**Анализ**

**Подзадача 1:**

 Подзадачата служи за по-добра обратна връзка от системата.

**Подзадача 2:**

За всеки последователен подниз на защитния ключ, правим проверка със сложност O(N) дали е префикс на атакуващия ключ, като пазим най-големият размер на префикс, който сме открили, за да сравним с P и да получим победителя.

 *Крайна сложност: O(N3)
 Имплементация:* ***captaink\_5p.cpp***

**Подзадача 3:**

Използваме същото решение като за предната подзадача, но този път добавяме още един цикъл, чрез който изпълняваме същите стъпки, но за всеки защитен ключ, получен след премахването на определен символ.

 *Крайна сложност: O(N4)
 Имплементация:* ***captaink\_17p.cpp***

**Подзадача 4:**

Тази подзадача е за награда на бавни, но оригинални (или много оптимизирани) решения, за които авторът не се е сетил :)

**Подзадача 5:**

Тъй като тази група не изисква търсенето на индекс за премахване, можем да се фокусираме изцяло върху намирането на най-дълъг подниз-префикс. Това лесно може да стане чрез Z функция, като по този начин смъкваме сложността от O(N3) на O(N+M), което е огромно подобрение.

 *Крайна сложност: O(N+M)
 Имплементация:* ***captaink\_50p.cpp***

**Подзадача 6:**

Сега остана само да намерим бърз начин за откриване на правилния индекс. Засега в алгоритъма за Z функция имаме цикъл while, който спира при достигането на символ, различен от съответния му символ в префикса. Можем да използваме това, като всеки път приемаме че премахваме този елемент и продължаваме с цикъл максимално напред до достигането на ново несъответствие, след което ако полученият размер е по-голям от P, значи сме намерили решение и няма нужда повече да премахваме символи в следващите итерации на цикъла for.

 *Имплементация:* ***author.cpp***