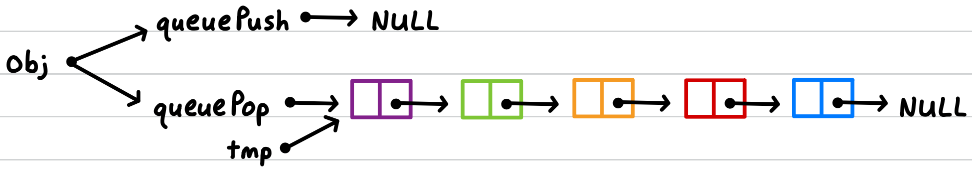
```c
bool myStackEmpty(MyStack* obj) {
    return obj->queuePush == NULL && obj->queuePop == NULL;
}
```

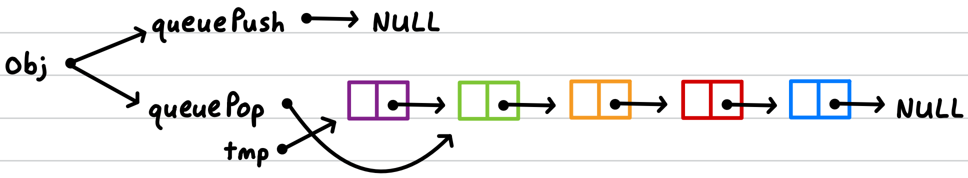
True ⟶ [queuePush ⟶ NULL] ∧ [queuePop ⟶ NULL]

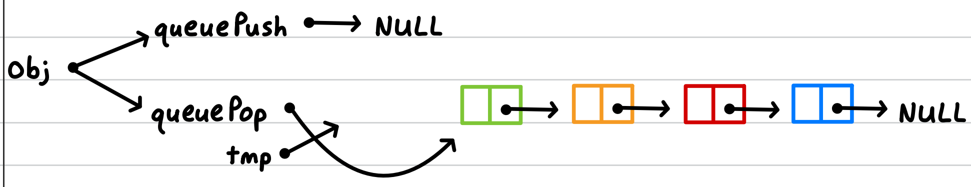


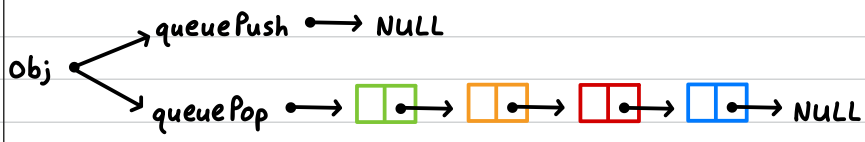
```c
void myStackFree(MyStack* obj) {
    while (obj->queuePush != NULL) {
        struct Node* tmp = obj->queuePush;
        obj->queuePush = (obj->queuePush)->next;
        free(tmp);
    }
    while (obj->queuePop != NULL) {
        struct Node* tmp = obj->queuePop;
        obj->queuePop = (obj->queuePop)->next;
        free(tmp);
    }
    free(obj);
}
```





