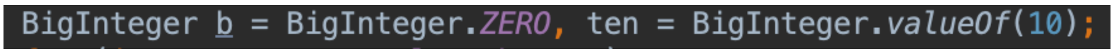
I pick the third practice question in this week's practical. This problem name is MatchNumbersEasy. The goal of this problem is to create the largest possible number using the matches that I have.

For this problem, I can use the dynamic programming approach or the greedy approach to solve. After reading the problem, it is easier to solve this problem by using the dynamic programming approach for me.

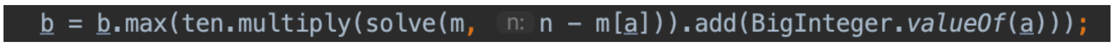
By using the dynamic programming function to iterate through all the numbers that can be created with up to n matches, tries to arrange the next number, deletes the used match, and calls itself recursively to determine the maximum number that can be created with the remaining matches. Then, to choose the best solution.

In the beginning, the method is String maxNumber. I use the BigInteger class. By using this method to get the result of b=0, ten=10.



```
BigInteger b = BigInteger.ZERO, ten = BigInteger.valueOf(10);
```

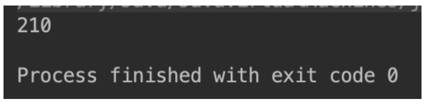
Then, I use the for loop, the statement is a=0; a < the length of m; a++. Next, I use the if statement and the basic function of BigInteger of max(), multiply() and add().
Like this:



```
  b = b.max(ten.multiply(solve(m,  n: n - m[a])).add(BigInteger.valueOf(a)));
```

Meanwhile, it is effective to create the least significant number to the most significant number for avoiding the trouble of the zero number.

I use the first example to test. The Return is "210".



```
210

Process finished with exit code 0
```

And the web submission result is:
Class: MatchNumbersEasy
total_marks: 76
out_of: 76

For this problem, I will try to use other methods to solve it in my spare time. And through this problem, I can use the BigInteger skillfully.