Div :- SE Comps B Roll No - 8953 Name :- Lucas

Page No.
Date 12 8 2020

Practice Queue

(1) Queue

(2) stock

(3) (a) push() (b) pop()

(4) (a) enqueue() (b) dequeue()

(5)

~~4~~ ← -44
~~66~~ 66
22
2

① push(s,2) ② push(&s, a)
③ push(s, a+b) ④ b = peek(&s); b = 66
⑤ pop(&s) ⑥ push(b) ⑦ push(&s, a-b)
⑧ ~~pop(s,s)~~ pop(&s)

```
while(!is Empty)
{ printf("%d", s top());
  pop(&s);
}
```

Op :- 66 22 2

7)

10
9
8
7
6
5
4
3
2
1

```
for(i=1; i<=10; i++)
    push(&s, i);
while(!is empty)
{ printf("%d", s.top());
```

Op :-
10 10 10 10 10 10 ... ---
(Infinite loop).

as there is no popping or hence there is never !is empty = 0
hence infinite loop

Page No.
Date

(5)

(i) Prefix notation: In this notation the operators are written before the operands

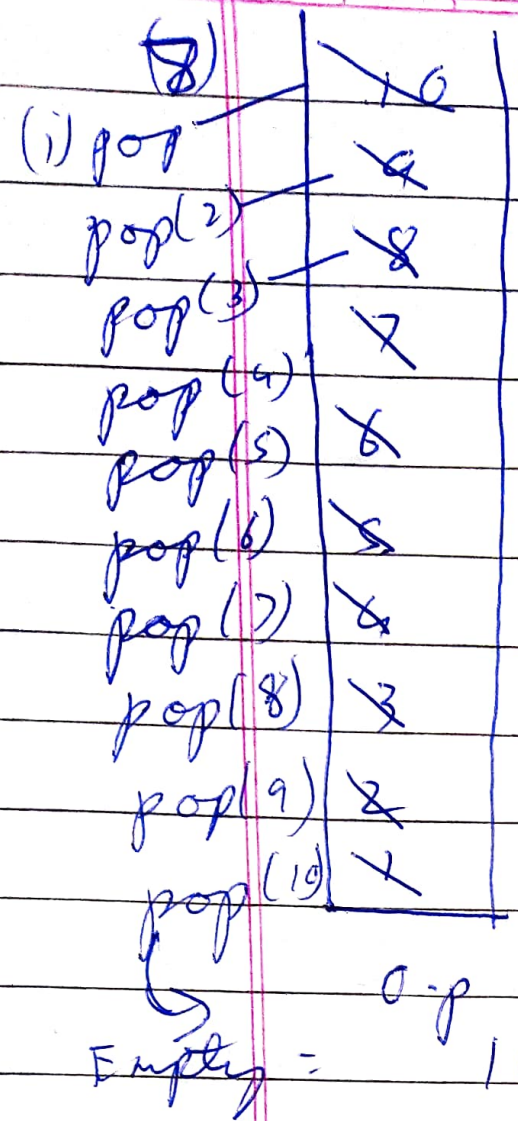
eg: $+A*BC$

(ii) Infix notation: In this notation the operators are written in between the operands.

eg: $A+B*C$

(iii) Postfix notation: In this notation the operators are written after the operands.

eg: $ABC*+$



← S.top

```
for (i = 0; i <= 10; i++)
    s.push(&s, i);
while (!s.isEmpty)
{
    printf("%d", s.top());
    s.pop();
}
```

o/p

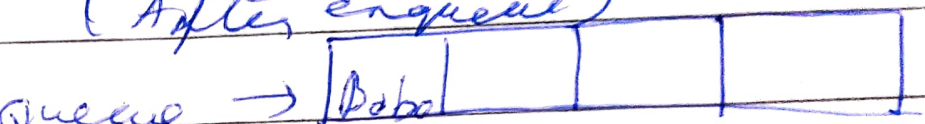
10 9 8 7 6 5 4 3 2 1

As !isEmpty = 0 the loop breaks and program terminates

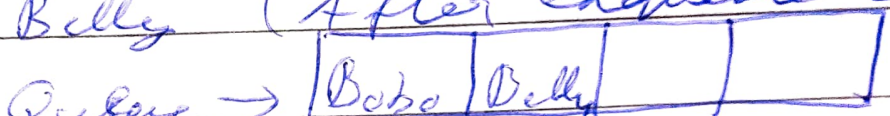
9 Output

(1) display(q) → Queue Underflow

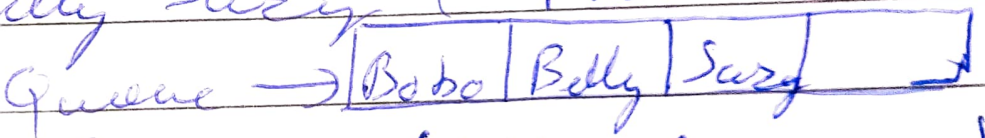
(2) display(q) → Bobo (After enqueue)



