Content –

- Reallocation Strategy
- Roadblock and modification used
- Improvement to get best solution: Brownie part (I felt very happy after finding this)

REALLOCATION STRATEGY:

1) Graph(BFS)-First we make a dictionary with key = departing airport and value a list of airports with direct flight from key.
2) Then we make 3 dictionaries with key = cancelled flight FID-
   a) I store if there is a direct flight or not
   b) II store nodes from where there is one layover possible
   c) III store pair of nodes where there are two layovers possible
3) Then one dictionary having key as cancelled flights FID and value has a list of two lists each list having tuples of (2 or 3) flight from connecting nodes – with a given condition of layover of at least 1 hour* [roadblock]
4) Then we do some sorting of flights allocation in such a way that the cancelled flight with least reallocation possible will be chosen first
5) Then we provide reallocate n passengers of a cancelled flight to flights of new route such that n = min(vacant spaces in both/three flights) and run this is in loop for every flight

ROADBLOCK AND MODIFICATION USED:

On applying a condition that a flight can be reallocated only if it's departure time is after the departure time of cancelled flight then I am getting no possible reallocation for any cancelled flight.

I try then checking it manually and found that actually it's true we won't get any possible flight then so I removed this condition.

This can be because the timings are allocated by a random generator in source code.

IMPROVEMENT TO GET BEST SOLUTION:

1) This is the main part of the solution.
2) I get a thought that the order of available flights I choose to give to the passengers may affect total passengers reallocated due to a simple reason-

   Let say from A to B there is a possible way A – C – D – B and flights used 200, 201, 202, the 201 flight is from C to D.
   Now for E to F there are two possible way E – C – D – F and E – M – N – F now if while filling E to F and choose E – C – D – F path then the flight 201 will be used here but if we use E – M – N – F we can reallocate this passenger also and flight 201 still have space so route A – C – D – B will also be used
3) This is just a small example if randomness is introduced we can not only just increase allocated passengers but also reduce the average layoff and time diff by using a simple ML type mathematical method but we will not directly apply ML we will run our solution in loop by applying randomness in order every time and update minimum value of the factor and try to output that minimum value solution after fixed iterations

   Maths – I have here taken benefit of randomness that the probability of getting best solution atleast once will increase if we make more iteration it is simple probability

Probability of getting best solution in n iterations (assume probability of best solution is 1/K where K is total possible permutations of reallocation) = $1-\left(1-\frac{1}{k}\right)^n$ here if we increase n to extent of k we will get probability near to n/k

4) I will attach snap of my code to explain my procedure in achieving so

```
N = 2000
reallocation_factor = 0
layover_factor = 10
timediff_factor = 100000000
output_variable = 1000
for z in range(N):
```

Run Cell | Run Above | Debug Cell

We will run code 2000 times where first 1000 iterations focus on getting best solution and in next 1000 iterations we will stop our code on that iteration when we reach that best solution again

```
    l = layover_count/managed_reallocation
    a = absolute_time_diffrence/managed_reallocation
    \
    reallocation_factor = max(reallocation_factor,managed_reallocation)
    timediff_factor = (timediff_factor*z + a)/(z+1)
    layover_factor = (layover_factor*z + l)/(z+1)
    new_output = 200*(l/layover_factor + a/timediff_factor)
    print(z,output_variable,new_output)
    if z <1000:
        output_variable = min(output_variable,new_output)
    else:
        if abs(new_output-output_variable)<=4 and reallocation_factor==managed_reallocation:
            break
```

Run Cell | Run Above | Debug Cell

Here output variable is our measure of solution where we are minimizing the average layover and time diff

I had used moving average in layover factor and timediff factor to get best result
In break condition I have used less than 4 to get nearly best solution as I don't want my code to return no output if it doesn't get the best solution again as probability of getting it again is very rare.