

Preamble

Load necessary packages and the lab data

```
In [154... from matplotlib import pyplot
         import matplotlib
         import numpy
         import scipy
         import pickle
         import pandas
         from IPython.display import Markdown
         with open('Lab3.pkl', 'rb') as f:
             Lab3 Data, Lab3 Problem1 demo, Lab3 Problem2, Lab3 synthetic identities =
         # The following are helper functions to facilitate printing outputs in the lab
         def safe print err(e, locals=locals(), globals=globals()):
             locals = locals or {}
             globals = globals or {}
              __problem = eval('__problem', locals, globals)
             err msq = str(e)
             if type(e) in (NameError, KeyError):
                 var name split = str(e).split("'")
                 if len(var name split) > 1:
                     var name = var name split[1]
                     err_msg = f"'{var_name}' is not defined."
             display(Markdown(f"<div class='alert alert-block alert-danger'><b>Error in
         def print mk(msg):
             display(Markdown(msg))
```

Data Description

The dataset we will be using this lab is the 2015-2016 portion of the National Health and Nutrition Examination Survey. The dataset and its documentation may be found at https://wwwn.cdc.gov/nchs/nhanes/continuousnhanes/default.aspx?BeginYear=2015.

We will be using a small subset of the dataset, compiled in the variable Lab3_Data, which is a data frame (for documentation click here). Here is a general description of the columns in the data frame (we use the terms data

frame and **table** interchangeably in this lab):

- Subject_ID: this is a sequence number, assigned uniquely to each respondent.
- Gender: this is the gender of the subject. 1=Male, 2=Female.
- Age : this is the age, in years, of the subject.
- Marital_Status: this representes the marital status of the subject. |
 Marital_Status: | Value Description | | :- | :- | | \$1\$ | Married | | \$2\$ |
 Widowed | | \$3\$ | Divorced | | \$4\$ | Separated | | \$5\$ | Never married | |
 \$6\$ | Living with partner | | \$77\$ | Refused | | \$99\$ | Don't know | | NaN |
 | Missing|
- Country_Birth: the country where the subject was born.
 | Country_Birth | Value Description | | :- | :- | | \$1\$ | Born in 50 US states or Washington, DC | | \$2\$ | Others | | \$77\$ | Refused | | \$99\$ | Don't know | | NaN | Missing|
- Race: the race of the subject.
 | Race: | Value Description: | | :- | :- | | \$1\$ | Mexican American: | | \$2\$ |
 Other Hispanic: | | \$3\$ | Non-Hispanic White: | | \$4\$ | Non-Hispanic Black
 | | \$6\$ | Non-Hispanic Asian: | | \$7\$ | Other Race Including Multi-Racial
 | | NaN | Missing: |
- LDL_Cholesterol: The subject's low density lipoprotein cholesterol in units of milligrams per deciliter (mg/dL).
- HDL_Cholesterol: The subject's high density lipoprotein cholesterol in units of milligrams per deciliter (mg/dL).
- Total_Cholesterol: The subject's total cholesterol level in units of milligrams per deciliter (mg/dL).
- Sys_Blood_Pressure: The subject's systolic blood pressure (average of 3 readings) in units of millimeters of mercury (mm Hg).

Before We Start (Please Read)

Function Pointers

numpy.median

In [155...

In [158...

Out[158... 4.0

Out[159... 4.0

A specification of Python that we will be using in this lab is passing a function as a parameter to another function. In order to achieve this, we utilize function pointetrs. A function name is just a pointer to the function itself. For example:

```
Out[155... <function median at 0x000001CC3911B1F0>

Which doesn't evaluate the function numpy.median but rather represents a reference to the function numpy.median as an object. This can be particularly useful if we would like to pass the function numpy.median to another function, or store it in a variable. For instance, a function pointer can be stored in a variable:

In [156... my_fcn_ptr_1 = numpy.median

The variable my_fcn_ptr_1 holds a reference to the function numpy.median:

In [157... my_fcn_ptr_1

Out[157... <function median at 0x000001CC3911B1F0>

From this moment, the variable my_fcn_ptr_1 on can be used as a function on its own right:
```

Lambda Functions

You can define a function and store its pointer in a variable without giving a name to that function. For example, let's say you want to write a function that takes two arguments, x and y, and returns the value \$20 \cdot $x^2 + \frac{y}{3}$ \$. One

 $my_fcn_ptr_1([1, 2, 3, 4, 5, 6, 7])$ # should return the median of the vector:

numpy.median([1, 2, 3, 4, 5, 6, 7]) # this is equivalent to the last call

way you can achieve this is by writing the following function:

```
In [160... def my_fancy_function(x, y):
    return 20*x**2+y/3
```

Alternatively, you can define the function without giving it a name as follows:

```
In [161... lambda x, y : 20*x**2+y/3
```

Out[161... <function __main__.<lambda>(x, y)>

But from this moment on, we don't have a good way to call this nameless function. To solve this, we can store this function pointer in a variable, which will serve as our way to access this "nameless" function. "Nameless" functions are called Lambda functions (or \$\lambda\$-Functions)

```
In [162... my_fcn_ptr_2 = lambda x, y : 20*x**2+y/3
```

Now, my_fcn_ptr_2 is a variable that holds a reference to the Lambda function that takes two inputs, x and y, and returns the value of the expression \$20 \cdot $x^2 + \frac{y}{3}$:

```
In [163... my_fcn_ptr_2(2,3) # should return 81
Out[163... 81.0
```

Passing a Function Pointer to Another Function

Now that we know that a function pointer can be stored in a variable, we can also pass it to any function as any other variable. This, in essence amounts to passing a function to another function. *Inception*, huh?

The full effect of this behavior and functionality is beyond the scope of our course, but for us this will be useful when we study **Mondrian** in Problem 1.

This is because Mondrian can work regardless of the way we select the next dimension to perform the cut on (we can select the next dimension to cut randomly, or according to some order, etc). So, the implementation of Mondrian should be the same regardless of the manner we want to chose the next column for the cut. Using function pointers, we can implement one version of Mondrian, which will expect the user to specify, through their own function, the manner Mondrian should select the next column for the cut.

But to understand the mechanism with a simple example, consider the following function:

```
In [164... def shift_data_1(data):
    """A function that shifts the input data left by the mean of the data.

Parameters
    ......
data : array
    The data to be shifted

Returns
    ......
data_out : array
    The shifted data
    """

representative_value = numpy.mean(data) # calculate a representative value return data - representative_value # Shift the data by the representative
```

The function shift_data_1 simply shifts the input data left by the **mean** of the data. For example, if we run the function shift_data_1 on the list [0, 1, 5] (mean=2), the output would be [-2, -1, 3]:

```
In [165... shift_data_1([0, 1, 5])
Out[165... array([-2., -1., 3.])
```

We can implement the following similar function:

```
In [166...

def shift_data_2(data):
    """A function that shifts the input data left by the median of the data.

Parameters
------
data : array
    The data to be shifted

Returns
------
data_out : array
    The shifted data
"""

representative_value = numpy.median(data) # calculate a representative val
return data - representative_value # Shift the data by the representative
```

The only difference between shift_data_1 and shift_data_2 is simply the choice of the "representative value." This time, we shift the input data left by the **median** of the data (instead of the **mean** as in the first function). This time,

running the function shift_data_2 on the list [0, 1, 5] (median=1) would return [-1, 0, 4]:

```
In [167... shift_data_2([0, 1, 5])
Out[167... array([-1., 0., 4.])
```

What if we would like to allow the user of the function to decide how to shift the data (maybe they would like to use some other measure other than **mean** or **median** altogether)? In this case, we can implement this function:

```
In [168... def shift_data(data, rep_val_fcn):
    """A function that shifts the input data left by the a reference amount the

Parameters
------
data : array
    The data to be shifted
cut_choice_fcn : types.FunctionType (lambda data: number)
    A pointer to the function calculating the reference amount to shift the

Returns
-----
data_out : array
    The shifted data
"""

representative_value = rep_val_fcn(data) # calculate a representative value return data - representative_value # Shift the data by the representative
```

This time, the function shift_data will "ask" the user for their choice of how to calculate the "representative value" from the input data through the second argument to the function, rep_val_fcn. That is, the function shift_data expects the second argument to be a reference to a function that takes one argument (data) and outputs a single number (representative_value).

Using this, we can replicate the first function by running

and if we wish to use the sum of the values as the representative value, then

Problem 1

In this problem, we will study Mondrian. For reference, the Mondrian algorithm, as presented in the asynchronous part of the course, is presented below:

```
Function Mondrian(part, k) is
Input: Partition part; parameter k
Output: Minimal multi-dimentional k-anonymized paritioning of part.
To Start: To get a minimal multi-dimentional k-anonymized paritioning of table
T w.r.t. Q, call Mondrian(T[Q], k).

1 if no allowable k-anonymous multi-dimensional cut exists for part then
2 | sum_stat $\leftarrow$ Summary_Statistic(part) # To replace all values in part
3 | let $\phi$:part$\rightarrow${sum_stat} such that
```

```
4 | | $\forall v \in$ part$: \phi(v) \triangleq$ sum stat
 5 | return $\phi$
 6 else
   | dim $\leftarrow$ Choose Dimension(part, k) # Select dimension w/
allowable cut
        fs $\leftarrow$ Frequency Set(part, dim) # Calculate frequency set w.r.t.
dim
 9 |
        v split $\leftarrow$ Find Median(fs) # Find the split value (where to
perform the cut)
10 | # The two sub-partitions
11
    | Ihs $\leftarrow \left\{t \in part | t.dim \leq v\ split \right\}$
12
   | rhs $\leftarrow \left\{t \in part | t.dim > v\ split \right\}$
        return Combine(Mondrian(Ihs, k), Mondrian(rhs, k))
13
```

The algorithm above (as presented in the async) is the default version that selects the next cut dimension at random, and always cuts on the median value of the selected dimension. We would like to expand this in our implementation and test various dimension and cut value choice strategies (thereafter: the extended Mondrian algorithm).

You are provided the implementation of the extended Mondrian algorithm. You don't need to edit the implementation of Mondrian. However, you are strongly encouraged to understand the code and how it corresponds to the algorithm presented above.

```
In [175...
        def Mondrian allowable dims(table partition, quasi identifiers, k, cut choice
             """A helper function that determines the list of dimensions with allowable
             Parameters
             table partition : pandas.DataFrame
                 The partition on which to determine the dimensions with allowable cuts
             quasi identifiers : list
                 A list containing the set of quasi-identifiers (or dimensions)
             k : int
                 The desired k
             cut choice fcn : types.FunctionType (lambda data, k: number)
                 A function pointer to the cut value selection strategy
             Returns
             allowable dims : list
                 A list containing the set of dimensions with allowable cuts.
             # We don't know which dimensions will have allowable cuts yet, so we initi
```

```
allowable dims = []
    for dim name in quasi identifiers:
        # For this dimension, the values are
        dim values = table partition.loc[:,dim name]
        # and the cut value is
        dim boundry cut = cut choice fcn(dim values.to list(), k)
        lhs = table partition.loc[table partition.loc[:,dim name] <= dim bound</pre>
        rhs = table partition.loc[table partition.loc[:,dim name] > dim boundr
        if lhs.shape[0] >= k and rhs.shape[0] >= k:
            allowable dims.append(dim name)
    return allowable dims
def Mondrian(table in, quasi identifiers, k, dim choice fcn, cut choice fcn):
    """The Mondrian algorithm implementation.
   Parameters
    _____
    table in : pandas.DataFrame
       The input table to be generalized
   k: int
        The desired k
   dim choice fcn : types.FunctionType (lambda partition, allowable dims: str
        A function pointer to the dimension selection strategy
    cut_choice_fcn : types.FunctionType (lambda data, k: number)
        A function pointer to the cut value selection strategy
   Returns
    _ _ _ _ _ _ _
    table out : pandas.DataFrame
        The generalized k-Anonymous table
    partition boundaries : pandas.DataFrame
        A dataframe describing for each partition:
            - the partition boundaries for each quasi-identifier (minimum, max
            - the partition's final k value
   Raises
    _ _ _ _ _ _
    Exception
       If table in cannot be made k-Anonymous.
   if Lab3 Data.shape[0] < k:</pre>
        # Impossible, we can't achieve k-Anonymity, there aren't enough
        # rows in the table!
        raise Exception('It is impossible to k-Anonymize the input table. Ther
   allowable dims = Mondrian allowable dims(table in, quasi identifiers, k, c
    if len(allowable dims) == 0:
        # In this case, there are no more allowable cuts for this partition.
        # Go through the different attributes and change their values by the oldsymbol{t}
        nr dims = len(quasi identifiers)
        # initialize the partitiens table
        partition boundaries = pandas.DataFrame(index=numpy.arange(1), columns
        # get the number of rows for the output table
```

```
nr rows = table in.shape[0]
    # initialize the output table
    table out = pandas.DataFrame(index=table in.index, columns=table in.cc
    # In the output table, set the values of the quasi identifiers accordi
    for dim name in quasi identifiers:
        # find the boundaries
        curr boundaries = (min(table in.loc[:,dim name]), max(table in.loc
        # set the boundries of this dimension in the output variable
        partition boundaries.loc[:,dim name] = [curr boundaries]
        # and set all values of this column to the boundaries in the outpu
        # display(numpy.tile(partition boundaries.loc[:,dim name],(nr rows
        table out.loc[:,dim name] = [curr boundaries]
        #table out.loc[:,dim name] = numpy.tile(partition boundaries.loc[:
    # Now that we've worked all quasi-identifiers, we need to copy the val
    for dim name in table in.columns.difference(quasi identifiers):
        table out.loc[:,dim name] = table in.loc[:,dim name]
    # # Package partition boundaries in a list (to prepare it to be append
    # partition boundaries = [partition boundaries]
    # finally, the output k for this table is the size of the table
    partition boundaries.loc[:,'k'] = [table out.shape[0]]
else:
    # In this case, there is at least one dimension with an allowable cut.
    # Choose a dimension according to our dimension choice function.
    dim = dim choice fcn(table in, allowable dims)
    # Now we calculate the cur value
    dim boundry values = table in.loc[:,dim]
    dim boundry cut = cut choice fcn(dim boundry values.to list(), k)
    # The left hand side cut
    lhs = table in.loc[table in.loc[:,dim] <= dim boundry cut,:]</pre>
    lhs out, lhs boundaries = Mondrian(lhs, quasi identifiers, k, dim choi
    # The right hand side cut
    rhs = table in.loc[table in.loc[:,dim] > dim boundry cut,:]
    rhs out, rhs boundaries = Mondrian(rhs, quasi identifiers, k, dim choi
    # Combine the paritions
    table out = pandas.concat([lhs out, rhs out])
    # Combine the boundaries definitions
    partition boundaries = pandas.concat([lhs boundaries, rhs boundaries])
# Sorting is unnecessary in practice, but doing it so that the results are
table out = table out.sort values(by=quasi identifiers).reset index(drop=1
partition boundaries = partition boundaries.sort values(by=quasi identifie
return table_out, partition boundaries
```

