NAME: HARSHBHAI SOLANKI
ROLL.N0:62
SE-4-D

EXPERIMENT N0-10

Every day several people want a reservation for a seat berth in a train. Let's think that only one birth is available and two passengers (threads) are asking for that birth. Let's assume that in reservation counter no.1, the clerk has sent a request to the server to allot that berth to his passenger. In counter no.2, the second clerk has also sent a request to the server to allot that birth to his passenger. It means two passengers are competing for the same birth. Write a Python program to ensure that seat is alloted to only one passenger using synchronization methods.

SOURCE CODE :-

```
from threading import *
class Train:
    def __init__(self,seat):
        self.seat=seat
        self.l=Lock()
    def reserve(self,need_seat):
        self.l.acquire(blocking=True)
        print(f"Available seats are: {self.seat}")
        if(self.seat >= need_seat):
            name=current_thread().name
            print(f"{need_seat} alloted to {name}")
            self.seat -= self.seat
        else:
            print("all seats are alloted")
        self.l.release()
```

```
if __name__ == '__main__':
    train = Train(1)
    name1=input("enter the name1:")
    name2 = input("enter the name2:")
    t1 = Thread(target=train.reserve,args=(1,),name=name1)
    t2 = Thread(target=train.reserve,args=(1,),name=name2)
    t1.start()
    t2.start()
```

OUTPUT: