Câu 1: Loại bỏ

Subtask 1:

- Với cách làm chung, chúng ta sẽ tìm UCLN của tất cả các số, sau đó chia tất cả các số cho UCLN.
- Sau đó, việc chúng ta cần làm là tìm một tập con của dãy sao cho UC của chúng lớn hơn 1.
- Với subtask này, ta chỉ cần for từng ước chung để đếm từng tập con.
- Đpt: $O(n^2)$

Subtask 2:

- Với n lớn hơn, ta không thể for từng ước chung từ 1 đến 10^6, nên ta phải thay đổi thuật toán.
- Ta thấy rằng ước chung trong một tập con thì đều là ước của từng số trong tập con đó. Do vậy, chúng ta sẽ liệt kê ra tất cả các ước của các phần tử, sau đó chọn ra ước có nhiều phần tử làm ước nhất, chúng ta sẽ ra được kết quả.
- Việc liệt kê ước này sẽ cần phải dùng sàng nguyên tố để thực hiện. Sau khi sàng nguyên tố, ta sẽ chọn ra được những số nguyên tố trong bảng. Sau đó, với từng số, ta for từng số nguyên tố để kiểm tra.

Subtask 3:

- Với subtask này, ta không thể liệt kê từng số nguyên tố rồi duyệt từng số một được. Ta cần phải tối ưu bằng cách gán số nguyên tố gần nhất với một số trong khi sàng với số nguyên tố luôn. Từ đó, ta sẽ không tốn nhiều thời gian để có thể duyệt hết các ước của một số.

```
for (int i = 2; i * i < MAXT; i++)
   if (prime[i] == 0)
      for (int j = i * i; j < MAXT; j += i)
         prime[j] = i;
```

Câu 2: Phần thưởng

Subtask 1:

- Ta chỉ cần dùng 3 vòng for để duyệt trâu qua tất cả các xâu
- Đpt: $O(n \times m \times 200)$

Subtask 2 + 3:

- Ta cần dùng hash và một map để đánh dấu. Với n xâu ban đầu, ta sẽ hash chúng, và đưa vào mảng đánh dấu như sau:
    - Với mỗi vị trí i, ta sẽ lưu lại mã hash từ đoạn $[0, i-1]$ và đoạn $[i+1, k]$ với k là độ dài của xâu hiện tại.
    - Giả sử mã hash ở đoạn $[0, i-1]$ và đoạn $[i+1, k]$ tương ứng là x và y, thì ta sẽ hợp chúng lại như sau: $z = x \ll 30 \mid y$.

    - Vậy thì còn ký tự ở vị trí i thì để làm gì?? Sau khi tính xong z, ta sẽ map chúng với key là z. Còn value sẽ là một cái bitmask gồm 26 bit. Mỗi bit tương ứng với vị trí của ký tự trên bảng chữ cái. Vậy nếu ký tự i nằm ở vị trí nào thì bit đó sẽ được bật lên.
- Sang đến các xâu cần truy vấn, ta lọc độ dài của chúng, rồi ta sẽ duyệt từng ký tự trong xâu đó làm ký tự khác nhau, và ta sẽ dùng map để truy vấn xem mã hash của các ký tự còn lại có giống nhau không.
- Đpt: $O(n \times logn)$.

Câu 3: Trò chơi

Subtask 1:

- Ta sẽ duyệt quay lui để xét tất cả các trường hợp, sau đó ta tìm giá trị lớn nhất.
- Đpt: $O(k^n \times n)$

Subtask 2:

- Ta thấy được rằng, thay vì nhân rải rác với mỗi phép biến đổi, ta nên dồn tất cả các biến đổi vào 1 số. Tại sao vậy??? Bởi vì đây là phép OR, chỉ có thêm bit 1 chứ không có mất bit 1, nên ta sẽ cố làm các số có vị trí bit 1 lớn nhất có thể. Do vậy, ta chỉ nên dùng tất cả các phép biển đổi để nhân vào 1 số làm cho số đó có vị trí bit 1 xa hơn.
- Ta sẽ thử nhân tất cả các phép biến đổi cho từng số 1, ta vẫn tính phần OR còn lại bình thường bằng vòng lặp
- Đpt: $O(n^2)$

Subtask 3:

- Ta có thể rút gọn vòng for tính phần OR bằng 2 mảng prefix và suffix, còn việc còn lại sẽ giống với subtask 2
- Đpt: $O(n)$.