Date:

Experiment 9

Queue

```
#include <stdio.h>
#include <stdlib.h>
#define size 5

int queue [SIZE];
int front = -1, rear = -1;

void insert (){
 int value;
 if(rear == size -1){
   printf("Queue is full (overflow)\n");
 }else{
   printf("Enter value to insert : ");
   scanf("%d", &value);
   if (front == -1){
    front =0;
    rear++;
   queue [rear] =value;
   printf(" Inserted %d \n", value);
   }
}
void delete (){
 if (front == -1 || front>rear){
 printf("Queue is empty (underflow) \n");
 }else{
 printf("Deleted %d \n", queue [front]);
 front ++;
 }
}
```

```
void display(){
if(front == -1 || front > rear){
}else{
    printf("Queue element : ");
    for(int i = front; i <= rear; i++){
    printf("%d", queue[i]);
}
    printf("\n");
}
}
int main(){
 int choice;
while(1){
 printf("\n1.Insert\n2.Delete\n3.Display\n
   4.Exit\n");
 printf("Enter your choice : ");
 scanf("%d", &choice);

switch(choice){
case 1: insert();
        break;

case 2:
    delete();
     break;

case 3:
     display();
     break;

case 4:
    exit(0);
default:
        printf("Invalid choice\n");
```

```
}
}
return 0;
}
```

O/P

1) Insert
2) Delete
3) Display
4) Exit

Enter your choice : 1
Enter a value to insert : 10
Inserted 10

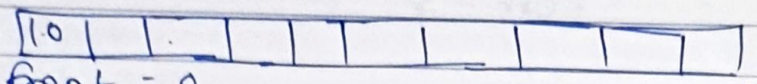
Menu . . .
Enter your choice : 1
Enter value to insert : 20
Inserted 20

Menu . . .
Enter your choice : 2
Deleted 10

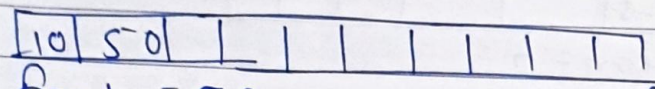
Menu . . .
Enter your choice : 4

Q) Perform following operation on linear queue in array size (10)

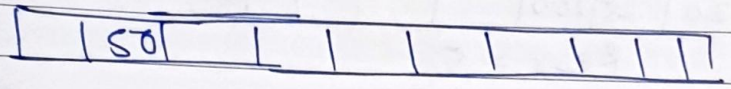
i) insert (10)



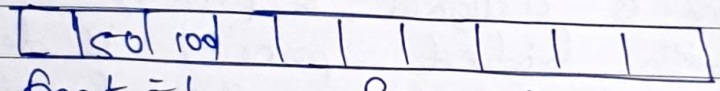
ii) Insert (50)



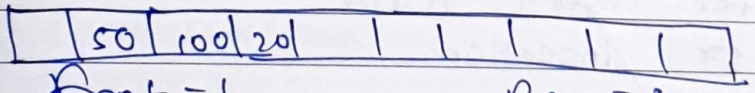
iii) Delete



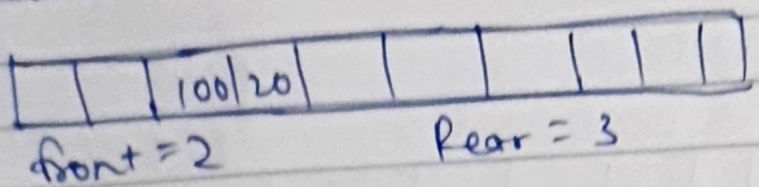
iv) Insert (100)



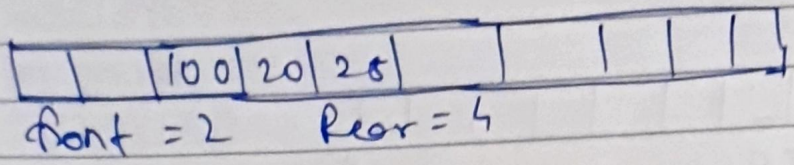
v) Insert (20)



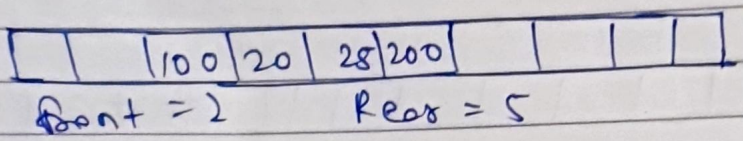
vii) Delete



Insert (25)



Insert (200)



3) Explain priority queue in detail.

→ Priority queue is a type of queue where each element is assigned a priority & the element with highest priority is delete before other elements regardless of order the were inserted. It do does not follow (FIFO)

4) Write a algorithm for insertion & deletion operation for insert queue

→ Algorithm for insertion.

Step 1

Check for the overflow.

Step 2.

Check is the queue is empty

Step 3
The value is sorted in the location pointed by rear.

Step 4.
Exit

for - deletion -
Step 1
Check for underflow condn on underflow occurs if front = -1 or front > rear however if queue has some value then front in incremented so that it now pointer to the next value in queue.
Step 2 - Exit.