### *Giải thuật :*

Tại mỗi bờ, phà trải qua 2 giai đoạn :

Giai đoạn 1: Chờ đến khi có xe chờ nó ở 1 trong 2 bờ

Giai đoạn 2: (Nếu có) Chạy qua bờ đối diện nếu xe chờ nó ở đó

Giai đoạn 3: Load tối đa N xe đang **chờ** nó ở cùng bờ.

### *Cài đặt :*

- List/Array queue gồm 2 phần tử để biểu xe đang chờ hai bên bờ.

- curSide => bờ phà đang ở (0: phà đang ở bờ bên trái, 1: phà đang ở bên phải )

0 => 1-0

1 => 1-1

chuyển bờ : curSide = 1 - curSide

- curTime => thời điểm hiện tại

- Array ans => thời điểm mỗi xe đến được bờ bên kia

### *Code:*

class Car {

int arrTime;

int idx;

}

function main() {

input C

for(idxTest : 1 => C) {

input N, T, M

Queue queue[2]

for(i : 1 => M) {

input arrivalTime, side

if (side == ‘left’) {

queue[0].push(new Car(arrivalTime, i)

} else {

queue[1].push(new Car(arrivalTime, i)

}

}

int curTime = 0;

int curSide = 0;

while(!queue[0].empty() && !queue[1].empty()) { --- O(2\*M)

int leftBank = queue[0].empty() ? INF : queue[0].top().arrTime

int rightBank = queue[1].empty() ? INF: queue[1].top().arrTime

int firstArrival = min(leftBank, rightBank)

curTime = max(curTime, firstArrival)

int loadedCar = 0;

while(!queue[curSide].empty()

&& queue[curSide].top().arrTime <= curTime

&& loadedCar < N) {

loadedCar += 1;

int id = queue[curSide].top().idx

ans[id] = curTime + T

}

curSide = 1 - curSide

curTime += T

}

for(int i : 1 => M) {

output ans[i] + “ “

}

if(idxTest < C) {

output “\n”

}

}

}

Độ phức tạp O(M \* C)