The function Mondrian accepts the following input arguments:

table in: The input table (as a whole, including quasi-identifiers and

- sensitive attributes).
- quasi_identifiers : The list of quasi-identifiers, represented as a list of the column names of the quasi-identifiers.
- k : the desired level of privacy in terms of k-Anonymity.
- dim_choice_fcn: A function pointer for the choice of the next dimension to perform the cut on (cf. line 7 in the algorithm). More information about function pointers is given in the beginning of the lab. The type of the function dim_choice_fcn is lambda partition, allowable_dims, where partition is the current partition represents as a table and allowable_dims is the set of quasi-identifiers with allowable cuts (represented as a list of column names). Read more on dimension choice functions below.
- cut_choice_fcn: A function pointer for the choice of the cut value (split value) of the data of a given dimension (cf. line 9 in the algorithm). More information about function pointers is given in the beginning of the lab. The type of the function cut_choice_fcn is lambda data, k where data is a list containing the data points of the desired dimension (a list, not a table) and k is the desired level of \$k\$-Anonymity. Read more on cut value choice functions below.

The function Mondrian returns two outputs:

- table_out: the output table sanitized to the desired level k in terms of \$k\$-Anonymity. In this table, the values of each quasi-identifier will be replaced by a pair of values representing the range of values (min and max) that the corresponding row falls in with respect to that quasi-idenfier (exmaples will follow).
- partition_boundaries : a data frame, each row representing the boundaries (minimum, maximum) of a partition used to achieve \$k\$-Anonymity in terms of the quasi-identifers. Note that each partition is an equivalence class. In addition to the boundaries of the partition, each row of this dataframe will include the calculated k value of the same partition. The format of this output is similar to the output of the function kAnonymity_Analyze from Lab 2 (exmaples will follow).

Dimension Choice Functions

A dimension choice function for Mondrian is a function that accepts two arguments:

- partition: The current partition on which the next cut is going to be performed.
- allowable_dims: The list of dimensions with allowable cuts on this partition. This list is represented as a list of column names from partition. You may assume that the provided Mondrian implementation will provide this parameter to your function correctly.

And returns one output:

• dimension: the dimension chosen for the next cut, from the list of dimensions with allowable cuts allowable dims.

Through this function, you may specify the mechanism by which you would like Mondrian to select the next dimension for a cut.

For example, consider the following function Mondrian_choose_dim_random that chooses a dimension at random (with equal probability) from the list of dimensions with allowable cuts (the default behavior described in class):

```
def Mondrian choose dim random(partition, allowable dims):
In [176...
              """A dimension choice function choosing a random dimension (with allowable
             Parameters
             partition : pandas.DataFrame
                 The partition on which to determine dimension for the next cut.
             allowable dims : list
                 A list containing the set of quasi-identifiers (or dimensions) with al
                 You may assume that the provided Mondrian implementation will provide
             Returns
              _ _ _ _ _ _ _
             dimension : string
                 The name of the quasi-identifier (or dimension) for the chosen cut.
             # Since we select the next dimension randomly from allowable dims, we igno
             dimension = allowable dims[numpy.random.randint(len(allowable dims))]
             return dimension
```

Any function with this signature (input arguments and output) can be passed to Mondrian through the dim_choice_fcn argument. Mondrian will use the function specified in dim_choice_fcn to choose, in each step, the dimension for the next cut.

Cut Value Choice Functions

A cut value choice function for Mondrian is a function that accepts two arguments:

- data: A list containing the data values from the dimension chosen for the cut (not a table, just the values from the chosen dimension. E.g., [94705, 94708, 94720, 94708, 94705]).
- k : The requested level of k -Anonymity.

And returns one output:

• cut value: the value chosen for the next cut based on data and k.

Through this function, you may specify the mechanism by which you would like Mondrian to select the value on which to perform the next cut. For example, consider the following function Mondrian_choose_cut_median that chooses a median value from the input data (the default behavior described in class):

Any function with this signature (input arguments and output) can be passed to Mondrian through the cut_choice_fcn argument. Mondrian will use the function specified in cut_choice_fcn to choose, in each step, the cut value for the next cut. This function will also be used by Mondrian to determine which

In this problem, we will study the effect of the different choices of 1) dimension choice functions; and 2) cut value choice functions, on the outcome of Mondrian.

(a) Let's Get Started: The Default Behavior

In this first part, we will famirialize ourselves with the different components of the code and the functions provided in this lab. For that purpose, we will use a simple example that we can analyze manually if needed:

In [178... display(Lab3_Problem1_demo)

	Zipcode	Age
0	94702	31
1	94704	31
2	94703	32
3	94705	32
4	94704	33
5	94703	34
6	94704	34
7	94706	31
8	94706	32
9	94706	34

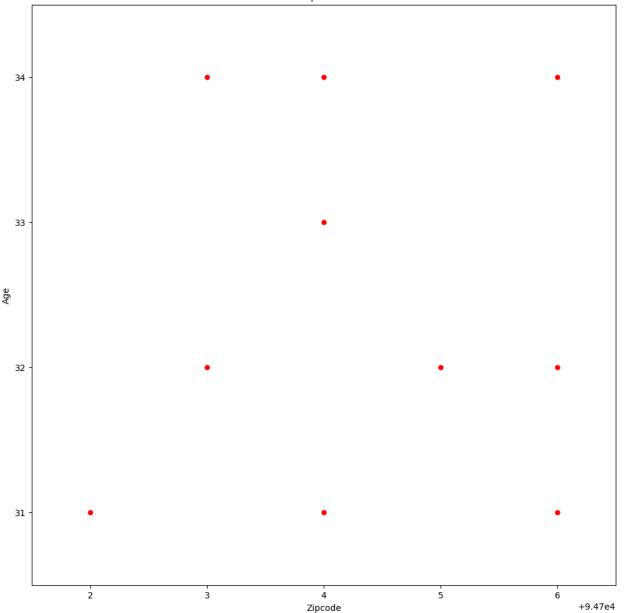
We will use a function provided to you in this lab visualize_data that visualizes 2D data:

```
0.000
assert len(quasi_identifiers) == 2, 'This function is meant to only be use
if not fh:
    pyplot.subplots(figsize=(12,12))
# Here are the dimension names to plot
dim1=quasi identifiers[0]
dim2=quasi identifiers[1]
# Plot our data points
pyplot.scatter(table in.loc[:,dim1], table in.loc[:,dim2], s=25, c='red')
# Focus on the relevant area of the plot
pyplot.xlim([min(table in.loc[:,dim1])-.5, max(table in.loc[:,dim1])+.5])
pyplot.ylim([min(table in.loc[:,dim2])-.5, max(table in.loc[:,dim2])+.5])
# Only the ticks that matter
pyplot.xticks(numpy.unique(table in.loc[:,dim1]))
pyplot.yticks(numpy.unique(table in.loc[:,dim2]))
# Add some labels
pyplot.xlabel(dim1)
pyplot.ylabel(dim2)
pyplot.title('Values of quasi-identifiers')
```

To visualize the same sample dataset:

```
In [180... qID = ['Zipcode', 'Age']
  visualize_data(Lab3_Problem1_demo, qID)
```





If we run the Mondrian algorithm on this data, with

- dim_choice_fcn to select a random dimension from the dimensions
 with allowable cuts (i.e., the function Mondrian_choose_dim_random),
 and
- cut_choice_fcn to select the median value from the values of the given dimension as the split value (i.e., the function Mondrian_choose_cut_median).

We should be able to manualy analyze the possible ways this algorithm can converge.

Using these (already implemented) functions, we run the following code to sanitize

the data from Lab3 Problem1 demo as desired in the description above:

```
In [181... desired_k = 3
    pla_tableOut, pla_boundaries = Mondrian(Lab3_Problem1_demo, qID, desired_k, Mc
    Let's take look at the output table
```

In [182... display(pla_tableOut)

	Zipcode	Age
0	(94702, 94706)	(31, 31)
1	(94702, 94706)	(31, 31)
2	(94702, 94706)	(31, 31)
3	(94703, 94706)	(32, 32)
4	(94703, 94706)	(32, 32)
5	(94703, 94706)	(32, 32)
6	(94703, 94706)	(33, 34)
7	(94703, 94706)	(33, 34)
8	(94703, 94706)	(33, 34)
9	(94703, 94706)	(33, 34)

Note how each quasi-identifier now has a range of values instead of a single value. The new value indicates the range that the original value falls in.

Let's take a look at the different partitions (equivalence classes) returned by Mondrian and the corresponding value \$k\$ of that partition:

```
In [183...
for partition_i in range(pla_boundaries.shape[0]):
    display(Markdown(f'Partition number {partition_i+1}:'))
    display(pla_boundaries.iloc[partition_i,:])
```

Partition number 1:

```
Zipcode (94702, 94706)
Age (31, 31)
k 3
Name: 0, dtype: object
Partition number 2:
Zipcode (94703, 94706)
Age (32, 32)
k 3
Name: 1, dtype: object
Partition number 3:
```

```
Zipcode (94703, 94706)
Age (33, 34)
k 4
Name: 2, dtype: object
```

For each partition, the value of each quasi-identifier is formatted as (minimum, maximum) which are the smallest and largest values of the quasi-identifier within the partition, respectively.

Concisely, we can view all partitions at once:

```
In [184... display(pla_boundaries)
```

	Zipcode	Age	k
0	(94702, 94706)	(31, 31)	3
1	(94703, 94706)	(32, 32)	3
2	(94703, 94706)	(33, 34)	4

Let's verify that this output using our trusted code from Lab 2:

```
In [185...

def kAnonymity_Analyze(data, qID):
        eqv_classes = data.groupby(qID).apply(lambda x: len(x)).reset_index(name='
        return min(eqv_classes.k), eqv_classes

qID = ['Zipcode','Age']
    [actual_k,ec_report] = kAnonymity_Analyze(pla_tableOut, qID)
        display(ec_report)
        display(Markdown(f'Yielding k = {actual_k}'))
```

	Zipcode	Age	K
0	(94702, 94706)	(31, 31)	3
1	(94703, 94706)	(32, 32)	3
2	(94703, 94706)	(33, 34)	4

Yielding k = 3

We see that the results match! Hooray!