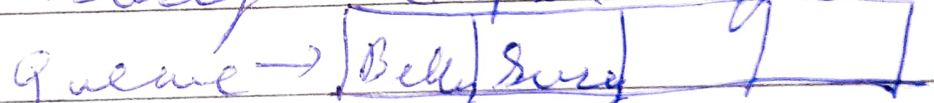
(3) display(q) → Bobo Billy (After enqueue 2)



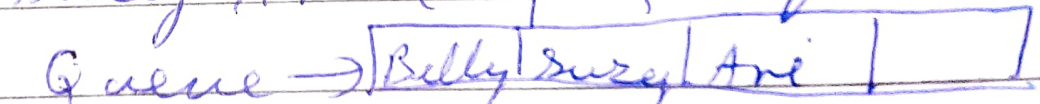
(4) display(q) → Bobo Billy Suzy (After enqueue 3)



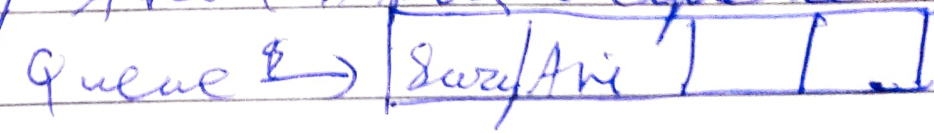
(5) display(q) → Billy Suzy (After dequeue)



(6) display(q) → Billy Suzy Ani (After enqueue 4)



(7) display(q) → Suzy Ani (After dequeue 2)

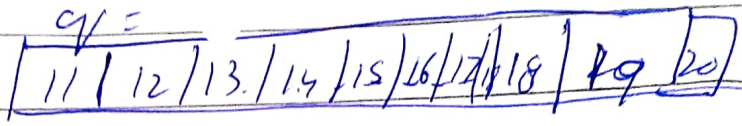


(10) Case 1 → (starting cond

stack

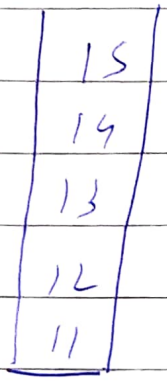
empty

q =

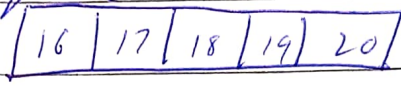


Case 2. (for i=0 is halfsize) → (push(s, deque(q

stack



queue

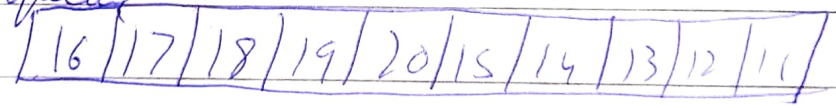


Case 3 (while stack is not empty) → (enque(q, pop(s

stack

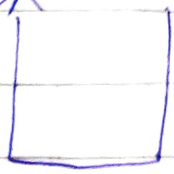
empty

queue



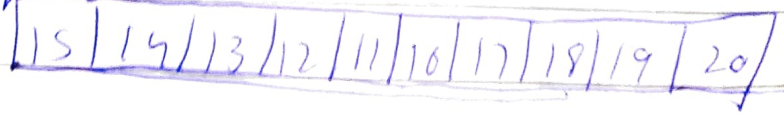
Case 4. (for i=0 is halfsize) → (enqueue(s, dequeue(q

stack



empty

queue



Page No.
Date

Case 5 (for (i=0, to half size) → (push(s, dequeue q)).

stack

11
12
13
14
15

queue

16	17	18	19	20

Case 6 (while stack is not empty) → (enqueue(q, pop(s)) // enque, deque).

stack empty

queue

11	16	12	17	13	18	14	19	15	20

final op:

→ 11 16 12 17 13 18 14 19 15 20.

(ii) char str[50];
gets(str); // let str = "rain"
int n = strlen(str); // n = 4.

for(i=0, i<n; i++) → push(&s, str[i]);

stack

n
i
a
r

for(i=0, i<n; i++) → str[i] = pop(&s);

str

n	i	a	r	'\0'

printf("%s", str);

op:-

n i a r // reverse of string.