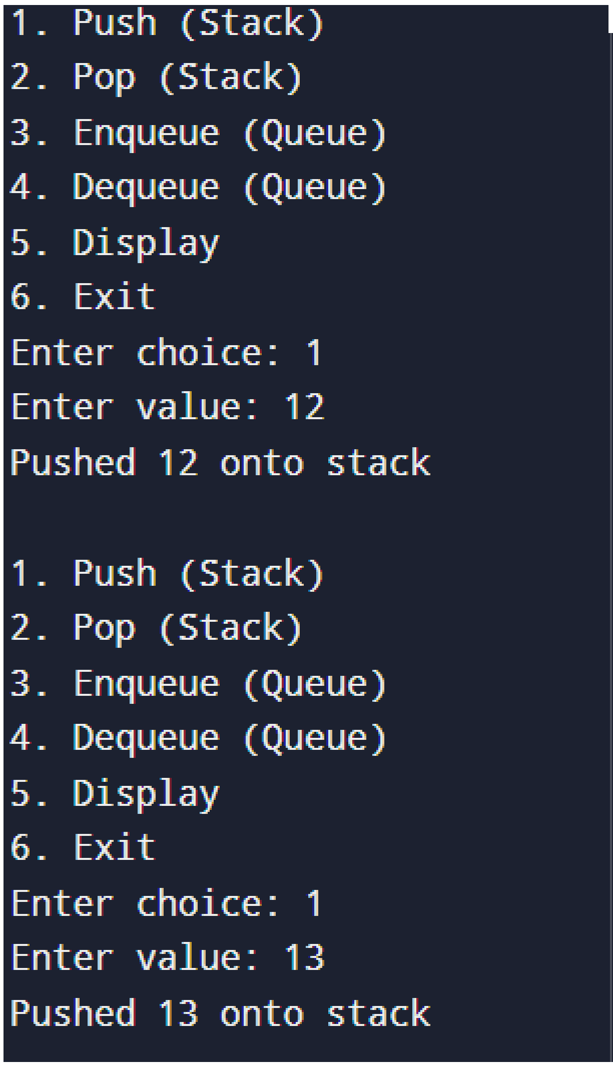
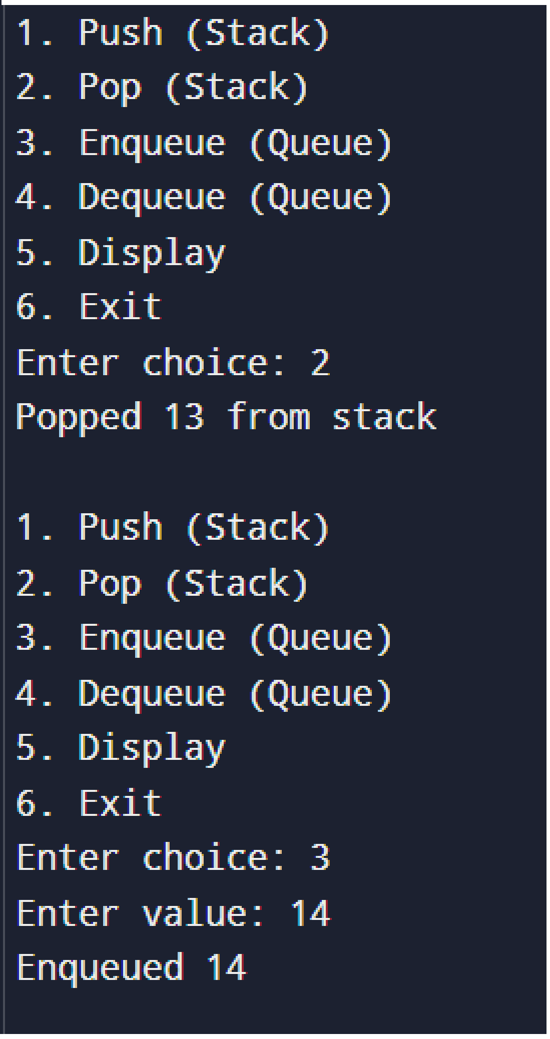
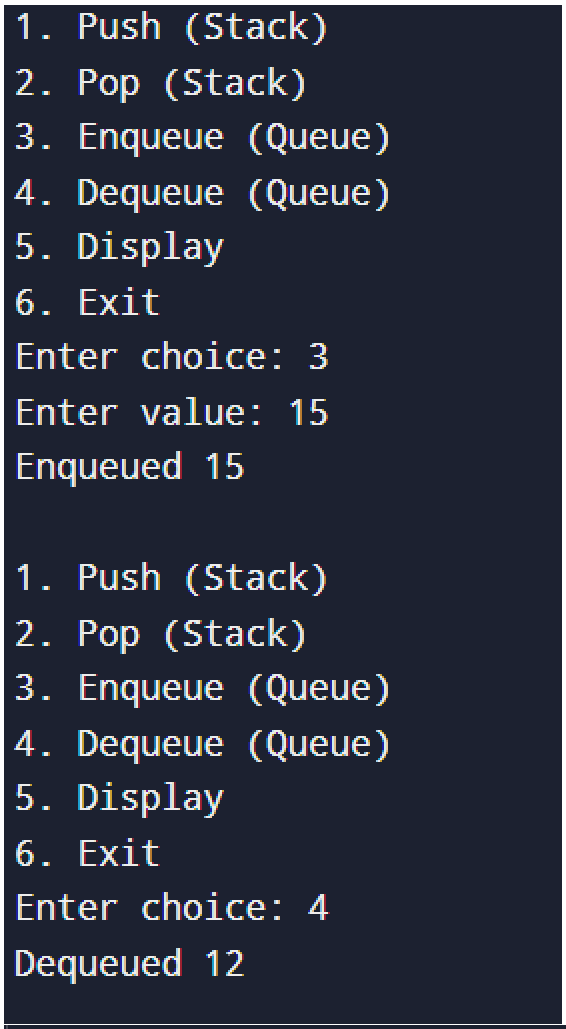
```c
#include <stdio.h>
#include <stdlib.h>

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

struct Node* head = NULL;

void push(int value) {
   struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
   newNode->data = value;
   newNode->next = head;
   head = newNode;
}

int pop() {
   if (head == NULL) return -1;
   struct Node* temp = head;
   int val = temp->data;
   head = head->next;
   free(temp);
   return val;
}

void enqueue(int value) {
   struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
   newNode->data = value;
   newNode->next = NULL;

   if (head == NULL) {
      head = newNode;
      return;
   }
   struct Node* temp = head;
   while (temp->next != NULL)
      temp = temp->next;
   temp->next = newNode;
}

int dequeue() {
   if (head == NULL) return -1;
   struct Node* temp = head;
   int val = temp->data;
   head = head->next;
   free(temp);
   return val;
}

void display() {
    if (head == NULL) {
        printf("List is empty!\n");
        return;
    }

    struct Node* temp = head;
    printf("Linked List: ");
    while (temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
}

int main() {
    int choice, value;

    while (1) {
        printf("\n1. Push (Stack)\n");
        printf("2. Pop (Stack)\n");
        printf("3. Enqueue (Queue)\n");
        printf("4. Dequeue (Queue)\n");
        printf("5. Display\n");
        printf("6. Exit\n");
        printf("Enter choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter value: ");
                scanf("%d", &value);
                push(value);
                printf("Pushed %d onto stack\n", value);
                break;

            case 2:
                value = pop();