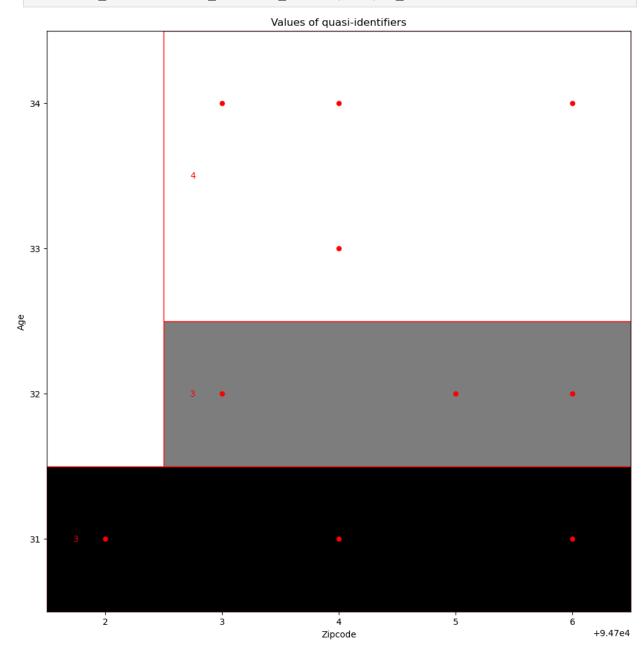
However, it may be easier to visualize the generalized table as we've seen in the async and in discussion. For this purpose, you are provided another function in this lab, visualize_Mondrian, which visulizes the resulting partitions of the function Mondrian and superimposes them on the original data (for cases with \$2\$ quasi-identifiers only):

```
In [186... def visualize_Mondrian(table_in, quasi_identifiers, boundaries):
```

```
"""A function to visualize the outcome of the Mondrian algorithm
Parameters
table in : pandas.DataFrame
   The original table (not the output of Mondrian, but the original non-
quasi identifiers : list
    A list containing the set of quasi-identifiers.
boundaries : pandas.DataFrame
    The second output of the function Mondrian() containing the boundaries
assert len(quasi_identifiers) == 2, 'This function is meant to only be use
# Start a figure
fh, ax= pyplot.subplots(figsize=(12,12))
# Calculate the min distances between datapoints in each dimension
dim granularities = numpy.full((table in.shape[1],1), numpy.nan)
for dim in range(2):
    dim unique = numpy.unique(table in.iloc[:,dim])
    dim granularities[dim] = min(numpy.diff(dim unique))/2
# Generate the colors for the boundary boxes
cols = pyplot.cm.gray(numpy.linspace(0,1,len(boundaries)), alpha=1)
# For each boundary
for boundary i in range(0,len(boundaries)):
    # Get the current boundary dimensions
    curr boundary = boundaries.loc[boundary i,quasi identifiers]
    # These are the x,y coordinates of the bottom left corner
    x = curr boundary.iloc[0][0]-dim granularities[0]
    y = curr boundary.iloc[1][0]-dim granularities[1]
    # These are the width and height of the box
   w = curr boundary.iloc[0][1] - x + dim granularities[0]
   h = curr boundary.iloc[1][1] - y + dim granularities[1]
    # Plot the box
    curr ec = matplotlib.patches.Rectangle((x,y), w, h, linewidth=1, edged)
    ax.add patch(curr ec)
    pyplot.text(x+dim granularities[0]*0.5, y+h/2+dim granularities[1]*0,
visualize data(table in, quasi identifiers, fh)
pyplot.show()
```

We will use this function to visualize the output of Mondrian on our sample

In [187... visualize_Mondrian(Lab3_Problem1_demo, qID, pla_boundaries)



Note how the first argument to the function visualize_Mondrian is the original dataset (the input to Mondrian, not the output of Mondrian), this is because we would like to visualize the partitions on top of the original (non-generalized) dataset.

The function visualize_Mondrian plots bounding boxes around each partition, color them differently, and depict the number of data points in each partition inside the figure.

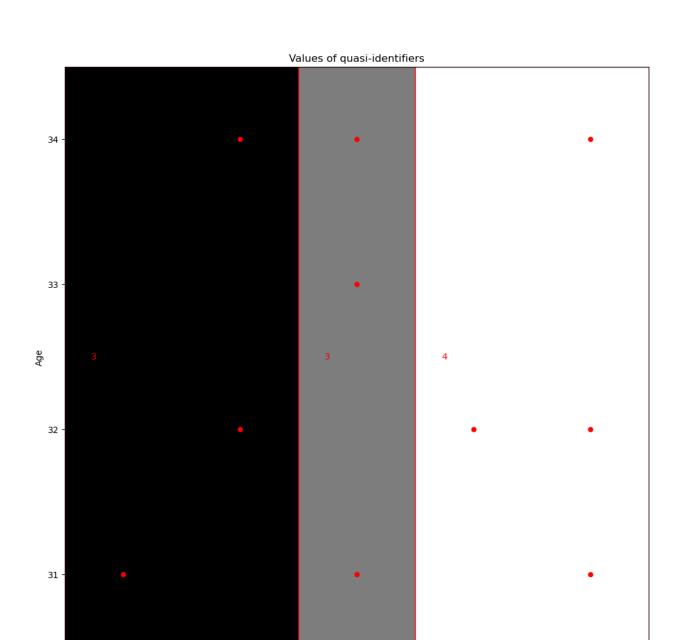
Since the choice of the next dimension to cut is random in this example, the algorithm will potentially output a different partition every run.

Run the following snippet of code repeatedly to determine the number of different outputs this algorithm can generate, and store that value in the variable pla nr outputs. You may hardcode the answer in your code.

Complete the following code snippet with your answer:

```
In [188... qID = ['Zipcode','Age']
    desired_k = 3
    [pla_tableOut, pla_boundaries] = Mondrian(Lab3_Problem1_demo, qID, desired_k,
    visualize_Mondrian(Lab3_Problem1_demo, qID, pla_boundaries)

#pla_nr_outputs = numpy.nan
    pla_nr_outputs = 4
```



Run the following cell to print the outcomes of your code.

```
In [189... __problem = 'la'

try:
    print_mk(f'**Question**: How many possible outputs can `Mondrian` output f
except Exception as e:
    safe_print_err(e)
```

Zipcode

+9.47e4

Question: How many possible outputs can Mondrian output for the sample dataset?

Answer: \$4\$

(b) More Data, Default Behavior

In this part, you will run the Mondrian algorithm on Lab3_Data on various sets of quasi-identifiers and \$k\$ values. All of the trials in this part should use the dimension choice function Mondrian_choose_dim_random and the cut value choice function Mondrian choose cut median.

Concretely, run the function Mondrian on Lab3_Data five times, and assign the results in the following variables:

- plb_tableOut_ra_20 and plb_boundaries_ra_20: For the results of Mondrian on Lab3_Data with qID = ['Race', 'Age'] and \$k=20\$.
 - p1b_actual_k_ra_20 : Store the resulting \$k\$ value of the table after generalization.
- plb_tableOut_ra_40 and plb_boundaries_ra_40: For the results of Mondrian on Lab3_Data with qID = ['Race', 'Age'] and \$k=40\$.
 - p1b_actual_k_ra_40 : Store the resulting \$k\$ value of the table after generalization.
- p1b_tableOut_ra_80 and p1b_boundaries_ra_80: For the results of Mondrian on Lab3_Data with qID = ['Race', 'Age'] and \$k=80\$.
 - p1b_actual_k_ra_80 : Store the resulting \$k\$ value of the table after generalization.
- plb_tableOut_rm_20 and plb_boundaries_rm_20: For the results of Mondrian on Lab3_Data with qID = ['Race', 'Marital Status'] and \$k=20\$.
 - p1b_actual_k_rm_20 : Store the resulting \$k\$ value of the table after generalization.
- plb_tableOut_gm_40 and plb_boundaries_gm_40: For the results of Mondrian on Lab3_Data with qID = ['Gender', 'Marital Status'], and \$k=40\$.
 - p1b_actual_k_gm_40 : Store the resulting \$k\$ value of the table after generalization.

Replace the following code snippet with your answer:

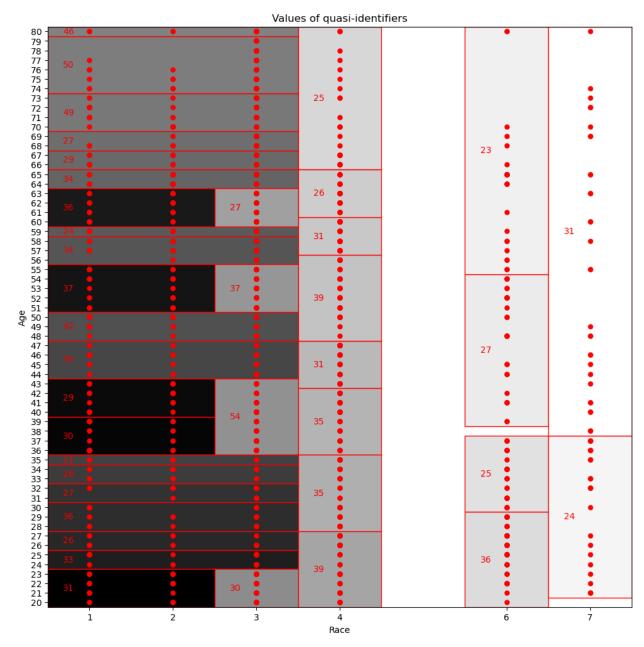
```
In [190... # Your solution goes here
qID = ['Race', 'Age']
desired_k = 20
```

```
[plb tableOut ra 20, plb boundaries ra 20] = Mondrian(Lab3 Data, qID, desired
[plb actual k ra 20, ec report 20] = kAnonymity Analyze(plb tableOut ra 20, qI
qID = ['Race', 'Age']
desired k = 40
[plb tableOut ra 40, plb boundaries ra 40] = Mondrian(Lab3 Data, qID, desired
[plb actual k ra 40, ec report 40] = kAnonymity Analyze(plb tableOut ra 40, qI
qID = ['Race', 'Age']
desired k = 80
[plb tableOut ra 80, plb boundaries ra 80] = Mondrian(Lab3 Data, qID, desired
[p1b actual k ra 80, ec report 80] = kAnonymity Analyze(p1b tableOut ra 80, q1
qID = ['Race', 'Marital Status']
desired k = 20
[plb tableOut rm 20, plb boundaries rm 20] = Mondrian(Lab3 Data, qID, desired
[p1b actual k rm 20, ec report rm 20] = kAnonymity Analyze(p1b tableOut rm 20,
qID = ['Gender', 'Marital Status']
desired k = 40
[plb tableOut gm 40, plb boundaries gm 40] = Mondrian(Lab3 Data, qID, desired
[p1b actual k gm 40, ec report gm 40] = kAnonymity Analyze(p1b tableOut gm 40,
```

Run the following cell to present the outcomes of your code.

```
In [191... | problem = '1b'
         try:
             to_print = [{'qID': ['Race', 'Age'], 'k': 20, 'tableOut': p1b_tableOut_ra_
                         {'qID': ['Race', 'Age'], 'k': 40, 'tableOut': p1b tableOut ra
                         {'qID': ['Race', 'Age'], 'k': 80, 'tableOut': p1b_tableOut_ra_
                         {'qID': ['Race', 'Marital Status'], 'k': 20, 'tableOut': plb t
                         {'qID': ['Gender', 'Marital Status'], 'k': 40, 'tableOut': plb
             for next_to_print in to print:
                 next qID = next to print["qID"]
                 next k = next to print["k"]
                 next tableOut = next to print["tableOut"]
                 next boundaries = next to print["boundaries"]
                 next actual k = next to print["actual k"]
                 print mk(f'The results of the run for qID={next qID} with $k={next k}$
                 visualize_Mondrian(Lab3_Data, next_qID, next_boundaries)
                 print mk('The generalized table is:')
                 display(next tableOut)
                 print_mk(f'The actual $k$ in the resulting generalization was ${next_a
                 print mk('<br>')
         except Exception as e:
             safe print err(e)
```

The results of the run for qID=['Race', 'Age'] with \$k=20\$ are:



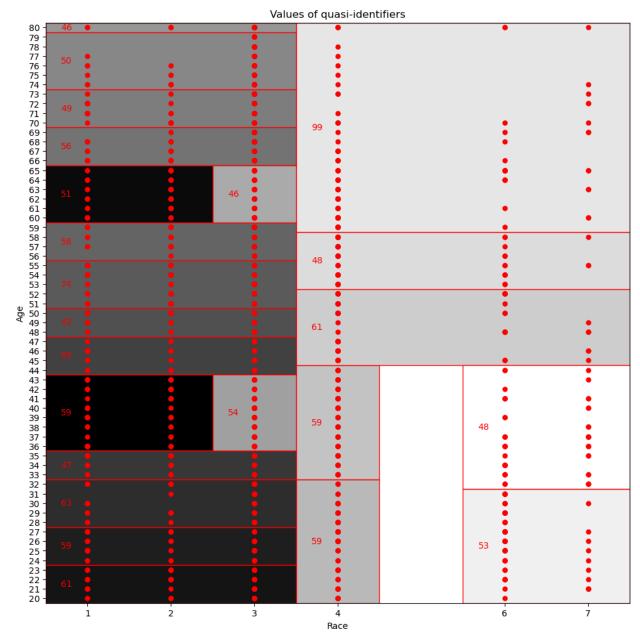
The generalized table is:

	Subject_ID	Gender	Age	Marital_Status	Country_Birth	Race	Alcohol_Aver
0	83988	1	(20, 23)	5	1	(1, 2)	
1	84587	2	(20, 23)	5	1	(1, 2)	
2	84748	1	(20, 23)	6	1	(1, 2)	
3	84964	2	(20, 23)	1	2	(1, 2)	
4	85042	1	(20, 23)	6	2	(1, 2)	
1296	91292	1	(38, 80)	5	1	(7, 7)	
1297	92486	1	(38, 80)	1	1	(7, 7)	
1298	92494	2	(38, 80)	1	1	(7, 7)	
1299	92625	2	(38, 80)	1	1	(7, 7)	
1300	93268	2	(38, 80)	3	1	(7, 7)	

1301 rows × 11 columns

The actual \$k\$ in the resulting generalization was \$21\$ (reference: requested \$k\$ was \$20\$).

