Spreadsheet Encoding

In this lesson, you will learn how to convert the column IDs in a spreadsheet into an integer in Python.

WE'LL COVER THE FOLLOWING ^

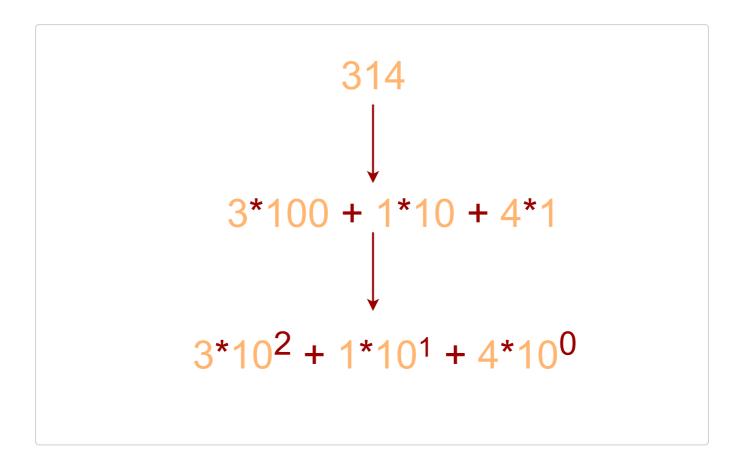
- Implementation
- Explanation

In this lesson, we will be considering how to solve the problem of implementing a function that converts a spreadsheet column ID (i.e., "A", "B", "C", …, "Z", "AA", etc.) to the corresponding integer. For example, "A" equals 1 because it represents the first column, while "AA" equals 27 because it represents the 27th column.

We will cover how to solve this problem algorithmically, and then code a solution to this question in Python.

The key insight is to think of the column IDs as base 26 integers.

In the illustration below, you can see how we represent "314" as a base 10 number.



From the illustration above, you can easily deduce the general formula. We start from the left-most digit and multiply it with the base raised to the power of (n-1) where n is the number of digits in that number. We continue with the same procedure with the other digits but keep subtracting one from the power of the base, so the power of the base will be 0 for the last digit. Finally, the sum of all the products is equivalent to the original number.

Now, let's go ahead and represent the numbers of base 26 according to the formula above. Here we go:

A = 1, B = 2, ..., Z = 26

AA

$$A*26^{1} + A*26^{0}$$
 $1*26^{1} + 1*26^{0} = 27$

Implementation

At this point, the base representation of a number will be pretty clear to you. Let's make use of it in the implementation in Python, but before we jump to the implementation, let's get introduced to ord function in Python:



ord() returns an integer which represents the Unicode code point of the Unicode character passed into the function. For the solution to work, we need to make sure of the following:

$$A = 1, B = 2,..., Z = 26$$

as we deal with base 26 numbers.

To make A represent 1, B represent 2, and so on, we need to determine the relative difference from the result given by the ord function and from the

representation we require for base 26 system. Now we know that ord('A')

equals 65. So if we find the Unicode code point using ord() of a character, subtract ord('A') from it, and then add 1 to it, we'll get the representation we want for the base 26 system. For instance, check out the code below for the characters that have been converted with this methodology:

```
print(ord('A')- ord('A')+ 1)
print(ord('B')- ord('A')+ 1)
print(ord('C')- ord('A')+ 1)
print(ord('Z')- ord('A')+ 1)
```

Now let's discuss the actual implementation:

```
def spreadsheet_encode_column(col_str):
    num = 0
    count = len(col_str)-1
    for s in col_str:
        num += 26**count * (ord(s) - ord('A') + 1)
        count -= 1
    return num

print(spreadsheet_encode_column("ZZ"))

spreadsheet_encode_column(col_str)
```

Explanation

num is set to 0 initially on line 2 and returned from the function on line 7.

count is initialized to len(col_str) - 1 where col_str is the string

representing the columnID that we have to convert into a number. count will
help us in determining the power of the base as we convert the columnID into
a corresponding integer. On line 4, s will represent each character of

col_str in each iteration of the for loop. Let's jump to line 5 where 26 (base)
is raised to the power of count where count is len(col_str) - 1 as indicated
in the formula. The result from the power operation (**) is multiplied with
the equivalent of s in base 26 system which we find using the ord function.

The final answer as a result of these calculations is added to the value of the existing num on line 5. On line 6, count is decremented by 1 as the power of the base decreases by 1 for the digits from left to the right. After the calculations have been done for every character in s, the for loop terminates, and the value of num is returned from the function.

That's all we have for this lesson. Hope you are having a great learning experience. Let's move on to discussing another problem in the next lesson.