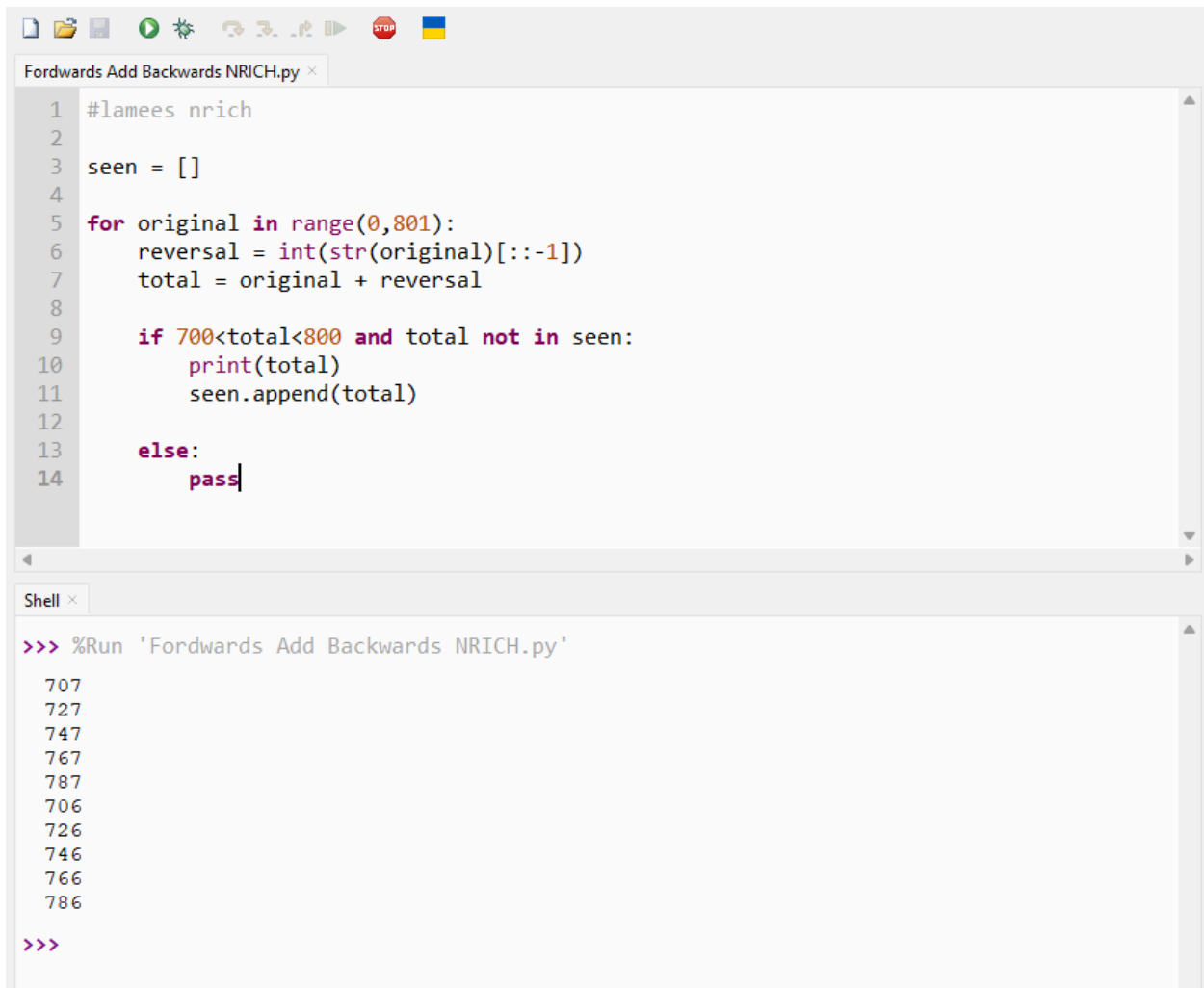


NRICH: Forwards Add Backwards - Lamees E

Here, I used python programming to find every possible solution to this problem.



```
Fordwards Add Backwards NRICH.py x
1 #lamees nrich
2
3 seen = []
4
5 for original in range(0,801):
6     reversal = int(str(original)[::-1])
7     total = original + reversal
8
9     if 700<total<800 and total not in seen:
10         print(total)
11         seen.append(total)
12
13     else:
14         pass

Shell x
>>> %Run 'Fordwards Add Backwards NRICH.py'
707
727
747
767
787
706
726
746
766
786
>>>
```

First, I set an empty list called 'seen' so that, in future, I could increase convenience as the computer would filter out duplicate sums (as addition is commutative, and that different numbers can still give the same sum) and give us a clean list. After that, I set a 'for loop', that would reiterate a process for a given number of times, until it doesn't satisfy a given range. The question asked for all sums between 700 to 800, so I decided to check the original plus reverse of all integers from 0 to 800 because python would filter out values that are extreme, based on a condition that I typed later on in the code. I knew that this range would produce some values that are too big and too small (for example, $0 + 0 = 0$ and $800 + 008 = 808$) but they still included all the sums that would work, and due to the upcoming limiting factor, that's all I needed to worry about.

You may have noticed that I wrote `range(0,801)`; this was on purpose, because for every range $a \leq x \leq b$, in python, that is equivalent to `range(a,b+1)`. Therefore, in order to check all numbers from 0 to 800, I wrote `range(0,801)`.

I used string slicing to find the reverse of the 'original' number; the `[::-1]` forced python to transcribe the number in question backwards, which we later added on to the original number to find the total of the original and reverse.

As aforementioned, I used an inequality to ensure that the sums I got were the ones I wanted: the sums greater than 700 and less than 800. That means that every original number that I started out with and its reverse must have had a total between 700 and 800 to be shown on that list, and the individual number must not have been a duplicate of another sum (as specified in the condition 'total not in seen'). Once it clears the conditions of the if statement, it moves on to output the sum in our shell (the output of the programme) and save it to memory so the sum isn't repeated again; if it didn't clear the if statement's conditions, it moves to the else, where I just wrote 'pass' – nothing is done.

The programme generates a list of all the positive integers whose reversal and their original's total is in between 700 and 800.

Thank you very much and I hope my explanation clarified my piece of code.