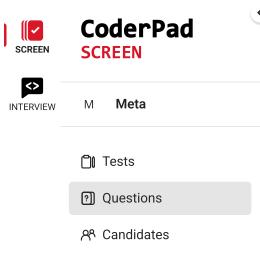
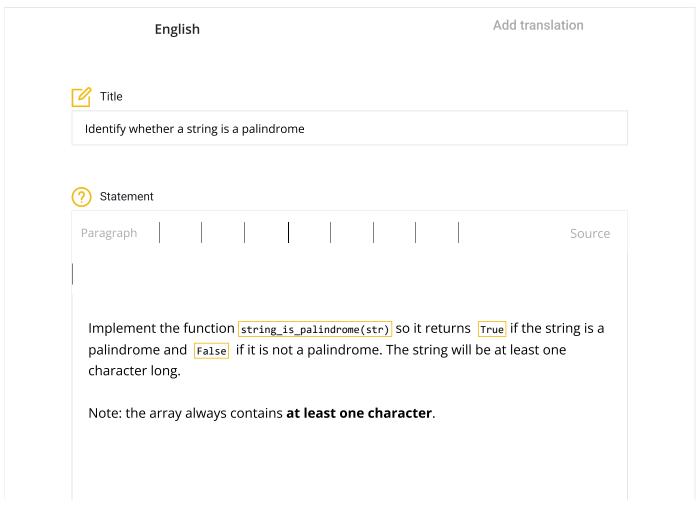
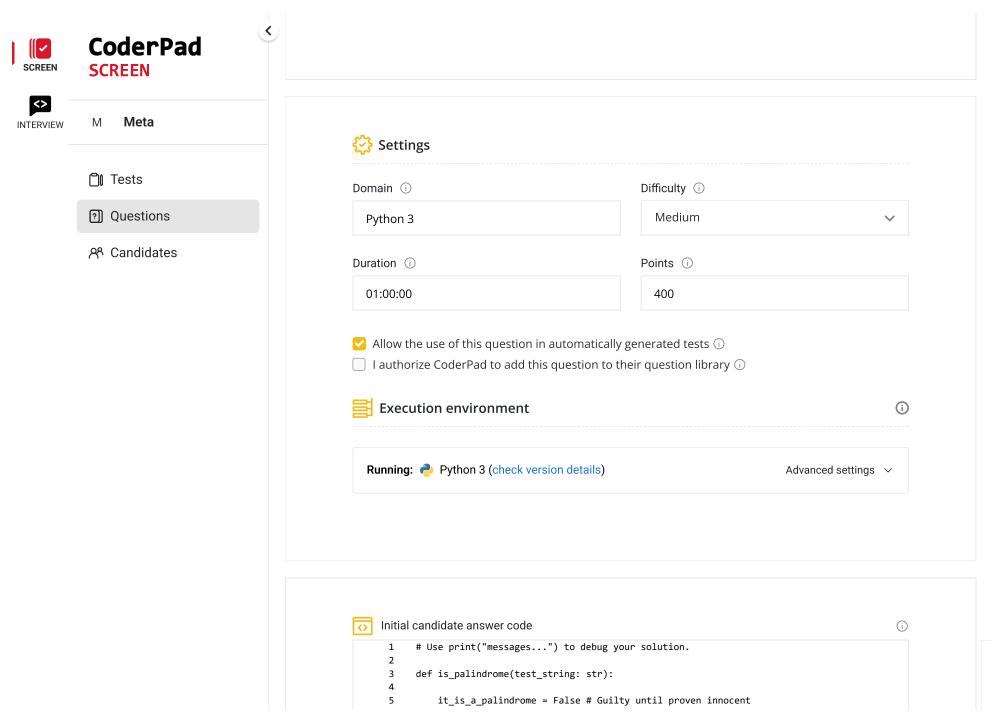
Upgrade

