Coding Competitions Farewell Rounds - Round A

Analysis: ASCII Art

Test Set 1

For relatively small values of $\mathbf{N}$, it suffices to simulate the program in the problem statement to generate characters accordingly and output whichever character gets generated at the $\mathbf{N}$-th position. Since each character is generated in order and no additional calculation needs to be done, the time complexity of this solution is $O(\mathbf{N})$, which suffices for Test Set 1.

Test Set 2

The solution described for Test Set 1 would fail for Test Set 2 as $\mathbf{N}$ is significantly larger here. We would need to utilize the pattern of the characters to improve on it.

One observation we can make is that if we know which iteration of Cody-Jamal's program printed out the $\mathbf{N}$-th character (let us say it is the i-th iteration) and how many characters were already printed before that (let us say M characters were already printed before the i-th iteration), we can easily figure out which character is the $\mathbf{N}$-th character.

Since we know M, we can say that our answer is the $(\mathbf{N} - M)$-th character printed in the i-th iteration. Let's define $k = \mathbf{N} - M$. Clearly, if $k \leq i$, the character is `A`, if $i < k \leq 2i$, the character is `B` and so on. Simply put, the $\mathbf{N}$-th character printed is the $\lceil \frac{k}{i} \rceil$-th letter of the alphabet, i.e. the $\lceil \frac{\mathbf{N}-M}{i} \rceil$-th letter of the alphabet. Also note that if we know i alone, we can compute M as $26 \times \sum_{j=1}^{i-1} j$. All that remains is to determine i, which can be done in the following ways:

Linear Search

We essentially need to find the largest i such that $M = 26 \times \sum_{j=1}^{i-1} j < \mathbf{N}$. We can simply compute this sum in order until the point where $26 \times \sum_{j=1}^{i-1} j = 13 \times i \times (i-1) \geq \mathbf{N}$. The time complexity for this approach is $O(\sqrt{\mathbf{N}})$ since $13 \times i \times (i-1)$ increases quadratically.

Binary Search

The previous approach can be sped up by utilizing the fact that $13 \times i \times (i-1)$ monotonically increases, which allows us to binary search for i instead. The bounds for i to binary search on can be roughly set as 1 and $\mathbf{N}$. The time complexity in this case is $O(\log \mathbf{N})$.

Directly Calculating i

As mentioned earlier, we want to find the largest i such that $13 \times i \times (i-1) < \mathbf{N}$. Let z be the positive root of the quadratic equation $13 \times i \times (i-1) - \mathbf{N} = 0$. For $0 \leq i < z$, the value of $13 \times i \times (i-1) - \mathbf{N}$ is negative, after which it becomes positive. Thus, our problem reduces to finding the largest integer $i < z$, which is simply $\lceil z - 1 \rceil$. z can be calculated using the quadratic formula as $\frac{13+\sqrt{169+52\mathbf{N}}}{26} = \frac{1+\sqrt{1+\frac{4\mathbf{N}}{13}}}{2}$.