```
(4) func maxProfit (N, vector<pair<int int>>& projects).
        sort the list of projects using cmp function
          maxDeadline = 0
       For each project in projects:
          if project.deadline > maxDeadline
                  maxDeadline = project.deadline
          Create a list 'slot' Of size (maxDeadline +1) init to
                                                        false.
          totalProfit = 0

   For each project in projects:
       deadline = project.deadline
         profit = project.profit
         For day from deadline down to 1:
            If slot [day] is false:
               Slot [day] = true ||
                 totalprofit += profit
               break

          return totalprofit
```

(4) DRY RUN :- Problem ①:- The Launch Day Puzzle.

i/p:- Projects = [ (2,100), (1,19) (2,27) ].

① sort by profit descending:

sorted: [(2,100), (2,27), (1,19)].

② initialize slot array.

max Deadline = 2 → Slot = [False, False, False] func maxProfit (pair<int,int>a, pa

Step3: Assign project:

(2,100) try day 2 → slot[2] is free → assign → profit += 100
(2,27) : try day 2 → slot[2] taken → try day 1 → slot[1] free → assign → profit += 27

(1,19) : slot[1] is already taken → skip.

final profit:

100 + 27 = 127

Ans.

(7) The VIP commute cousin Toll gate optimization.

```
Func mintotal (T, VIP, W):
    init list of cars with (arrivalTime, isVIP,
                            processed = false).
    set currentTime = 0, totalwait = 0, processedcount = 0.

    While not all cars are processed:
    available cars: all unprocessed cars that have
    arrived ( arrivalTime ≤ currenttime ).

        chosen = -1

     For car in availablecars:

        if car is VIP and (currenttime - arrivalTime
                                        > W ):
              Choose the earliest such VIP car.

     if no urgent VIP:
          choose car with earliest arrivalTime from
            available cars.

              if no car chosen:
                   currentTime += 1.
                   continue.
              totalwait += (Time - AT.
                mark car as processed
                   CT += 1
                     pcount += 1
                return totalwait.
```

(7) DRY RUN:-

time =0 , car 0→wait =0

time =1 car1 →wait = 0

time =2 car2 →wait=0

time =3 car3 →wait= 0.