The results of the run for qID=['Race', 'Age'] with k=40 are:



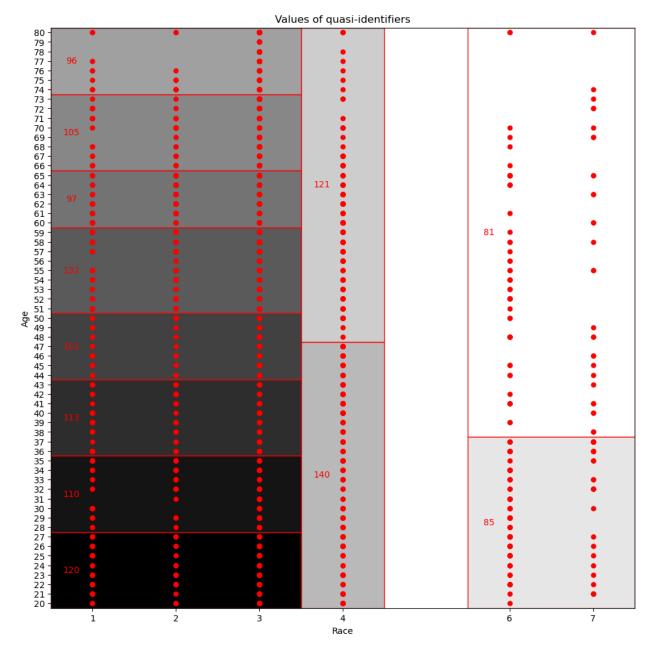
The generalized table is:

	Subject_ID	Gender	Age	Marital_Status	Country_Birth	Race	Alcohol_Aver
0	83828	2	(36, 43)	1	2	(1, 2)	
1	84073	2	(36, 43)	5	1	(1, 2)	
2	84329	1	(36, 43)	6	2	(1, 2)	
3	84363	2	(36, 43)	1	2	(1, 2)	
4	84626	1	(36, 43)	3	2	(1, 2)	
					•••		
1296	92663	2	(32, 44)	1	2	(6, 7)	
1297	92994	1	(32, 44)	1	2	(6, 7)	
1298	93373	1	(32, 44)	3	2	(6, 7)	
1299	93504	1	(32, 44)	1	2	(6, 7)	
1300	93550	1	(32, 44)	1	2	(6, 7)	

1301 rows × 11 columns

The actual \$k\$ in the resulting generalization was \$42\$ (reference: requested \$k\$ was \$40\$).

The results of the run for qID=['Race', 'Age'] with k=80 are:



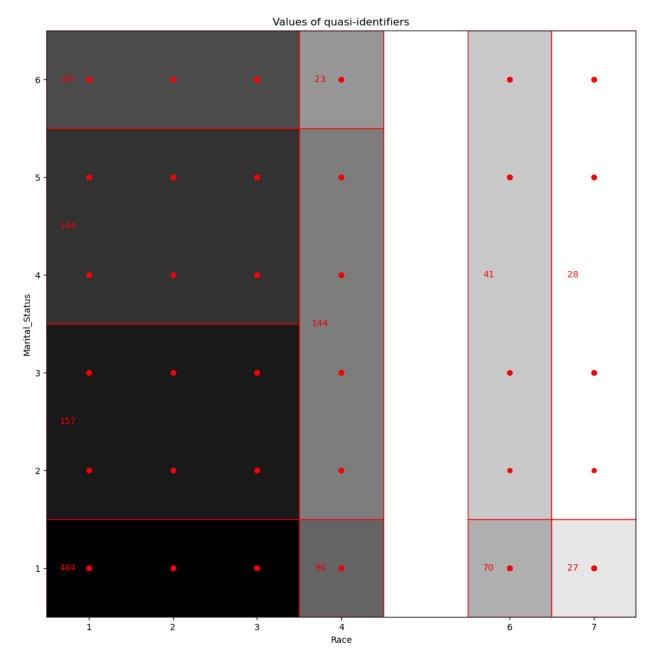
The generalized table is:

	Subject_ID	Gender	Age	Marital_Status	Country_Birth	Race	Alcohol_Aver
0	83813	1	(20, 27)	3	1	(1, 3)	
1	83816	1	(20, 27)	6	1	(1, 3)	
2	83844	1	(20, 27)	6	2	(1, 3)	
3	83988	1	(20, 27)	5	1	(1, 3)	
4	84058	2	(20, 27)	5	1	(1, 3)	
1296	92850	1	(38, 80)	1	1	(6, 7)	
1297	92895	1	(38, 80)	1	2	(6, 7)	
1298	93268	2	(38, 80)	3	1	(6, 7)	
1299	93373	1	(38, 80)	3	2	(6, 7)	
1300	93374	2	(38, 80)	1	2	(6, 7)	

1301 rows × 11 columns

The actual k in the resulting generalization was \$81\$ (reference: requested k was 80).

The results of the run for qID=['Race', 'Marital_Status'] with k=20 are:



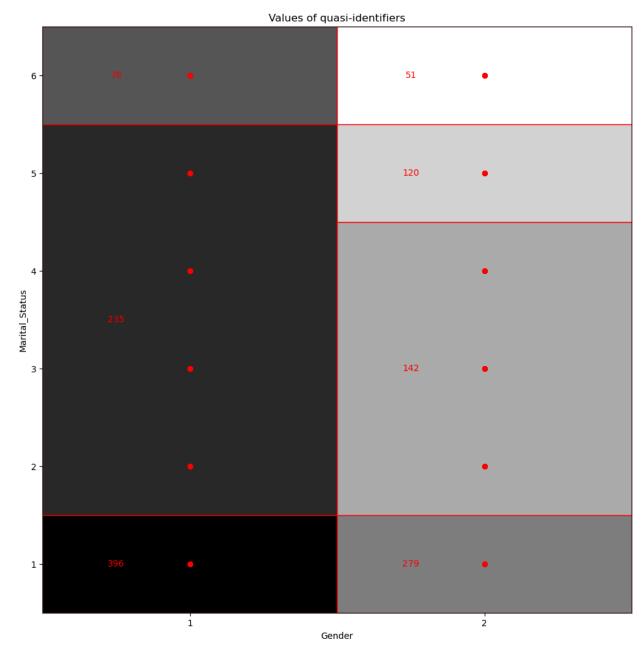
The generalized table is:

	Subject_ID	Gender	Age	Marital_Status	Country_Birth	Race	Alcohol_Aver
0	83754	2	67	(1, 1)	1	(1, 3)	
1	83801	2	80	(1, 1)	1	(1, 3)	
2	83828	2	39	(1, 1)	2	(1, 3)	
3	83851	2	37	(1, 1)	1	(1, 3)	
4	83854	2	46	(1, 1)	1	(1, 3)	
1296	91742	1	23	(2, 6)	1	(7, 7)	
1297	91807	2	25	(2, 6)	1	(7, 7)	
1298	92044	1	21	(2, 6)	1	(7, 7)	
1299	92417	1	37	(2, 6)	1	(7, 7)	
1300	93268	2	73	(2, 6)	1	(7, 7)	

1301 rows × 11 columns

The actual \$k\$ in the resulting generalization was \$23\$ (reference: requested \$k\$ was \$20\$).

The results of the run for $qID=['Gender', 'Marital_Status']$ with k=40 are:



The generalized table is:

	Subject_ID	Gender	Age	Marital_Status	Country_Birth	Race	Alcohol_Aver
0	83860	(1, 1)	41	(1, 1)	1	4	
1	83863	(1, 1)	35	(1, 1)	2	1	
2	83886	(1, 1)	74	(1, 1)	1	3	
3	83894	(1, 1)	60	(1, 1)	1	4	
4	83908	(1, 1)	51	(1, 1)	2	4	
1296	92358	(2, 2)	21	(6, 6)	1	3	
1297	92914	(2, 2)	65	(6, 6)	1	3	
1298	93002	(2, 2)	30	(6, 6)	1	3	
1299	93206	(2, 2)	44	(6, 6)	2	2	
1300	93325	(2, 2)	28	(6, 6)	1	3	

1301 rows × 11 columns

The actual \$k\$ in the resulting generalization was \$51\$ (reference: requested \$k\$ was \$40\$).

In this Markdown cell, answer the following question

• Are the actual levels of \$k\$-Anonymity of the table after generalization always **equal** to the desired/requested levels of \$k\$-Anonymity? Why is that the case?

Place your answer here No, the actual levels of \$k\$-Anonymity of the table after generalization are not always **equal** to the desired/requested levels of \$k\$-Anonymity. Since we will get the same or better privacy after generalization, we will get the actual levels of \$k\$-Anonymity of the table to be equal or higher than the desired/requested levels of \$k\$-Anonymity

(c) Not Median This Time

In this part, we will implement the cut value choice mechanism Mondrian_choose_cut_first_split .

This mechanism is for Mondrian to select the split value for the cut (cf. line 9 in the algorithm), instead of being the median, this time based on the first value in the sorted order of values that permits both cuts to be of size at least \$k\$. If no such value exists, your function should return numpy.nan.

For example, if our dimension has the following values [1, 2, 3, 3, 3, 3, 4, 5, 6, 6, 6, 10, 10, 10] with \$k=5\$, the function

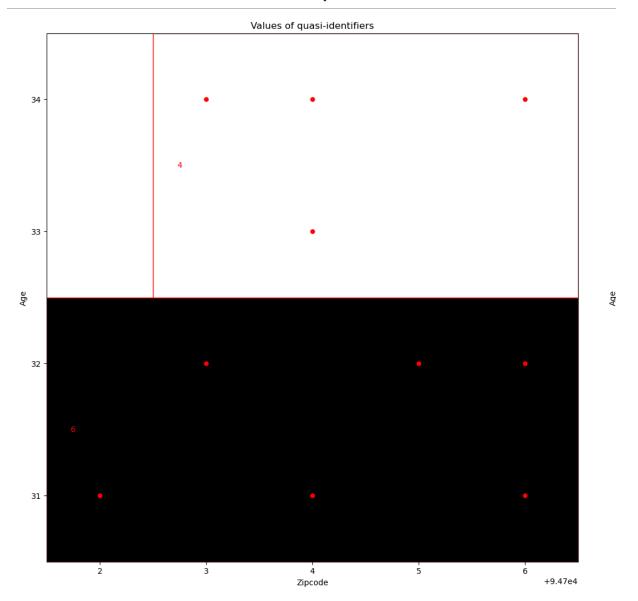
Mondrian_choose_cut_first_split should select \$3\$ as the cut value because it is the smallest value that will yield at least five values to its left (to be exact, there are values to its left in this example: [1, 2, 3, 3, 3, 3,]) and at least five values to its right (to be exact, there are nine values in this example: [4, 5, 6, 6, 6, 6, 10, 10, 10]).