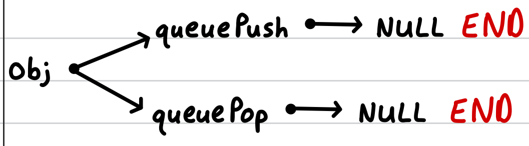


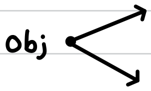




while obj→queuePush != NULL
      obj→queuePop != NULL

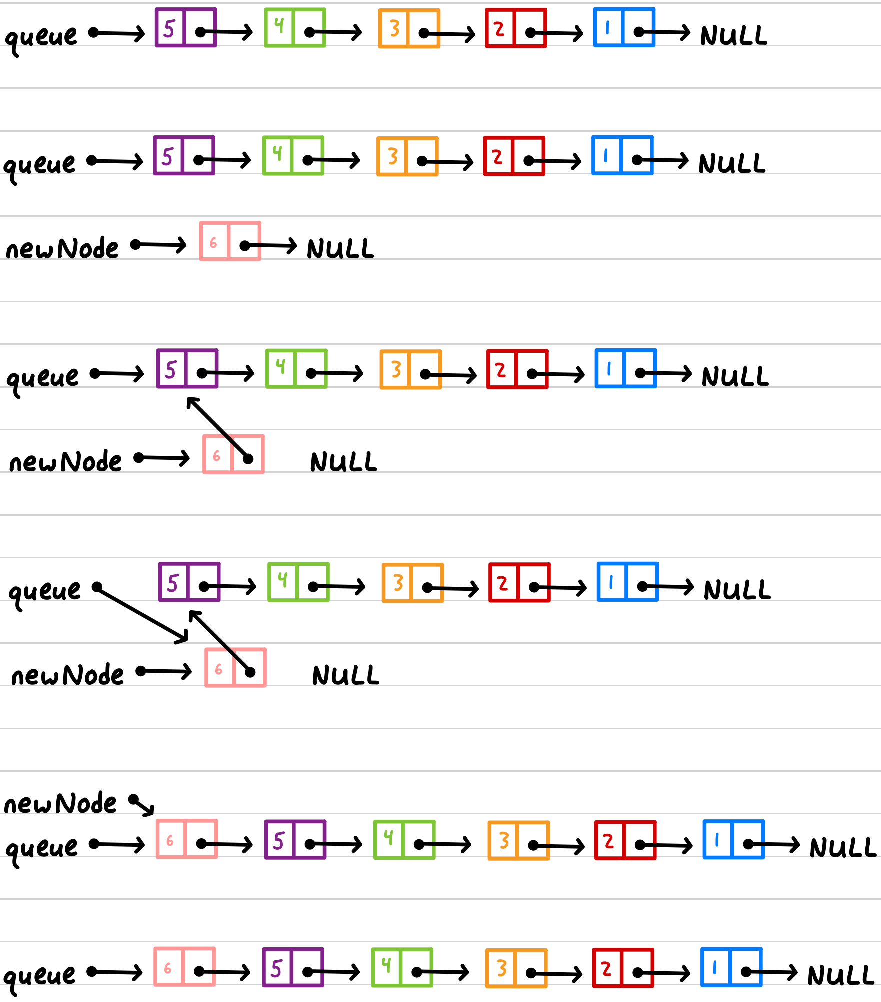
Final

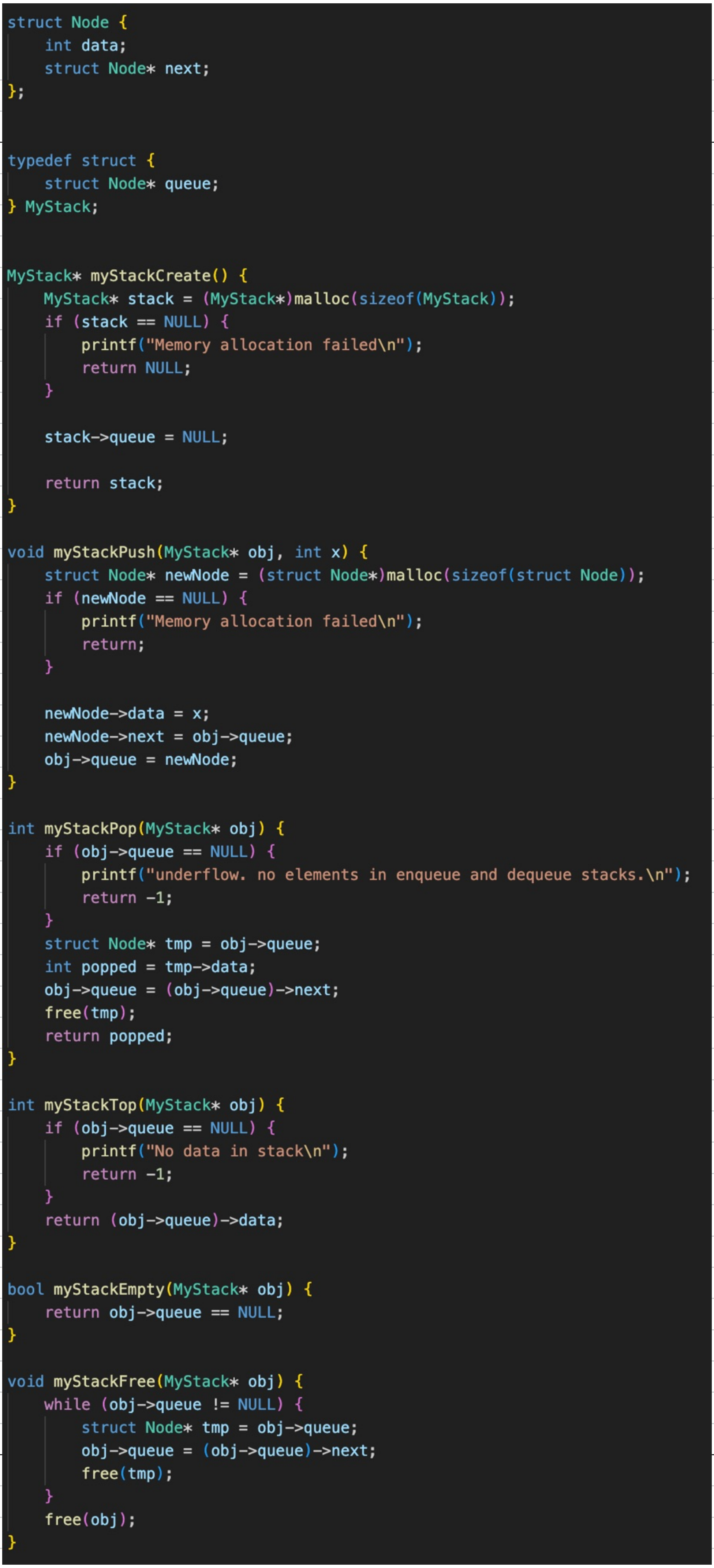




1 Queue Push

one queue





```c
struct Node {
    int data;
    struct Node* next;
};


typedef struct {
    struct Node* queue;
} MyStack;


MyStack* myStackCreate() {
    MyStack* stack = (MyStack*)malloc(sizeof(MyStack));
    if (stack == NULL) {
        printf("Memory allocation failed\n");
        return NULL;
    }

    stack->queue = NULL;

    return stack;
}

void myStackPush(MyStack* obj, int x) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    if (newNode == NULL) {
        printf("Memory allocation failed\n");
        return;
    }

    newNode->data = x;
    newNode->next = obj->queue;
    obj->queue = newNode;
}

int myStackPop(MyStack* obj) {
    if (obj->queue == NULL) {
        printf("underflow. no elements in enqueue and dequeue stacks.\n");
        return -1;
    }
    struct Node* tmp = obj->queue;
    int popped = tmp->data;
    obj->queue = (obj->queue)->next;
    free(tmp);
    return popped;
}

int myStackTop(MyStack* obj) {
    if (obj->queue == NULL) {
        printf("No data in stack\n");
        return -1;
    }
    return (obj->queue)->data;
}

bool myStackEmpty(MyStack* obj) {
    return obj->queue == NULL;
}

void myStackFree(MyStack* obj) {
    while (obj->queue != NULL) {
        struct Node* tmp = obj->queue;
        obj->queue = (obj->queue)->next;
        free(tmp);
    }
    free(obj);
}
```