


































# Practice Arena

Practice problems aimed to improve your coding skills.

-  PRACTICE-02\_SCAN-PRINT
-  PRACTICE-03\_TYPES
-  LAB-PRAC-02\_SCAN-PRINT
-  LAB-PRAC-01
-  PRACTICE-04\_COND
-  BONUS-PRAC-02
-  LAB-PRAC-03\_TYPES
-  PRACTICE-05\_COND-LOOPS
-  LAB-PRAC-04\_COND
-  LAB-PRAC-05\_CONDLLOOPS
-  PRACTICE-07\_LOOPS-ARR
-  LAB-PRAC-06\_LOOPS
-  LAB-PRAC-07\_LOOPS-ARR
-  LABEXAM-PRAC-01\_MIDSEM
  -  The D List
  -  The D Factor
  -  All Charged Up
  -  The S Factor
  -  The S List
  -  Smith Numbers
-  PRACTICE-09\_PTR-MAT
-  LAB-PRAC-08\_ARR-STR
-  PRACTICE-10\_MAT-FUN
-  LAB-PRAC-09\_PTR-MAT
-  LAB-PRAC-10\_MAT-FUN
-  PRACTICE-11\_FUN-PTR
-  LAB-PRAC-11\_FUN-PTR
-  LAB-PRAC-12\_FUN-STRUC
-  LABEXAM-PRAC-02\_ENDSEM
-  LAB-PRAC-13\_STRUC-NUM
-  LAB-PRAC-14\_SORT-MISC

# The S Factor

## LABEXAM-PRAC-01\_MIDSEM

### The S Factor [25 marks]

---

#### Problem Statement

You will be given a **long integer**  $n$  as input. Using this integer, you have to print the following on **four different lines**

1. The smallest divisor of  $n$  that is strictly greater than 1 (notice that this divisor has to be prime)
2. The number of times the digit 3 appears when we write down  $n$
3. The number of times the first digit of  $n$  (when seen from the left) appears when we write down  $n$
4. The power of the smallest divisor of  $n$  in its prime factorization

#### Problem-specific Words of Caution:

1. We promise that when we give you the long integer, the first digit of the integer will never be zero
  2. We promise that the number we give you will itself never be 0, it will be strictly positive
  3. Your outputs may not fit inside int variables - take precautions.
- 

#### EXAMPLE:

INPUT

123456

OUTPUT:

2

1

1

6

**Explanation:** The smallest non unity divisor of 123456 is 2. The digit 3 appears once in 123456. The first digit of 123456 is 1 and that also appears only once in 123456. The number factorizes as  $64 * 1929$  and hence, the power of 2, the smallest divisor, is 6 since  $64 = 2^6$ .

---

#### General Words of Caution

1. **Do not forget to submit your code.** You can submit multiple times. Your last submission will get graded.

## 2. Marks will be allotted for the following

1. Proper and meaningful variable names
  2. Nice looking and consistent indentation
  3. At least a couple of comments explaining to the human grader what are you doing, especially when the calculations are not obvious
  4. Comments, good indentation and meaningful variable names are very important for the human grader to understand what are you doing and why. If they cannot understand your code, do not expect them to give you (partial) marks either.
- 
3. Solutions that indulge in hard-coding **will get a straight zero** even if they are passing some test cases. Hard-coding is a form of cheating strategy where someone write code of the form `"if(input == A ) printf( B )"` without doing any calculations on A to obtain B. The values of A and B are either read from the evaluation/submission window or else guessed.
  4. Questions will be graded by the **autograder as well as a human grader**
  5. Be careful about extra/missing lines and extra/missing spaces if you do not want to lose autograder marks
  6. Proportion of marks allotted to autograder (in particular, weightage to visible and hidden test cases) and human grader will be revealed when marks and grading rubrics are released
  7. The total marks of this exam are 150.
  8. You are allowed to use the libraries `math.h` and `stdlib.h` **but not any other library**. Use of unpermitted libraries will carry a penalty. You may use programming tools such as arrays, functions, recursion, pointers, in case you are familiar with the use of these. However, you will be given no special credit for using these advanced programming techniques, nor will you receive any help should you face difficulties in using them, for example, TLE or segmentation fault errors. Use these advanced techniques at your own risk.

---

### Grading Scheme:

Total marks: **[25 Points]**

There will be partial grading in this question. There are four lines in your output. Printing each line correctly, in the correct order, carries some weightage. The first two lines are worth 12.5% each. The third line is worth 25% and the fourth line is worth 50% weightage. There are 2 visible and 4 hidden test cases.

Please remember, however, that when you press Submit/Evaluate, you will get a green bar only if all parts of your answer are correct. Thus, if your answer is only partly correct, Prutor will say that you have not passed that test case completely, but when we do autograding afterwards, you will get partial marks.

 **Start Solving!** (/editor/practice/6155)