B. Lucky String

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Petya loves lucky numbers. We all know that lucky numbers are the positive integers whose decimal representations contain only the lucky digits **4** and **7**. For example, numbers **47**, **744**, **4** are lucky and **5**, **17**, **467** are not.

Petya recently learned to determine whether a string of lowercase Latin letters is lucky. For each individual letter all its positions in the string are written out in the increasing order. This results in 26 lists of numbers; some of them can be empty. A string is considered lucky if and only if in each list the absolute difference of any two **adjacent** numbers is a lucky number.

For example, let's consider string "`zbcdzefdzc`". The lists of positions of equal letters are:

- `b`: 2
- `c`: $3, 10$
- `d`: $4, 8$
- `e`: 6
- `f`: 7
- `z`: $1, 5, 9$
- Lists of positions of letters `a`, `g`, `h`, ..., `y` are empty.

This string is lucky as all differences are lucky numbers. For letters `z`: $5 - 1 = 4$, $9 - 5 = 4$, for letters `c`: $10 - 3 = 7$, for letters `d`: $8 - 4 = 4$.

Note that if some letter occurs only once in a string, it doesn't influence the string's luckiness after building the lists of positions of equal letters. The string where all the letters are distinct is considered lucky.

Find the lexicographically minimal lucky string whose length equals n.

Input

The single line contains a positive integer n ($1 \le n \le 10^5$) — the length of the sought string.

Output

Print on the single line the lexicographically minimal lucky string whose length equals n.

Examples

input
5
output
abcda

input
3
output
abc

Note

The lexical comparison of strings is performed by the < operator in modern programming languages. String a is lexicographically less than string b if exists such i ($1 \le i \le n$), that $a_i < b_i$, and for any j ($1 \le j < i$) $a_j = b_j$.