УКАЗАНИЯ ПО РЕШЕНИЮ ЗАДАЧ

Кто хочет стать миллионером

Можно в цикле каждый раз удваивать предыдущее число. Если на каком-то шаге получается не круглый приз, то в цикл добавляется условие: если шаг такой-то, то вместо удвоения замени предыдущее число на данное число. Оказывается, что для первых двадцати пяти призов понадобится сделать три изменения.

ROT13

Техническая задача на умение работать с символами и строками. Один из способов решения — создать константные строки, которые соответствуют алфавиту и преобразованию. Далее для каждой буквы необходимо найти её в первой строке. Это порядковый номер в алфавите и по этому номеру во второй строке находится результат преобразования.

```
a1 = "abcdefghijklmnopqrstuvwxyz"
a2 = "nopqrstuvwxyzabcdefghijklm"

s = input()
for c in s:
  if c in a1:
    i = a1.find(c)
    print(a2[i], end="")
  else:
    print(c, end="")
```

Последовательность Морса — Туэ

На каждом шаге последовательность удваивается и вторая половина это копия первой с некоторой модификацией. Это значит, что если ищется символ на позиции N, в строке длины P, то он находится либо в первой половине и тогда можно отбросить вторую половину и искать символ на позиции N в строке длины $P/2$, либо он находится во второй половине. В последнем случае, искомый символ это символ противоположный соответствующему символу в первой половине. То есть если найти элемент на позиции N-$P/2$ в строке $P/2$, то ответ это противоположный ему.

Таким образом, на каждом шаге в два раза уменьшается длина рассматриваемой строки и очень быстро она станет равна одному, а для этой строки ответ это ноль. И останется учесть изменения элемента на противоположный.

Пример: найти элемент на позиции 27. Чтобы получить этот символ, необходимо выполнить пять удвоений и получить последовательность длины 32. Значит, в строке длины 32 требуется найти 27-й символ. Он находится во второй половине и является противоположностью 11-го (27-16) элемента в последовательности длины 16. Который в свою очередь находится во второй половине и является противоположностью третьего (11-8) элемента в последовательности длины 8. Он находится в первой половине и поэтому вторая половина может быть отброшена, чтобы найти третий элемент в последовательности длины четыре. Опять вторая половина и искомый элемент это противоположность первого элемента в последовательности длины два. Первый элемент это ноль. Теперь в процессе произошло три раза взятие противоположного элемента и значит 27-й элемент это один.

```
n = int(input())
p = 1
while p < n:
  p  = 2 * p
i = 0
while p > 1:
  q = p // 2
  if n > q:
    i = 1 - i
    n -= q
  p = q
print(i)
```

Сортировка перестановки

Правильное решение — сделать так, чтобы после каждого обмена какое-то число встало на своё место. Например, найти единицу и поменять её с тем числом, которое стоит на первом месте. Потом повторить то же самое с двойкой, тройкой и всеми оставшимися числами.

Если дословно реализовать этот алгоритм, то он будет правильный, но медленный. Правильный и быстрый вариант — это, например, взять число, которое стоит на первом месте и поставить его на своё. После этого обмена на первом месте окажется новое число. Если оно не единица, то операцию нужно повторить и так, пока на первом месте не окажется единица. Потом повторить со вторым местом, с третьим местом и т. д. до последнего.

```
n = int(input())
a = [int(x) - 1 for x in input().split()]
k = 0
for i in range(n):
  while a[i] != i:
    j = a[i]
    a[i] = a[j]
    a[j] = j
    k += 1
print(k)
```

Почему эта стратегия работает за минимальное количество обменов? Для доказательства необходимо рассмотреть перестановку как набор циклов. Что такое цикл легко понять на примере перестановки 6 3 2 4 7 5 1. Встанем на первую позицию, там стоит шесть, поэтому пойдём на шестую позицию, там стоит пять, поэтому пойдём на пятую позиция, там стоит семь, поэтому пойдём на седьмую позицию, там стоит один поэтому пойдём на первую позицию. Это цикл: $1 \to 6 \to 5 \to 7 \to 1$. Другие циклы в этой перестановке это $2 \to 3 \to 2$ и $4 \to 4$. За одну операцию обмена можно увеличить количество циклов на один. Для этого необходимо поменять два элемента из одного цикла. Если поменять элементы из разных циклов, то количество циклов уменьшиться. В отсортированной перестановке N циклов и значит минимальное количество обменов это N минус количество циклов в исходной перестановке. Или, как в примере программы выше, минимальное количество обменов можно получить, если производить эти обмены, постоянно меняя элементы из одного цикла.