**Day 28\_Java Assignment**

**1. Problem Description:**

ReentrantLock – Advanced multi-threading concept

**2. My Solution:**

**Account Main class:**

**public** **class** AccountMain **implements** Runnable {

**private** Account fromAccount;

**private** Account toAccount;

**private** **double** amount;

**public** AccountMain(Account fromAccount, Account toAccount, **double** amount) {

**this**.fromAccount = fromAccount;

**this**.toAccount = toAccount;

**this**.amount = amount;

}

**public** **static** **void** main(String[] args) {

Account mahaAccount = **new** Account(1, 100000, "Maha");

Account meenuAccount = **new** Account(2, 150000, "Meenu");

System.***out***.println(mahaAccount.getBalance());

System.***out***.println(meenuAccount.getBalance());

Thread thread1 = **new** Thread(**new** AccountMain(mahaAccount, meenuAccount, 5000));

Thread thread2 = **new** Thread(**new** AccountMain(meenuAccount, mahaAccount, 10000));

thread1.start();

thread2.start();

**try** {

thread1.join();

thread2.join();

} **catch** (InterruptedException e) {

e.printStackTrace();

}

System.***out***.println(mahaAccount.getBalance());

System.***out***.println(meenuAccount.getBalance());

}

@Override

**public** **void** run() {

fromAccount.transferMoney(toAccount, amount);

**try** {

Thread.*sleep*(100);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

**Account class:**

**import** java.util.concurrent.locks.Lock;

**import** java.util.concurrent.locks.ReentrantLock;

**public** **class** Account {

**private** **int** id;

**private** **double** balance;

**private** String name;

**private** **final** Lock lock;

**public** Account(**int** id, **double** balance, String name) {

**this**.id = id;

**this**.balance = balance;

**this**.name = name;

**this**.lock = **new** ReentrantLock();

}

**public** **int** getId() {

**return** id;

}

**public** **double** getBalance() {

**return** balance;

}

**public** String getName() {

**return** name;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** **void** setBalance(**double** balance) {

**this**.balance = balance;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **double** transferMoney(Account toAccount, **double** money) {

lock.lock();

**try** {

**if** (**this**.balance < money) {

System.***out***.println("Insufficient money");

} **else** {

**this**.balance -= money;

toAccount.setBalance(toAccount.getBalance() + money);

}

**return** **this**.balance;

} **finally** {

lock.unlock();

}

}

}

**Output:**

100000.0

150000.0

105000.0

145000.0