                if (value == -1)
                    printf("Stack is empty!\n");
                else
                    printf("Popped %d from stack\n", value);
                break;

            case 3:
                printf("Enter value: ");
                scanf("%d", &value);
                enqueue(value);
                printf("Enqueued %d\n", value);
                break;

            case 4:
                value = dequeue();
                if (value == -1)
                    printf("Queue is empty!\n");
                else
                    printf("Dequeued %d\n", value);
                break;

            case 5:
                display();
                break;

            case 6:
                exit(0);

            default:
                printf("Invalid choice!\n");
        }
    }

    return 0;
}
```



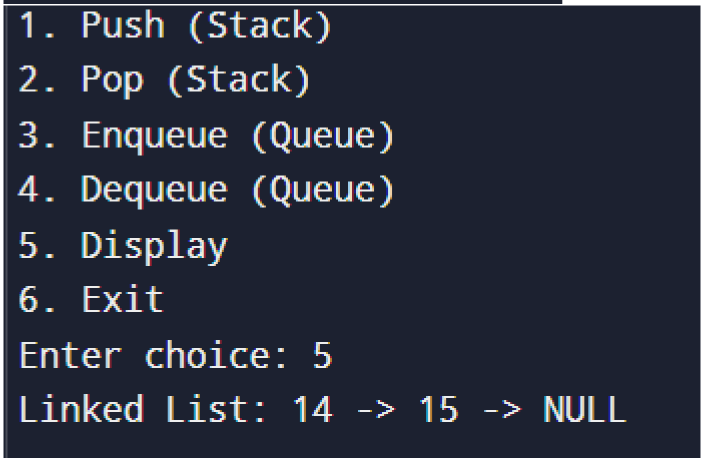
```
1. Push (Stack)
2. Pop (Stack)
3. Enqueue (Queue)
4. Dequeue (Queue)
5. Display
6. Exit
Enter choice: 1
Enter value: 12
Pushed 12 onto stack

1. Push (Stack)
2. Pop (Stack)
3. Enqueue (Queue)
4. Dequeue (Queue)
5. Display
6. Exit
Enter choice: 1
Enter value: 13
Pushed 13 onto stack
```



```
1. Push (Stack)
2. Pop (Stack)
3. Enqueue (Queue)
4. Dequeue (Queue)
5. Display
6. Exit
Enter choice: 2
Popped 13 from stack

1. Push (Stack)
2. Pop (Stack)
3. Enqueue (Queue)
4. Dequeue (Queue)
5. Display
6. Exit
Enter choice: 3
Enter value: 14
Enqueued 14
```



```
1. Push (Stack)
2. Pop (Stack)
3. Enqueue (Queue)
4. Dequeue (Queue)
5. Display
6. Exit
Enter choice: 3
Enter value: 15
Enqueued 15

1. Push (Stack)
2. Pop (Stack)
3. Enqueue (Queue)
4. Dequeue (Queue)
5. Display
6. Exit
Enter choice: 4
Dequeued 12
```



```
1. Push (Stack)
2. Pop (Stack)
3. Enqueue (Queue)
4. Dequeue (Queue)
5. Display
6. Exit
Enter choice: 5
Linked List: 14 -> 15 -> NULL
```