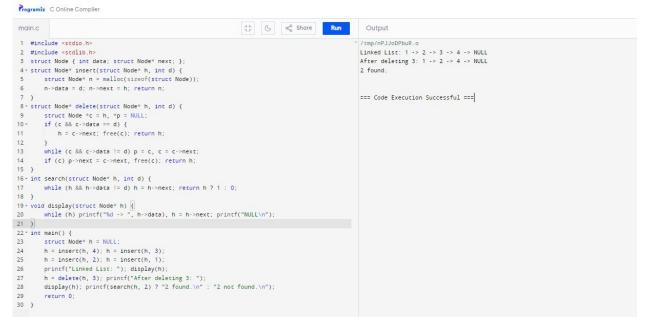
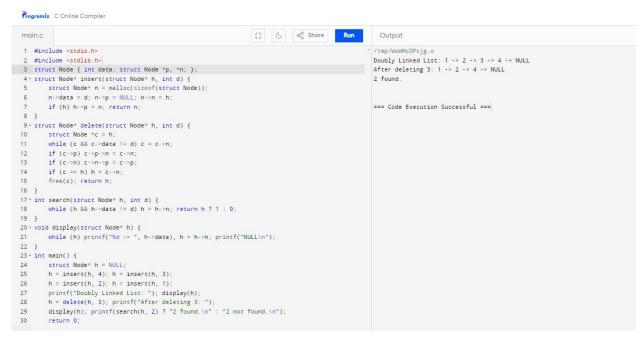
1) Write a c program for linked list.

```
Programiz C Online Compiler
                                                                                             Share Run
                                                                                                                                              /tmp/nPJJoDPbuR.o
   1 #include <stdio.h>
                                                                                                                                              Linked List: 1 -> 2 -> 3 -> 4 -> NULL
   3 struct Node { int data; struct Node* next; };
4 * struct Node* insert(struct Node* h, int d) {
                                                                                                                                              After deleting 3: 1 -> 2 -> 4 -> NULL
  5    struct Node* n = malloc(sizeof(struct Node));
6    n->data = d; n->next = h; return n;
                                                                                                                                              === Code Execution Successful ===
  8 - struct Node* delete(struct Node* h, int d) {
 9 struct Node *c = h, *p = NULL;
10 * if (c && c->data == d) {
                h = c->next; free(c); return h;
            while (c && c->data != d) p = c, c = c->next;
            if (c) p->next = c->next, free(c); return h;
  17 while (h && h->data != d) h = h->next; return h ? 1 : 0; 18 }
19 · void display(struct Node* h) { 20 while (h) printf("%d -> ", h->data), h = h->next; printf("NULL\n"); 21 }
 22 * int main() {
23    struct Node* h = NULL;
            struct Node: n = Note,
h = insert(h, 4); h = insert(h, 3);
h = insert(h, 2); h = insert(h, 1);
printf("Linked List: "); display(h);
h = delete(h, 3); printf("After deleting 3: ");
display(h); printf(search(h, 2) ? "2 found.\n" : "2 not found.\n");
            return 0;
```

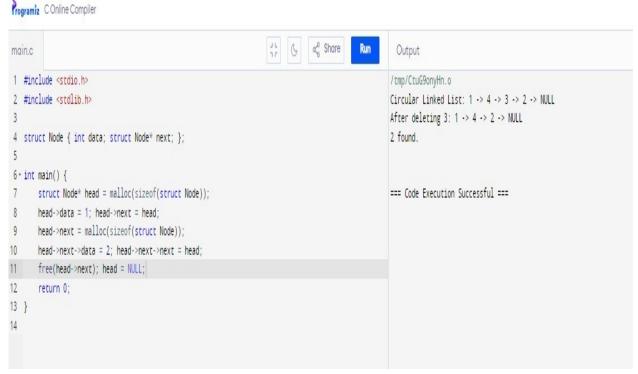
2) Write a c program for single linked list.



3) Write a c program for double linked list.



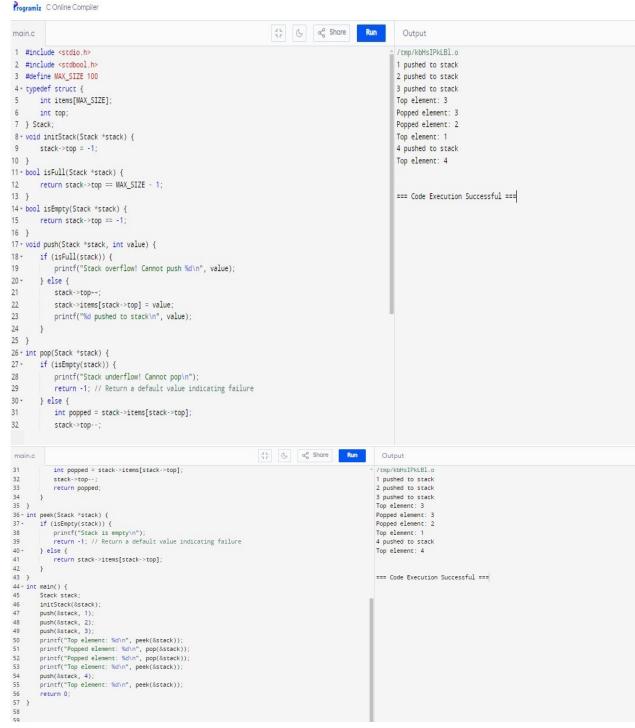
4) Write a c program circular linked list.