Complete the implementation of the function Mondrian choose cut first split below.

```
In [192... def Mondrian choose cut first split(data,k):
             # Your solution here
             #raise Exception("Implementation of <i>Mondrian choose cut first split</i>
             cut value = 0
             try:
                 data.sort()
                 print(data)
                 if (k > (len(data) // 2)):
                     return numpy.nan
                  cut value = data[k-1]
                  # kl is length of second part of list
                  k1 = len(data) - k
                  if (k1 < k):
                     return numpy.nan
                  for i in range(k, len(data)-1):
                     # kl is length of second part of list
                      k1 = len(data) - 1 - i
                     if (k1 < k):
                          break
                     if (cut value == data[i]):
                         cut value = data[i]
             except:
                  cut value = numpy.nan
             return cut value
```

Once you complete the implementation of Mondrian_choose_cut_first_split, use the following code for a sanity check. The only two possible outputs of the following code are listed in the following table. Confirm that the code is generating both of these outputs, and no other output.

Possible Output #1

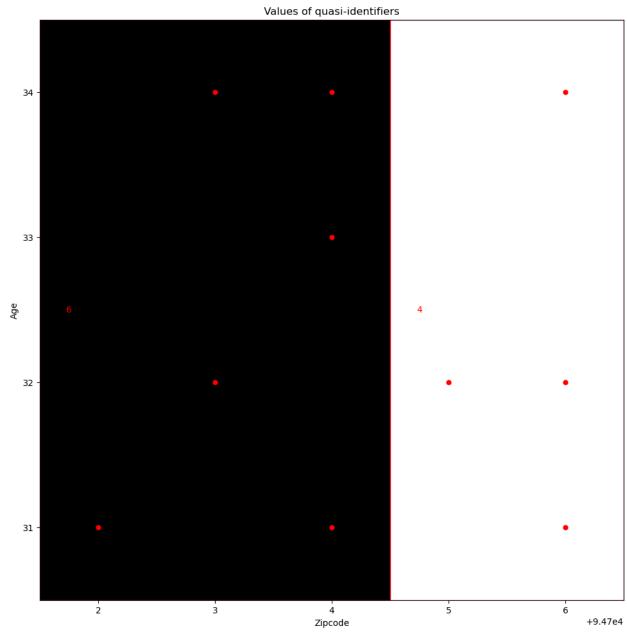


```
In [193... __problem = 'lc'

try:
    qID = ['Zipcode','Age']
    desired_k = 4
    plc_tableOut, plc_boundaries = Mondrian(Lab3_Problem1_demo, qID,desired_k,
    visualize_Mondrian(Lab3_Problem1_demo, qID, plc_boundaries)

except Exception as e:
    safe_print_err(e)
```

```
[94702, 94703, 94703, 94704, 94704, 94704, 94705, 94706, 94706, 94706]
[31, 31, 31, 32, 32, 32, 33, 34, 34, 34]
[94702, 94703, 94703, 94704, 94704, 94704, 94705, 94706, 94706, 94706]
[94702, 94703, 94703, 94704, 94704, 94704]
[31, 31, 32, 33, 34, 34]
[94705, 94706, 94706, 94706]
[31, 32, 32, 34]
```



(d) Not Median This Time - Continued

In this part, we will compare the behavior of Mondrian between using the Mondrian_choose_cut_first_split cut value choice mechanism, and the

Mondrian choose cut median cut value choice mechanism.

Concretely, run the function Mondrian on Lab3_Data twice, and assign the results in the following variables:

- pld_tableOut_cfs and pld_boundaries_cfs: For the results of Mondrian on Lab3_Data with qID = ['Race', 'Marital_Status'] and \$k=60\$ using the Mondrian choose cut first split cut value choice mechanism.
 - pld_actual_k_cfs: Store the resulting \$k\$ value of the table after generalization.
- pld_tableOut_cm and pld_boundaries_cm: For the results of Mondrian on Lab3_Data with qID = ['Race', 'Marital_Status'] and \$k=60\$ using the Mondrian_choose_cut_median cut value choice mechanism.
 - pld_actual_k_cm : Store the resulting \$k\$ value of the table after generalization.

In both cases, use the Mondrian_choose_dim_random dimension choice mechanism.

Complete the following code snippet with your code:

```
In [194... qID=['Race','Marital_Status']
  desired_k=60

# Your solution here
  [pld_tableOut_cfs, pld_boundaries_cfs] = Mondrian(Lab3_Data, qID, desired_k, M
  [pld_actual_k_cfs, ec_report_cfs] = kAnonymity_Analyze(pld_tableOut_cfs, qID)

  [pld_tableOut_cm, pld_boundaries_cm] = Mondrian(Lab3_Data, qID, desired_k, Mon [pld_actual_k_cm, ec_report_cm] = kAnonymity_Analyze(pld_tableOut_cm, qID)
```

```
3, 3,
3, 3,
3, 3,
7]
```

```
2, 2,
3, 3,
5, 5,
6]
```

```
5, 5,
6]
```

```
3, 3,
2, 2,
```

```
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1]
```

```
7, 7, 7, 7, 7]
1, 1, 1, 1, 1]
7, 7, 7, 7, 7]
```

```
7, 7, 7, 7, 7, 7, 7, 7]
1, 1, 1, 1, 1, 1, 1, 1, 1]
7, 7, 7, 7, 7, 7, 7, 7]
```

```
7, 7]
6, 6]
```

```
6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
5, 5,
5, 5,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7]
7, 7, 7]
5, 5, 5]
```

```
7, 7, 7]
7, 7, 7, 7, 7, 7, 7]
5, 5, 5, 5, 5, 5, 5]
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7]
```

Run the following cell to present the outcomes of your code.

```
In [195... __problem = 'ld'

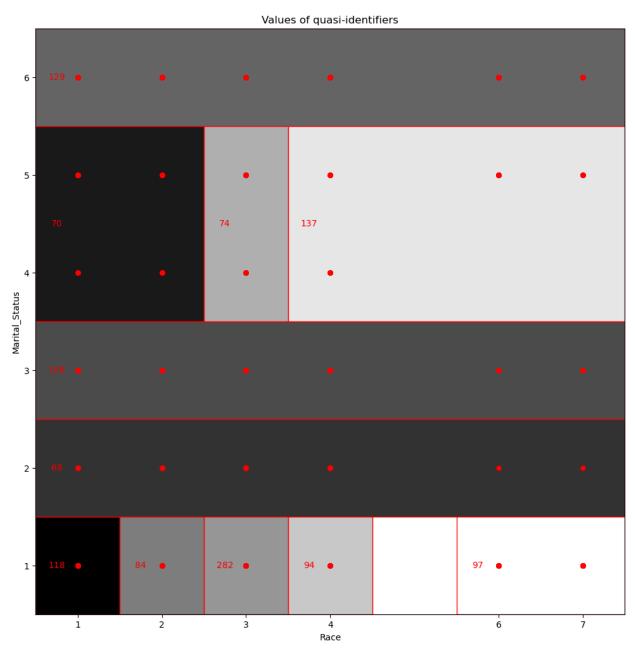
try:
    qID=['Race','Marital_Status']
    desired_k=60

# Case 1
    print_mk(f'The results of the run using `Mondrian_choose_cut_first_split`
    visualize_Mondrian(Lab3_Data, qID, pld_boundaries_cfs)
    print_mk('The generalized table is:')
    display(pld_tableOut_cfs)
    print_mk(f'The actual $k$ in the resulting generalization was ${pld_actual print_mk('<br>>br><br/># Case 2</br>
    print_mk(f'The results of the run using `Mondrian_choose_cut_median` with visualize_Mondrian(Lab3_Data, qID, pld_boundaries_cm)
```

```
print_mk('The generalized table is:')
display(pld_tableOut_cm)
print_mk(f'The actual $k$ in the resulting generalization was ${pld_actual print_mk('<br>')

except Exception as e:
    safe_print_err(e)
```

The results of the run using Mondrian_choose_cut_first_split with qID=['Race', 'Marital_Status'] and \$k=60\$ are:



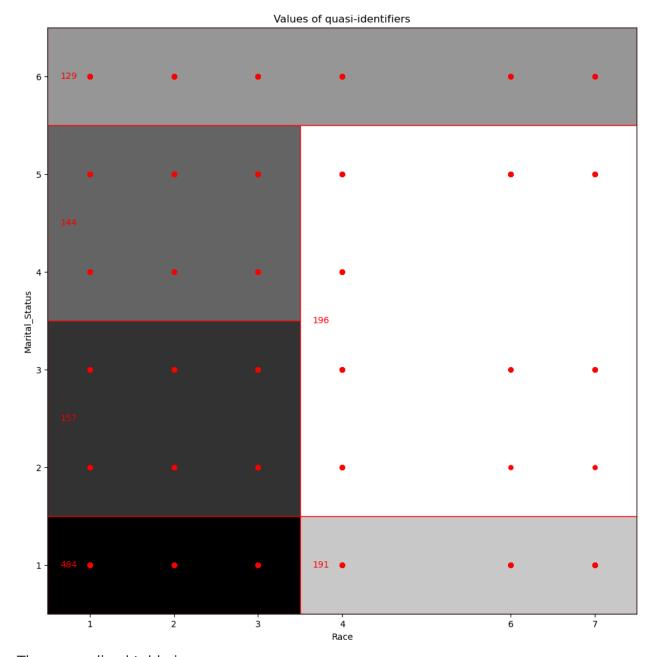
The generalized table is:

	Subject_ID	Gender	Age	Marital_Status	Country_Birth	Race	Alcohol_Aver
0	83828	2	39	(1, 1)	2	(1, 1)	
1	83854	2	46	(1, 1)	1	(1, 1)	
2	83863	1	35	(1, 1)	2	(1, 1)	
3	83987	1	68	(1, 1)	2	(1, 1)	
4	84029	2	28	(1, 1)	1	(1, 1)	
1296	93330	2	30	(1, 1)	2	(6, 7)	
1297	93370	1	30	(1, 1)	2	(6, 7)	
1298	93374	2	50	(1, 1)	2	(6, 7)	
1299	93504	1	37	(1, 1)	2	(6, 7)	
1300	93550	1	37	(1, 1)	2	(6, 7)	

1301 rows × 11 columns

The actual \$k\$ in the resulting generalization was \$68\$ (reference: requested \$k\$ was \$60\$).

The results of the run using Mondrian_choose_cut_median with $qID=['Race', 'Marital_Status']$ and k=60 are:



The generalized table is:

	Subject_ID	Gender	Age	Marital_Status	Country_Birth	Race	Alcohol_Aver
0	83754	2	67	(1, 1)	1	(1, 3)	
1	83801	2	80	(1, 1)	1	(1, 3)	
2	83828	2	39	(1, 1)	2	(1, 3)	
3	83851	2	37	(1, 1)	1	(1, 3)	
4	83854	2	46	(1, 1)	1	(1, 3)	
1296	93482	1	29	(2, 5)	1	(4, 7)	
1297	93499	2	30	(2, 5)	1	(4, 7)	
1298	93503	2	54	(2, 5)	1	(4, 7)	
1299	93528	1	27	(2, 5)	1	(4, 7)	
1300	93582	1	20	(2, 5)	2	(4, 7)	

1301 rows × 11 columns

The actual \$k\$ in the resulting generalization was \$129\$ (reference: requested \$k\$ was \$60\$).

(e) More Than Random

In this part, we will implement the dimension choice mechanism Mondrian choose dim highest distinct.

This mechanism is for Mondrian to select the next dimension for a cut (cf. line 7 in the algorithm), instead of randomly, this time based on the dimension with the highest number of distinct values **out of the dimensions with allowable cuts**. Note that the function should return **the name** of the dimension as a string (i.e., not its index).

For example, if we have two dimensions with allowable cuts:

- 'Q1' with values [10, 10, 10, 10, 20, 20, 20, 20, 20]; and
- 'Q2' with values [1, 1, 2, 2, 3, 3, 3, 3, 3, 3]

Then Mondrian_choose_dim_highest_distinct should select 'Q2' since it has \$3\$ distinct values (as opposed to \$2\$ distinct values for 'Q1'). In case of a tie between multiple dimensions, your function should output the dimension (out of the tying dimensions) that appears first from the list of dimensions allowable dims.