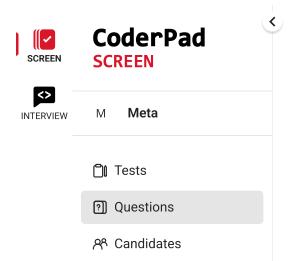






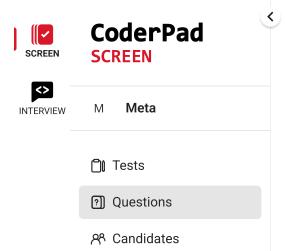
∜ Upgrade



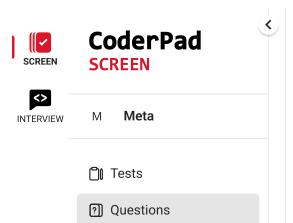


```
10
```

```
Initial candidate test code
                                                                                              (i)
 1
      # Add test code for the candidate.
      # Use the print(...) function to output data.
      # Only the lines of code between DISPLAY_BEGIN and DISPLAY_END
      # will be shown to the candidate.
 5
 6
      # To use the candidate's code, the Answer module must be imported
 7
      import Answer
 8
 9
      # ##DISPLAY_BEGIN##
10
      print(Answer.find_largest("madam"))
                                                    # True
      print(Answer.find_largest("monsieur"))
                                                    # False
11
      print(Answer.find_largest("kayak"))
                                                    # True
12
13
      print(Answer.find_largest("true"))
                                                    # False
      print(Answer.find_largest("false"))
                                                    # False
      print(Answer.find_largest("two2owt"))
15
                                                    # True
16
      print(Answer.find_largest("2357"))
                                                    # False
                                                   # True
17
      print(Answer.find_largest("638836"))
      print(Answer.find_largest("626"))
18
                                                    # True
      print(Answer.find_largest("a"))
                                                    # True
19
20
      print(Answer.find_largest("abcdefghijklm"))
                                                   # False
21
      print(Answer.find_largest("fwgxkhkxgwf"))
                                                    # True
22
      print(Answer.find_largest("iiii"))
                                                    # True
23
      # ##DISPLAY END##
24
```

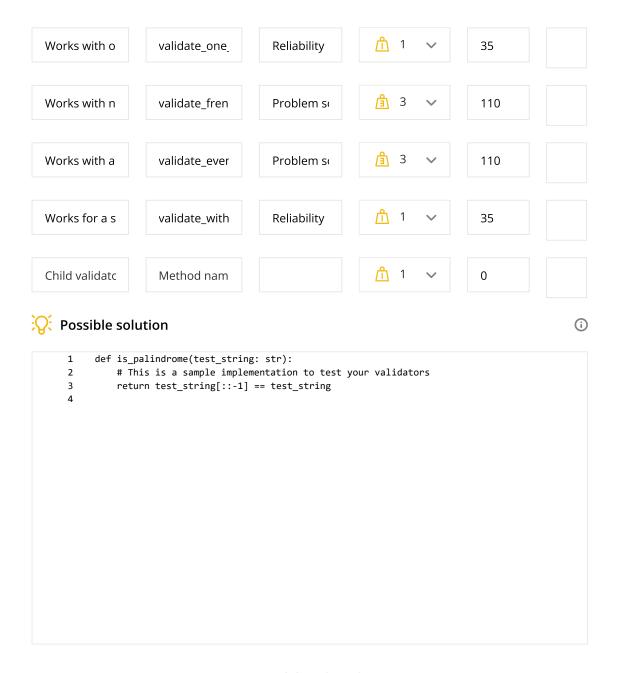


```
Code validator
                                                                                              (j)
      # The goal of the "Code validator" is to validate a programming task.
 1
      # A validator function will mark a candidate submission as "passed" if:
          1) it does not throw an Exception (use assert(...) to throw Exceptions)
          2) executes quickly enough
      # otherwise the submission is marked as "failed".
 5
 6
 7
      # Each function declared here becomes a validator by filling its name in the
 8
      # tree of validators (see at the bottom of this page).
 9
      # If the parent validator fails, child validators will not be executed.
      # Each validator function is started in its own process and should call the
10
11
      # candidate's code using different test cases.
12
13
      import Answer
14
15
      answer1 = True
16
      _string1 = "madam"
17
18
      answer2 = False
19
      _string2 = "madame"
20
21
      answer3 = True
22
      _string3 = "273372"
23
24
25
      def validate():
26
          assert(Answer.is_palindrome(_string1) == _answer1)
27
28
29
      def validate_one_letter_string():
30
          assert(Answer.is_palindrome("a") == True)
31
32
33
      def validate french():
34
          assert(Answer.is_palindrome(_string2) == _answer2)
35
36
37
      def validate even number of chars():
38
          assert(Answer.is_palindrome(_string3) == _answer3)
39
40
      def validate with non ascii():
41
          assert(Answer.is_palindrome("ABç∂EF") == False)
```

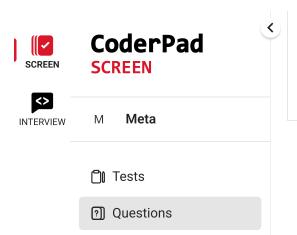


유 Candidates

Your trial ends in 14 days.



Validate the solution



∠ Candidates

