Question 1

We all know the prime number series (p_i): 2,3,5,7,11,13,17,19… where p_i is the i^{th} prime number. I introduce a new k-series (k_i): 2,3,5,5,7,7,11,13,13,17,19,19,... where k_i is the i^{th} k-series number.
This series is the same as prime numbers; the only catch is that if p_i-p_{i-1}=2 and k_j=$p_{i,}$ then k_j=k_{j+1}
Your input (present in “q1-input.txt” in the same directory) is as follows =>
The first line has a natural number **q,** followed by **q** queries, each a natural no.
For each query (say if it is n), you need to print the n^{th} k-series number or k_n.
Note:-

1) p_1=k_1=2, and there is no term like p_0 or k_0
2) Input is always present in “q1-input.txt” and no other file.
3) Try to be as efficient as possible (try not to generate the entire {k_i} series from scratch for every query. You may use the series you generated for some previous queries for the next queries).
4) Write your script in playing_with_primes.py
5) A test case has been provided in “q1-test_input.txt” and the expected output in “q1-expected_output.txt”.

How to run the program:-
python3 playing_with_primes.py

Question 2

You are given some sets. You need to find and print the size of the union of those sets. The input will be given in “q2-input.txt”. Each line corresponds to a separate set. All elements are integers separated by commas.
Example => If the input (in “q2-input.txt”) is:-
3,1,2,4
5,2,3,7
The output should be => 6
Note:-

1) Any no. of sets of any non-zero size will be given. If no sets, print 0.
2) Write your script in playing_with_sets.py

How to run the program:-
python3 playing_with_sets.py

Question 3

A Mathematics exam was held for class VI students. There are various “sections” in which the entire VI standard is divided into {VI#1, VI#2, VI#3, …}. The marks of each student were filled in with their roll numbers (section-wise), but due to some problem with the machine, the entries got jumbled; hence, no more section-wise.

Your task is to “un-jumble” this mark sheet and create a new mark sheet where

firstly VI#1 marks come, then VI#2, then VI#3 and so on. For the same section, students with higher marks should appear upper in the list.
Students are denoted by their roll numbers <year>_<class>_<section>_<4-char unique ID>, example: 2022_5_2_MK21.
Marks will be non-negative numbers next to roll no. separated by a space.
Input will be in “buggy_marksheet.txt”, and output will be in “fixed_marksheet.txt”
Example:- If input is =>

```
2022_6_4_KG19 95
2019_6_3_SR17 90
2021_6_3_GS18 91
23_6_4_MK21 88
2020_6_2_LL18 90
```

The output will be =>

```
2020_6_2_LL18 90
2021_6_3_GS18 91
2019_6_3_SR17 90
2022_6_4_KG19 95
23_6_4_MK21 88
```

Note:-

1) ANY number of sections can exist. ANY number of entries can be there.
2) Same section students with same marks can be arranged in any order.
3) The year in a roll number may not always be 4 digits.
4) Write your script in playing_with_marks.py

Hint: Use the sorted() function of Python.

How to run the program:-

```
python3 playing_with_marks.py
```

Question 4

Again, a simple program. An untouched string s (say “KAVYA”) has an index array of {0,1,2,3,4}. I jumble up this array, making a permutation P: {3,1,0,2,4}. The corresponding string of permutation P is “YAKVA”.
Your task is to find the corresponding string for a permutation of its index array. Using standard input (i.e. input() function), you will be given an input string and a permutation (in this, indices are separated by space).
If the given permutation array is invalid, print “INVALID INPUT”.
Example:- If your input is =>

```
SAMPLE
5 4 3 2 1 0
```

The output should be =>

```
ELPMAS
```

Clarification on taking input:-

The input file will have two lines (“q4-input.txt”). The first line will contain the string, and the second line will have the permutation array. To extract the indices from the second line, use the “split()” function.

Note:-

1) A permutation is invalid if any index is missing or there are extra indices or out-of-range values.
2) The input string will have capital letters only. There's no need to worry about it.
3) Write your script in playing_with_strings.py

How to run the program:-

```
python3 playing_with_strings.py < q4-input.txt
```