Complete the implementation of the function

Mondrian choose dim highest distinct below.

```
In [196... def Mondrian_choose_dim_highest_distinct(table_partition, allowable_dims):
    # Your solution here
    #raise Exception("Implementation of <i>Mondrian_choose_dim_highest_distinc

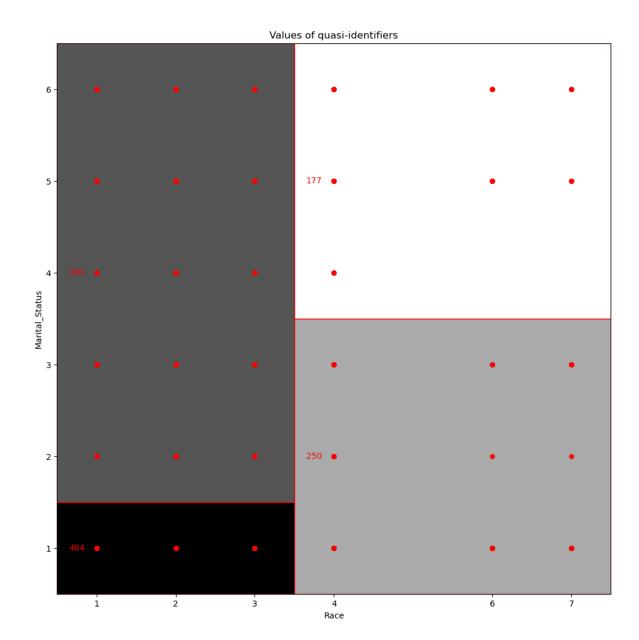
dim_dict = {}
    for i in range(len(allowable_dims)):
        dimension = allowable_dims[i]
        dim_boundry_values = table_partition.loc[:, dimension]
        dim_dict[dimension] = len(numpy.unique(dim_boundry_values))

dim_dict = dict(sorted(dim_dict.items(), key=lambda item: item[1], reverse dimension = list(dim_dict)[0]

return dimension
```

Once you complete the implementation of

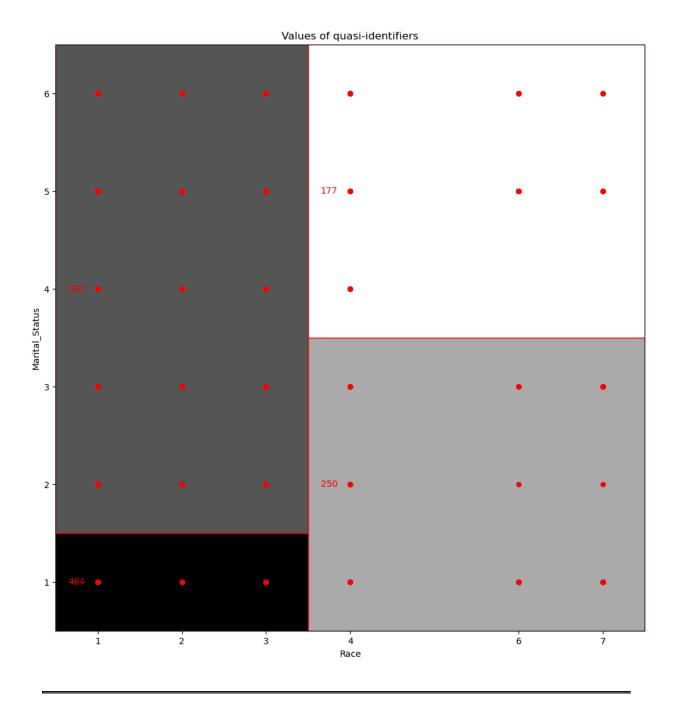
Mondrian_choose_dim_highest_distinct, use the following code for a sanity check. The only possible outputs of the following code is depicted below. Confirm that the code is generating this output, and no other output.



```
In [197... __problem = 'le'

try:
    qID = ['Race', 'Marital_Status']
    desired_k = 150
    ple_tableOut, ple_boundaries = Mondrian(Lab3_Data, qID, desired_k, Mondrian visualize_Mondrian(Lab3_Data, qID, ple_boundaries)

except Exception as e:
    safe_print_err(e)
```



(f) More Than Random - Continued

In this part, we will investigate the behavior of Mondrian using the Mondrian_choose_dim_highest_distinct dimension choice mechanism.

Concretely, run the function Mondrian on Lab3_Data twice, and assign the results in the following variables:

• plf_tableOut_cfs and plf_boundaries_cfs: For the results of Mondrian on Lab3 Data with qID = ['Race',

'Marital_Status'] and \$k=60\$ using the
Mondrian choose cut first split cut value choice mechanism.

- p1f_actual_k_cfs : Store the resulting \$k\$ value of the table after generalization.
- plf_tableOut_cm and plf_boundaries_cm: For the results of Mondrian on Lab3_Data with qID = ['Race', 'Marital_Status'] and \$k=60\$ using the Mondrian choose cut median cut value choice mechanism.
 - plf_actual_k_cm : Store the resulting \$k\$ value of the table after generalization.

In both cases, use the Mondrian_choose_dim_highest_distinct dimension choice mechanism.

Complete the following code snippet with your code:

```
In [198... qID=['Race','Marital_Status']
    desired_k=60

# Your solution here
[plf_tableOut_cfs, plf_boundaries_cfs] = Mondrian(Lab3_Data, qID, desired_k, N
[plf_actual_k_cfs, ec_report_cfs] = kAnonymity_Analyze(plf_tableOut_cfs, qID)

[plf_tableOut_cm, plf_boundaries_cm] = Mondrian(Lab3_Data, qID, desired_k, Mon
[plf_actual_k_cm, ec_report_cm] = kAnonymity_Analyze(plf_tableOut_cm, qID)
```

```
3, 3,
3, 3,
3, 3,
7]
```

```
2, 2,
3, 3,
5, 5,
6]
```

```
2, 2,
3, 3,
7]
1, 1, 1]
```

```
6, 6, 6
6, 6, 6]
[2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
```

```
2, 2,
```

```
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
3, 3,
```

```
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1]
7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1]
7, 7, 7, 7, 7]
7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1, 1, 1, 1]
7, 7, 7, 7, 7, 7, 7, 7]
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2,
```

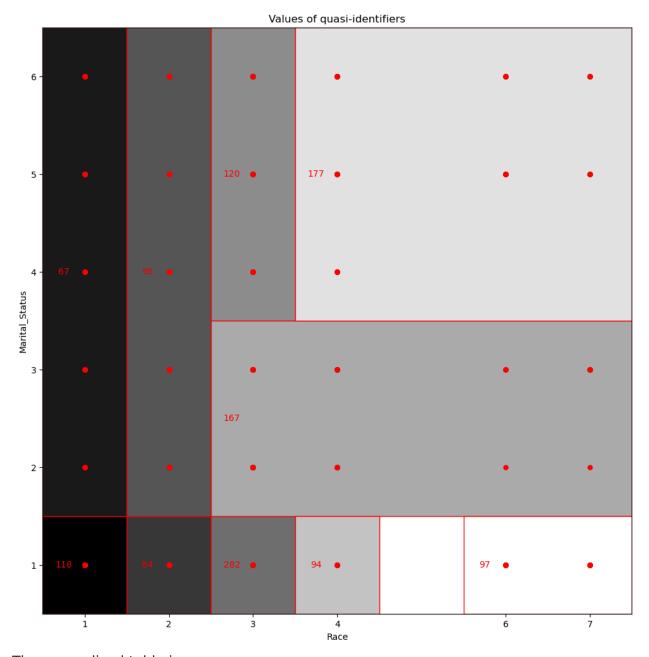
```
5, 5,
```

```
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3]
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

Run the following cell to present the outcomes of your code.

```
In [199... problem = '1f'
         try:
             qID=['Race','Marital_Status']
             desired k=60
             # Case 1
             print mk(f'The results of the run using `Mondrian choose cut first split`
             visualize Mondrian(Lab3 Data, qID, p1f boundaries cfs)
             print mk('The generalized table is:')
             display(p1f_tableOut_cfs)
             print mk(f'The actual $k$ in the resulting generalization was ${plf actual}
             print mk('<br>>')
             # Case 2
             print mk(f'The results of the run using `Mondrian choose cut median` with
             visualize_Mondrian(Lab3_Data, qID, p1f_boundaries_cm)
             print mk('The generalized table is:')
             display(p1f tableOut cm)
             print mk(f'The actual $k$ in the resulting generalization was ${plf actual
             print mk('<br>>')
         except Exception as e:
             safe print err(e)
```

The results of the run using Mondrian_choose_cut_first_split with qID=['Race', 'Marital_Status'] and \$k=60\$ are:

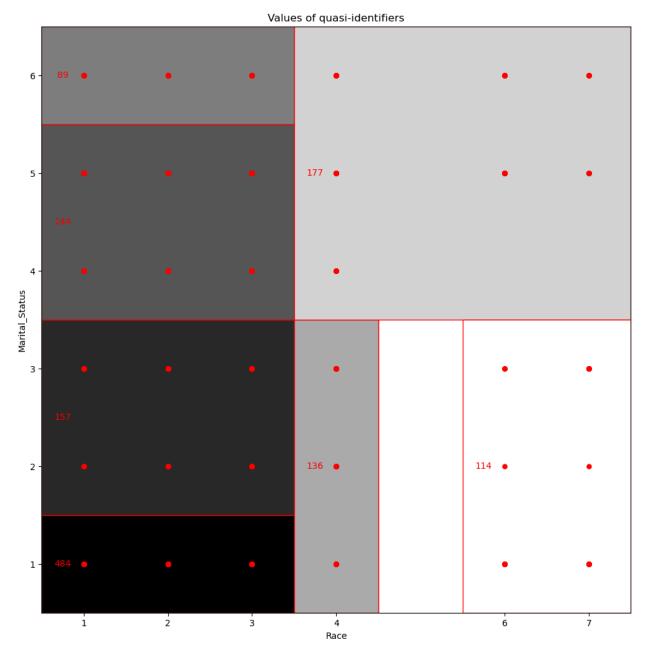


	Subject_ID	Gender	Age	Marital_Status	Country_Birth	Race	Alcohol_Aver
0	83828	2	39	(1, 1)	2	(1, 1)	
1	83854	2	46	(1, 1)	1	(1, 1)	
2	83863	1	35	(1, 1)	2	(1, 1)	
3	83987	1	68	(1, 1)	2	(1, 1)	
4	84029	2	28	(1, 1)	1	(1, 1)	
1296	93330	2	30	(1, 1)	2	(6, 7)	
1297	93370	1	30	(1, 1)	2	(6, 7)	
1298	93374	2	50	(1, 1)	2	(6, 7)	
1299	93504	1	37	(1, 1)	2	(6, 7)	
1300	93550	1	37	(1, 1)	2	(6, 7)	

1301 rows × 11 columns

The actual \$k\$ in the resulting generalization was \$67\$ (reference: requested \$k\$ was \$60\$).

The results of the run using Mondrian_choose_cut_median with $qID=['Race', 'Marital_Status']$ and k=60 are:



	Subject_ID	Gender	Age	Marital_Status	Country_Birth	Race	Alcohol_Aver
0	83754	2	67	(1, 1)	1	(1, 3)	
1	83801	2	80	(1, 1)	1	(1, 3)	
2	83828	2	39	(1, 1)	2	(1, 3)	
3	83851	2	37	(1, 1)	1	(1, 3)	
4	83854	2	46	(1, 1)	1	(1, 3)	
1296	93370	1	30	(1, 3)	2	(6, 7)	
1297	93373	1	44	(1, 3)	2	(6, 7)	
1298	93374	2	50	(1, 3)	2	(6, 7)	
1299	93504	1	37	(1, 3)	2	(6, 7)	
1300	93550	1	37	(1, 3)	2	(6, 7)	

1301 rows x 11 columns

The actual \$k\$ in the resulting generalization was \$89\$ (reference: requested \$k\$ was \$60\$).

(g) One More Dimension Selection Mechanism

In this part, we will implement the dimension choice mechanism Mondrian choose dim highest var.

This mechanism is for Mondrian to select the next dimension for a cut (cf. line 7 in the algorithm), instead of randomly or based on the dimension with the highest number of distinct values, this time based on the dimension with the highest variance **out of the dimensions with allowable cuts**. Note that the function should return **the name** of the dimension as a string (i.e., not its index). In case of a tie between multiple dimensions, your function should output the dimension (out of the tying dimensions) that appears first from the list of dimensions allowable_dims.

For example, if we have two dimensions with allowable cuts:

• 'Q1' with values [10, 10, 10, 10, 20, 20, 20, 20, 20]

