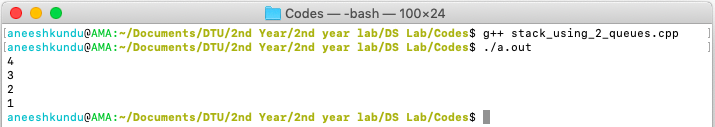
**Experiment 5**

**AIM:** Write a program to implement a stack using 2 queues.

**THEORY:** A stack is an abstract data structure in which insertion is done at the end and deletion is done at the end. It is LIFO i.e Last in First Out. It has 1 pointer top. Top is for deletion and insertion. In linked list we do not have to give the size and insertion and deletion is done using pointers. Create a stack to store operands (or values).

**Code:**

1. #include < iostream >
2. #include < queue >
3. using namespace std;
4. struct Stack {
5. queue < int > q1, q2;
6. void push(int v) {
7. bool e1 = !q1.empty(), e2 = !q2.empty();
8. if (e1 && e2)
9. q1.push(v);
10. else if (e1) {
11. q2.push(v);
12. while (!q1.empty()) {
13. q2.push(q1.front());
14. q1.pop();
15. }
16. } else {
17. q1.push(v);
18. while (!q2.empty()) {
19. q1.push(q2.front());
20. q2.pop();
21. }
22. }
23. }
24. int top() {
25. bool e1 = q1.empty(), e2 = q2.empty();
26. if (e1 && e2) return -1;
27. else if (e1) return q2.front();
28. else return q1.front();
29. }
30. void pop() {
31. bool e1 = q1.empty(), e2 = q2.empty();
32. if (e1 && e2) return;
33. else if (e1) return q2.pop();
34. else return q1.pop();
35. }
36. bool empty() {
37. return top() == -1;
38. }
39. };
40. int main() {
41. Stack s;
42. s.push(1);
43. s.push(2);
44. s.push(3);
45. s.push(4);
46. while (!s.empty()) {
47. cout << s.top() << endl;
48. s.pop();
49. }
50. return 0;
51. }

**Output:**