PROBLEM:

Some numbers have a curious property: when rotating the number to the right, the new number is the double of the original. Rotating the number to the right means choosing the last digit and moving it to the beginning of the number, as the leftmost digit. For example, the number 421052631578947368 when rotated to the right gives 842105263157894736, which is the double of the original. These numbers are expressed in the decimal system. In any numeral system such numbers exist, for example, in the binary (base 2) numeral system, numbers 01 and 0101 have this property. Note that leading zeros are required in this case.

Write a program that, for any given base B (2 <= B <= 250), finds the smallest number in that numeral system which has this property.

INPUT:

The input consists of a number, B.

OUTPUT:

For each number B, output one or more numbers separated by a blank space that represent the digits in base B (written as decimal numbers) of the smallest number which has this property.

SAMPLE:

- Input:
  2

  Output:
  0 1

- Input:
  10

  Output:
  0 5 2 6 3 1 5 7 8 9 4 7 3 6 8 4 2 1

- Input:
  35

  Output:
  11 23

HINT:

In sample #1
The initial number (when converted to decimal system):
0 * 2^1 + 1 * 2^0 = 1
After applying the rotation:

1 * 2^1 + 0 * 2^0 = 2.
In sample #3
The initial number (when converted to decimal system):
11 * 35^1 + 23 * 35^0 = 408
After applying the rotation:
23 * 35^1 + 11 * 35^0 = 816.

Note: To handle very large values (as in decimal system) ,use long long int data type. (Format specifier “%lld” in C, supported by newer compliers).