```
(has variance $25$); and'Q2' with values [1, 1, 2, 2, 3, 3, 3, 3, 3] (has variance
```

Then Mondrian_choose_dim_highest_var should select 'Q1' since it has higher variance than 'Q2'.

Complete the implementation of the function Mondrian_choose_dim_highest_var below.

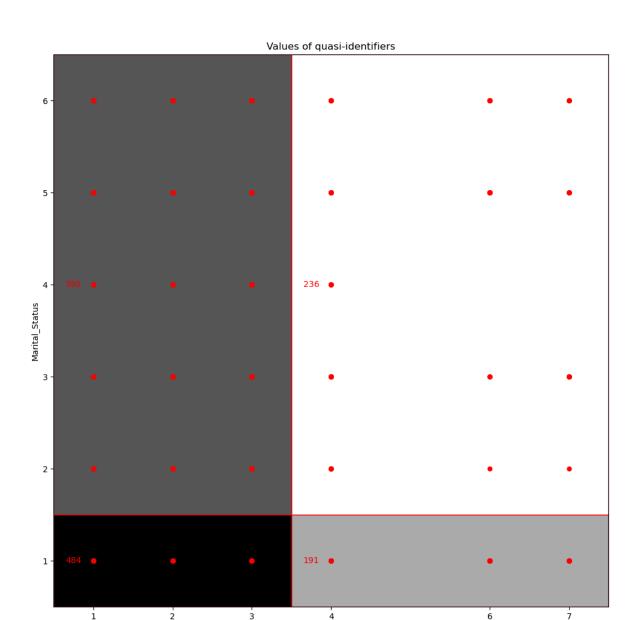
\$0.64\$)

```
In [200... def Mondrian_choose_dim_highest_var(table_partition, allowable_dims):
    # Your solution here
    #raise Exception("Implementation of <i>Mondrian_choose_dim_highest_var</i>
    dim_dict = {}
    for i in range(len(allowable_dims)):
        dimension = allowable_dims[i]
        dim_boundry_values = table_partition.loc[:, dimension]
        dim_dict[dimension] = numpy.var(dim_boundry_values)

dim_dict = dict(sorted(dim_dict.items(), key=lambda item: item[1], reverse dimension = list(dim_dict)[0]

return dimension
```

Once you complete the implementation of Mondrian_choose_dim_highest_var , use the following code for a sanity check. The only possible outputs of the following code is depicted below. Confirm that the code is generating this output, and no other output.

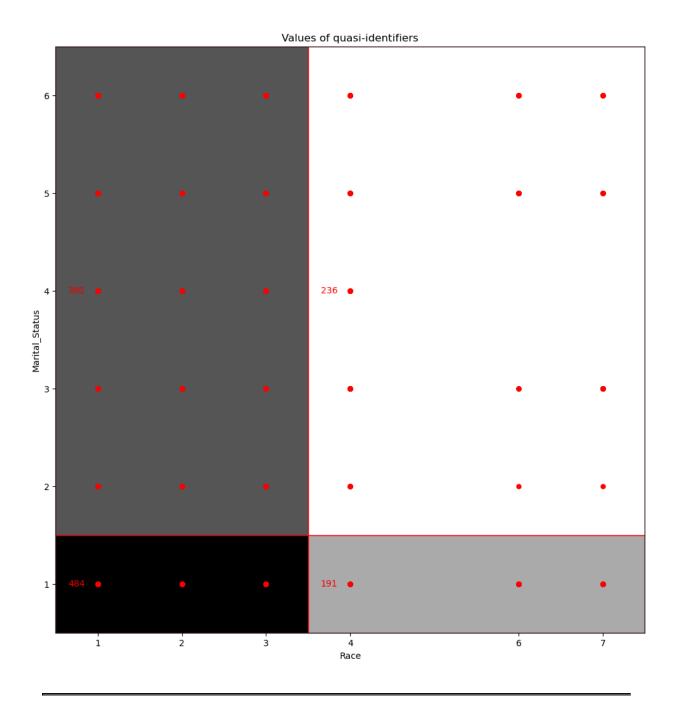


```
In [201... __problem = 'lg'

try:
    qID = ['Race', 'Marital_Status']
    desired_k = 150
    [plg_tableOut, plg_boundaries] = Mondrian(Lab3_Data,qID,desired_k,Mondrian visualize_Mondrian(Lab3_Data, qID, plg_boundaries)

except Exception as e:
    safe_print_err(e)
```

Race



(h) One More Dimension Selection Mechanism - Continued

In this part, we will investigate the behavior of Mondrian using the Mondrian choose dim highest var dimension choice mechanism.

Concretely, run the function Mondrian on Lab3_Data twice, and assign the results in the following variables:

• p1h_tableOut_cfs and p1h_boundaries_cfs : For the results of

```
Mondrian on Lab3_Data with qID = ['Race',
'Marital_Status'] and $k=60$ using the
Mondrian choose cut first split cut value choice mechanism.
```

- plh_actual_k_cfs : Store the resulting \$k\$ value of the table after generalization.
- p1h_tableOut_cm and p1h_boundaries_cm: For the results of Mondrian on Lab3_Data with qID = ['Race', 'Marital_Status'] and \$k=60\$ using the Mondrian choose cut median cut value choice mechanism.
 - plh_actual_k_cm : Store the resulting \$k\$ value of the table after generalization.

In both cases, use the Mondrian_choose_dim_highest_var dimension choice mechanism.

Complete the following code snippet with your code:

```
In [202... qID=['Race','Marital_Status']
    desired_k=60

# Your solution here
[p1h_tableOut_cfs, p1h_boundaries_cfs] = Mondrian(Lab3_Data, qID, desired_k, N
[p1h_actual_k_cfs, ec_report_cfs] = kAnonymity_Analyze(p1h_tableOut_cfs, qID)

[p1h_tableOut_cm, p1h_boundaries_cm] = Mondrian(Lab3_Data, qID, desired_k, Mon
[p1h_actual_k_cm, ec_report_cm] = kAnonymity_Analyze(p1h_tableOut_cm, qID)
```

```
3, 3,
3, 3,
3, 3,
7]
```

```
2, 2,
3, 3,
5, 5,
6]
```

```
5, 5,
6]
```

```
3, 3,
2, 2,
```

```
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1]
```

```
7, 7, 7, 7, 7]
1, 1, 1, 1, 1]
7, 7, 7, 7, 7]
```

```
7, 7, 7, 7, 7, 7, 7, 7]
1, 1, 1, 1, 1, 1, 1, 1, 1]
7, 7, 7, 7, 7, 7, 7, 7]
```

```
7, 7]
6, 6]
```

```
7, 7]
[2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
3, 3,
6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

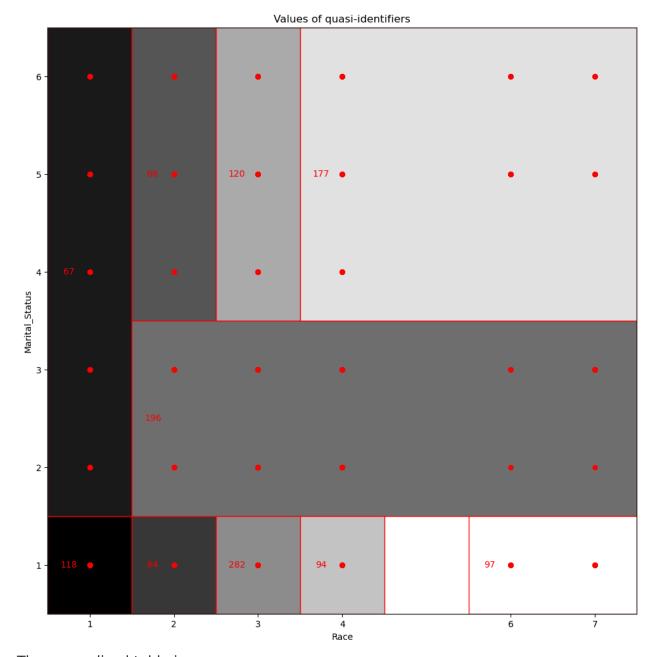
```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

Run the following cell to present the outcomes of your code.

```
_{\rm problem} = '1h'
In [203...
         try:
             qID=['Race','Marital_Status']
             desired k=60
             # Case 1
             print mk(f'The results of the run using `Mondrian choose cut first split`
             visualize Mondrian(Lab3 Data, qID, p1h boundaries cfs)
             print mk('The generalized table is:')
             display(p1h tableOut cfs)
             print mk(f'The actual $k$ in the resulting generalization was ${plh actual}
             print_mk('<br>')
             # Case 2
             print_mk(f'The results of the run using `Mondrian_choose_cut_median` with
             visualize_Mondrian(Lab3_Data, qID, p1h_boundaries_cm)
             print mk('The generalized table is:')
             display(p1h tableOut cm)
             print mk(f'The actual $k$ in the resulting generalization was ${p1h_actual
             print mk('<br>>')
         except Exception as e:
             safe_print_err(e)
```

The results of the run using Mondrian_choose_cut_first_split with qID=['Race', 'Marital Status'] and \$k=60\$ are:

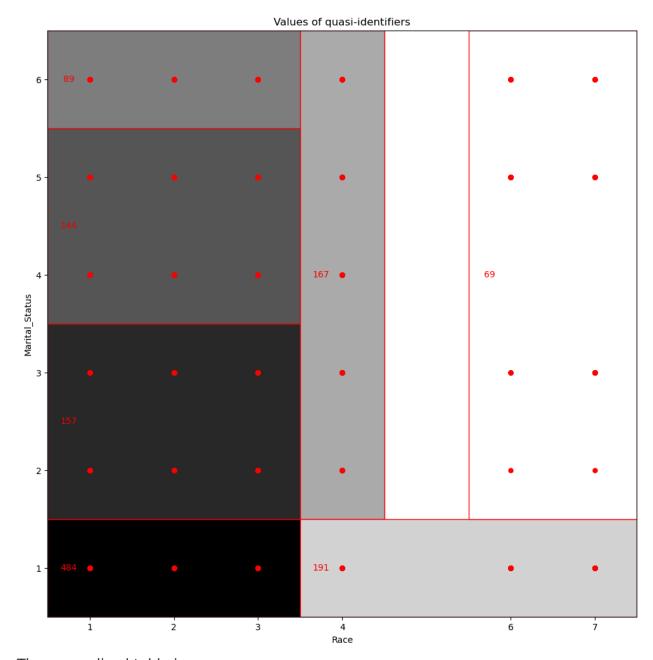


	Subject_ID	Gender	Age	Marital_Status	Country_Birth	Race	Alcohol_Aver
0	83828	2	39	(1, 1)	2	(1, 1)	
1	83854	2	46	(1, 1)	1	(1, 1)	
2	83863	1	35	(1, 1)	2	(1, 1)	
3	83987	1	68	(1, 1)	2	(1, 1)	
4	84029	2	28	(1, 1)	1	(1, 1)	
1296	93330	2	30	(1, 1)	2	(6, 7)	
1297	93370	1	30	(1, 1)	2	(6, 7)	
1298	93374	2	50	(1, 1)	2	(6, 7)	
1299	93504	1	37	(1, 1)	2	(6, 7)	
1300	93550	1	37	(1, 1)	2	(6, 7)	

1301 rows × 11 columns

The actual \$k\$ in the resulting generalization was \$66\$ (reference: requested \$k\$ was \$60\$).

The results of the run using Mondrian_choose_cut_median with $qID=['Race', 'Marital_Status']$ and k=60 are:



	Subject_ID	Gender	Age	Marital_Status	Country_Birth	Race	Alcohol_Aver
0	83754	2	67	(1, 1)	1	(1, 3)	
1	83801	2	80	(1, 1)	1	(1, 3)	
2	83828	2	39	(1, 1)	2	(1, 3)	
3	83851	2	37	(1, 1)	1	(1, 3)	
4	83854	2	46	(1, 1)	1	(1, 3)	
1296	92741	2	27	(2, 6)	1	(6, 7)	
1297	93092	1	28	(2, 6)	1	(6, 7)	
1298	93268	2	73	(2, 6)	1	(6, 7)	
1299	93373	1	44	(2, 6)	2	(6, 7)	
1300	93582	1	20	(2, 6)	2	(6, 7)	

1301 rows \times 11 columns

The actual \$k\$ in the resulting generalization was \$69\$ (reference: requested \$k\$ was \$60\$).

(i) More Than Two Dimensions

In this part, we will sanitize the data in Lab3 Data, according to quasi identifiers

qID = ['Gender', 'Age', 'Marital_Status', 'Country_Birth', 'Race'] We will not be able to visualize the resulting generalization for this part, given that there are more than \$2\$ dimensions in the set of quasi-identifiers. Therefore, we will analyze the privacy-utility tradeoff.

Write code that sanitizes Lab3_Data, with respect to quasi-identifiers qID to achieve k-Anonymity with the various values of \$k=50,55,60,65,\dots,200\$ (from \$50\$ to \$200\$ in increments of \$5\$; this is given to you in the variable pli_desired_ks). Use the dimension choice mechanism Mondrian_choose_dim_highest_distinct and the cut value choice mechanism Mondrian_choose_cut_first_split .

For each value from the list of desired \$k\$ values, record the actual \$k\$ achieved

and the Discernability Cost of the sanitization, and store them in the following variables:

- pli_actual_k_values : The list of achieved \$k\$ values, in the same order as the values in the variable pli desired ks .
- pli_discernability_costs : The list of values of Discernability Cost of the output generalizations, in the same order as the values in the variable pli desired ks.

Complete the following code snippet with your code:

