Code Jam 2015 - Round 1A

Analysis: Mushroom Monster

Each method can be solved independently because they answer different questions. With the first method, Kaylin can eat any number of mushroom pieces at any time. Since the only way mushrooms can leave the plate is by being eaten, whenever we see a decrease in the number of mushrooms (in an interval) it must be because they were eaten by Kaylin. The minimum number of mushrooms Kaylin could have eaten using this method is the sum of the observable decreases for each interval. Since we only care about how many mushrooms Kaylin ate, we do not need to calculate how many mushrooms Bartholomew added.

With the second method, Kaylin always eats mushrooms at a constant rate whenever there are mushrooms on her plate. For each interval, we can observe Kaylinâ€™s eat-rate (i.e., the decrease of the number of mushrooms for that time interval). Since we want to find the minimum number of mushrooms Kaylin could have eaten, **we should find Kaylinâ€™s minimum eat-rate**. Since the eat-rate must be constant for each interval from the beginning until the end, **only the highest observable eat-rate makes sense**. It may appear that Kaylin eats fewer mushrooms than her eat-rate in some intervals, either because her plate becomes empty during the interval and she stops eating, or because Bartholomew added more mushrooms during the interval.

The number of mushrooms Kaylin could have eaten using the second method is the sum of min(M[i], **max_rate**) for all intervals i, where M[i] is the number of mushrooms at the beginning of interval i and **max_rate** is the highest observable eat-rate. That is, if at the beginning of the interval the number of mushrooms is larger than the maximum eat-rate, Kaylin can only eat **max_rate** mushrooms, otherwise Kaylin can only eat M[i] mushrooms and the plate becomes empty until the end of that interval. Note that we donâ€™t care about the number of mushrooms at the end of an interval. Since we want to minimize the eat-rate, we should assume that Bartholomew puts in mushrooms instantaneously at the end of the interval to maximize Kaylinâ€™s idle time.

Below is a sample implementation in Python:

```
def first_method(M, N):
  min_eat = 0
  for i in range(1, N):
    min_eat += max(0, M[i - 1] - M[i])
  return min_eat

def second_method(M, N):
  max_rate = 0
  for i in range(1, N):
    max_rate = max(max_rate, M[i - 1] - M[i])

  min_eat = 0
  # exclude the last mushroom
  for i in range(0, N - 1):
    min_eat += min(M[i], max_rate)
  return min_eat

for tc in range(int(input())):
  N = int(input())
  M = map(int, raw_input().split())
```

```
print "Case #%d: %d %d" % (tc + 1,
  first_method(M, N), second_method(M, N))
```