5) Write a c program stack using all operations push, pop, peek, full, empty.

```
1 #include <stdio.h>
2 #include <stdbool.h>
3 #define MAX_SIZE 100
4- typedef struct {
5 inclines [MAX_SIZE];
6 inclines [MAX_SIZE];
                                                                                                                           /tmp/hPXDwM2lVp.o
                                                                                                                           Top element: 3
                                                                                                                           Popped element: 3
Popped element: 2
Top element: 1
 6 int top;
7 } Stack;
8 * void initStack(Stack *stack) {
                                                                                                                           Top element: 4
=== Code Execution Successful ===
return stack->items[stack->top--];
}
29
30
31 }
32 - int peek(Stack *stack) {
32 - int peek(Stack *stack) {
33 +
34
35
36 +
         if (isEmpty(stack)) {
   printf("Stack is empty\n");
   return -1;
                                                                                                                                     Top element: 3
Popped element: 3
                                                                                                                                     Popped element: 2
Top element: 1
        return stack->items[stack->top];
}
         } else {
37
38
39 }
40 - int main() {
41 Stack stack;
                                                                                                                                     === Code Execution Successful ===
         initStack(&stack);
push(&stack, 1);
44
45
          push(&stack, 2);
          push(&stack, 3);
         printf("Top element: %d\n", peek(&stack));
printf("Popped element: %d\n", pop(&stack));
printf("Popped element: %d\n", pop(&stack));
46
47
48
49
         printf("Top element: %d\n", peek(&stack));
push(&stack, 4);
          printf("Top element: %d\n", peek(&stack));
         return 0;
```

6) Write a c program for array implementation of stack.



7) Write a c program that implements a stack using linked list.

```
αο Share Run
                                                                                                                               Output
main.c
                                                                                                                               /tmp/kbHsIPkLBl.o
1 #include <stdio.h>
2 #include <stdbool.h>
                                                                                                                               1 pushed to stack
 3 #define MAX_SIZE 100
                                                                                                                              2 pushed to stack
 4 - typedef struct {
                                                                                                                              3 pushed to stack
 5 int items[MAX_SIZE];
                                                                                                                              Top element: 3
                                                                                                                              Popped element: 3
         int top;
                                                                                                                               Popped element: 2
8 - void initStack(Stack *stack) {
                                                                                                                              Top element: 1
                                                                                                                              4 pushed to stack
                                                                                                                              Top element: 4
11 - bool isFull(Stack *stack) {
12 return stack->top == MAX_SIZE - 1;
13 }
                                                                                                                              === Code Execution Successful ===
14 - bool isEmpty(Stack *stack) {
       return stack->top == -1;
15
16 }
17 - void push(Stack *stack, int value) {
18 -
       if (isFull(stack)) {
             printf("Stack overflow! Cannot push %d\n", value);
19
         } else {
           stack->top++;
21
       printf("%d pushed to stack\n", value);
}
22
23
24
25 }
26 - int pop(Stack *stack) {
27 -
        if (isEmpty(stack)) {
           printf("Stack underflow! Cannot pop\n");
return -1; // Return a default value indicating failure
28
29
30 -
         int popped = stack->items[stack->top];
stack->top--;
31
32
                                                                                 를 C 로 Share Run
31
              int popped = stack->items[stack->top];
                                                                                                                        /tmp/kbHsIPkLBl.o
32
33
             stack->top--;
return popped;
                                                                                                                       1 pushed to stack
2 pushed to stack
34 }
35 }
                                                                                                                       3 pushed to stack
Top element: 3
36 · int peek(Stack *stack) {
37 · if (isEmpty(stack)) {
                                                                                                                        Popped element: 3
                                                                                                                        Popped element: 2
38
39
         printf("Stack is empty\n");
return -1; // Return a default value indicating failure
                                                                                                                        Top element: 1
                                                                                                                        4 pushed to stack
41 return stack->items[stack->top];
42 }
43 }
                                                                                                                       === Code Execution Successful ===
44 - int main() {
45
46
         Stack stack;
         initStack(&stack);
         push(&stack, 1);
push(&stack, 2);
47
48
49
         push(&stack, 3);
         positions.fit.position printf("Top element: %d\n", peek(&stack));
printf("Popped element: %d\n", pop(&stack));
printf("Popped element: %d\n", pop(&stack));
printf("Top element: %d\n", peek(&stack));
50
51
52
53
         push(&stack, 4);
printf("Top element: %d\n", peek(&stack));
54
55
56
57 }
         return 0;
```