```
In [204... qID = ['Gender','Age','Marital_Status','Country_Birth','Race']
    pli_desired_ks = range(50,201,5)

# Your solution here
    pli_actual_k_values = []
    pli_discernability_costs = []

for desired_ki in pli_desired_ks:
        [pli_tableOut_ra, pli_boundaries_ra_20] = Mondrian(Lab3_Data, qID, desired_lest)
        [pli_actual_k_ra, ec_report_li] = kAnonymity_Analyze(pli_tableOut_ra, qID)
        pli_actual_k_values.append(pli_actual_k_ra)
        pli_discernability_costs.append(pli_boundaries_ra_20['k'].apply(lambda_num_size)
```

```
2]
```

```
80, 80]
2, 2,
     2,
     2, 2,
3, 3,
3, 3, 3,
3, 3,
3,
     3, 3, 3, 3,
3, 3,
4, 4, 4,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5,
5, 5, 5,
 5, 5, 5, 5, 5, 5, 5, 5,
   5, 5,
    5, 5, 5, 5,
     5,
     5, 5, 5,
5, 5,
5, 5, 5, 5, 5,
  5, 5, 5, 5, 5, 5,
   5,
    5, 5, 5, 5, 5,
     5,
     5,
     5, 5,
5, 5,
5, 5,
5,
5, 5, 5, 5, 5, 5,
  5, 5, 5, 5, 5, 5,
   5, 5,
    5, 5, 5, 5, 5, 5, 5, 5,
```

```
6]
```

3, 3, 7]

```
80, 80]
22, 22, 22, 22, 22, 22, 22, 22, 22, 22]
5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
    2, 2, 2,
     2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2,
    2,
    2,
    2, 2,
2, 2,
2, 2, 2,
2,
2,
    2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2, 2,
2, 2, 2, 2, 2, 2]
48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50, 50, 50, 50,
```

```
6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
```

```
7, 7, 7, 7, 7, 7, 7, 7, 7]
48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50, 50, 50, 50,
```

```
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80]
```

```
3, 3,
```

```
2, 2,
2, 2,
3, 3,
3, 3,
```

```
80]
2, 2]
6, 6]
2, 2]
7, 7]
```

```
2,
     2, 2, 2, 2,
2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
    2, 2, 2, 2, 2,
2, 2,
2,
2,
    2, 2, 2,
     2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
   2,
   2, 2, 2, 2, 2,
    2,
    2,
     2, 2,
2, 2,
2,
2,
2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2, 2, 2,
    2,
     2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80]
2,
   2, 2, 2, 2,
2, 2,
   3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
5, 5,
5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
  5, 5,
  5, 5, 5, 5, 5, 5, 5, 5,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2,
2,
3, 3,
3,
  3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3,
  3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
30, 30, 30, 30, 30, 30]
5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7]
2, 2,
2, 2, 2, 2, 2,
```

```
80, 80]
```

```
6]
```

```
2]
2, 2,
3, 3,
3, 3,
```

```
7]
```

```
80, 80]
2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6,
6, 6, 7, 7, 7, 7, 7]
```

```
63, 63, 63, 63, 63, 63, 63, 63, 63, 64, 64, 64, 64, 64, 64, 64, 64, 64,
78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
```

```
3, 3,
```

```
3, 3,
3, 3,
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
```

```
2, 2, 2, 2]
1, 1, 1, 1, 3, 3, 3, 3, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6,
6, 6, 6, 6
2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6,
6, 6, 7, 7
```

```
80, 80, 80, 80, 80, 80, 80]
1, 1, 1,
```

```
2, 2,
2, 2,
3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
```

```
80, 80, 80, 80, 80, 80, 80]
1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7]
```

```
1, 1,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2,
2, 2, 2, 2,
2, 2,
2, 2,
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50,
```

```
2, 2,
 2, 2, 2, 2, 2,
3, 3, 3, 3, 3,
3, 3,
5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
```

```
3, 3,
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
40, 40, 40, 40, 40, 40]
1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 4, 6, 6, 7, 7, 7, 7]
2, 2, 2, 2]
```

```
6, 6, 6, 6
2, 2, 2, 2]
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3,
3,
3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3,
3,
3,
3,
  3, 3, 3, 3, 3,
  3,
   3,
   3,
3, 3,
3,
  3, 3,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
43, 43, 43, 43, 43]
1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5,
5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 6, 6, 6, 6, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80]
```

```
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3,
3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3, 3,
3,
3,
 3,
   3, 3, 3, 3, 3, 3,
    3,
    3,
    3,
3, 3,
3,
3,
```

```
80, 80, 80, 80, 80, 80, 80]
46, 46, 46, 46, 46, 46, 46]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4,
5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2, 2]
```

```
2, 2, 2, 2, 2,
3, 3, 3,
2,
3, 3,
3,
5,
5, 5,
6, 6, 6]
```

```
2, 2, 2]
7, 7, 7]
```

```
2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4,
4, 4, 5, 5, 6, 6, 6]
2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2,
 2,
 2, 2,
2, 2,
2,
2, 2, 2, 2,
80]
```

```
2, 2,
```

```
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
80]
1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 6, 6]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6]
1, 1,
1, 1,
2, 2,
2, 2, 2, 2, 2,
```

```
2, 2,
```

```
3, 3,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
54, 54, 54, 54, 54, 54, 54]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
```

```
4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2]
57, 57]
1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 5, 5,
5, 5, 5, 5, 5, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 6, 6, 6, 6, 6, 7, 7]
```

```
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80. 80. 80. 801
```

```
2, 2, 2,
2]
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
71
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
```

```
2, 2]
1, 1, 1, 1, 1, 2, 2, 2, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 6,
6, 6]
2, 2]
6, 7]
2, 2,
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
3, 3,
3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5,
2, 2, 2, 2, 2,
3, 3,
3, 3,
 3, 3, 3, 3, 3,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 6, 6, 6]
2, 2,
2, 2, 2, 2]
```

```
6, 6, 6, 6]
2, 2, 2, 2]
2, 2,
2, 2, 2, 2, 3,
7, 7, 7, 7]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2]
80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6,
6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7]
80]
2, 2, 2, 2]
6, 6, 6, 6]
2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 6, 6]
2, 2]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
6, 6]
2, 2]
7, 7]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2]
72, 72, 72]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 4, 4, 5, 5, 5, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 6, 6, 7, 7, 7, 7, 7]
```

```
78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 7, 7, 7]
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 5, 5, 6]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 6, 6, 7]
```

```
2,
2, 2, 2, 2,
2,
2, 2, 2, 2,
2]
21,
```

```
80, 80]
2, 2,
2, 2, 2,
3, 3, 3, 3,
3, 3,
3, 3, 3,
```

```
2]
2, 2,
3, 3,
3, 3,
3, 3,
```

```
7]
```

```
80, 80]
22, 22, 22, 22, 22, 22, 22, 22, 22, 22]
5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
```

```
2,
 2, 2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2]
```

```
48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50, 50, 50,
```

```
3, 3,
5, 5,
6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
```

```
3, 3, 3, 3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7]
48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50, 50, 50, 50,
```

```
72, 72, 72, 72, 72, 72, 72, 72, 72, 73, 73, 73, 73, 73, 73, 73, 73, 73,
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
```

```
80]
```

```
5, 5,
```

```
2, 2,
2, 2,
2, 2,
3, 3,
3, 3,
```

```
80]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
```

```
1, 1, 1, 1, 1,
2, 2, 2, 2, 2,
2,
  2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
2, 2,
2, 2, 2,
2,
2,
  2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 3,
```

```
2, 2,
2, 2,
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
```

```
60, 60, 60, 60, 60, 60, 60, 60, 60, 61, 61, 61, 61, 61, 61, 61, 61, 61,
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
31, 31, 31, 31, 31, 31, 31]
5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7]
2, 2, 2, 2, 2, 2, 2]
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
```

```
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7, 7]
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
34, 34, 34, 34, 34]
5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2,
3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
```

```
2, 2, 2, 2, 2, 2, 2, 2,
2]
36, 36]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 7, 7, 7, 7, 7]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
1, 1, 1,
2, 2, 2, 2,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
3, 3,
3, 3,
    3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
    3, 3, 3, 3, 3,
3,
3,
3,
 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2]
1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6,
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 7,
7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2,
2,
   2, 2, 2, 2, 2,
2,
2,
   2, 2, 2,
   2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2,
  2, 2,
  2, 2,
   2,
   2,
   2,
   2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2]
3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7]
```

```
4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1,
1, 1,
2,
2, 2, 2, 2, 2,
2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2,
2, 2,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
2, 2,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2]
45, 45]
1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6,
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 6, 6, 6, 6, 7, 7, 7]
```

```
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2,
  2, 2, 2,
  2,
  2,
   2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2,
2, 2, 2,
2, 2,
2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
```

```
80, 80, 80, 80, 80]
6, 6]
```

```
2, 2,
2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
2, 2,
2, 2,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2, 2,
2, 2, 2,
```

```
2, 2, 2, 2, 2, 2]
2, 2,
2, 2, 2,
3, 3, 3, 3, 3,
```

```
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2]
51]
4, 4, 5, 5, 5, 5, 6, 6]
2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 6, 6, 6, 7]
```

```
2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
54, 54, 54, 54, 54, 54, 54]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2]
57, 57]
1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 5, 5,
5, 5, 5, 5, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 6, 6, 6, 6, 6, 7, 7]
2]
```

```
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
2, 2, 2, 2, 2,
6]
```

```
3, 3,
3, 3,
  3, 3, 3, 3, 3,
3, 3,
7]
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
2, 2, 2, 2, 2, 2]
60, 60, 60, 60, 60]
```

```
2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3,
6, 6, 6, 7, 7, 7]
```

```
3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
2, 2,
3, 3,
3, 3, 3, 3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
63, 63, 63, 63, 63, 63]
3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 7, 7]
2, 2, 2, 2, 2, 2, 2]
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7]
2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
80]
2, 2,
5, 5, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80]
70, 70, 70, 70, 70, 70, 70, 70]
3, 3, 3, 3, 4, 4, 5, 5, 5, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 6, 6, 6]
3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2]
74]
3, 3, 4, 4, 4, 5, 6, 6]
2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 7, 7, 7, 7]
2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 5, 5, 6]
1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 6, 6, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2]
```

```
80, 80]
```

```
2, 2,
3, 3,
5, 5,
6]
```

```
2]
```

```
3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3,
   3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
3, 3,
```

```
80, 80]
```

```
22, 22, 22, 22, 22, 22, 22, 22, 22, 22]
5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
```

```
2, 2,
2,
   2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
2,
  2, 2, 2,
   2,
2,
  2,
   2, 2,
2, 2, 2, 2, 2, 2]
48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50, 50, 50, 50,
```

```
2, 2, 2, 2, 2,
1, 1,
2, 2,
3, 3,
3, 3, 3, 3, 3,
```

```
6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7]
```

```
48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50, 50, 50,
```

```
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
```

```
2, 2,
2,
    2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
    2, 2, 2, 2, 2,
2, 2,
2,
2,
    2, 2, 2,
    2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2,
  2, 2, 2, 2, 2,
   2,
    2,
    2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80]
3, 3,
3, 3, 3,
```

```
3, 3,
3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
```

```
80]
```

```
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3,
```

```
2, 2]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3,
```

```
2, 2,
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
3, 3,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
31, 31, 31, 31, 31, 31, 31]
5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7]
```

```
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2,
2, 2, 2,
2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2]
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
  3, 3, 3,
```

```
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
```

```
2, 2,
2,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7]
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
34, 34, 34, 34, 34, 34]
5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
```

```
59, 59, 59, 59, 59, 59, 59, 59, 59, 60, 60, 60, 60, 60, 60, 60, 60, 60,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
 2,
  2, 2, 2,
```

```
2, 2, 2, 2, 2,
2, 2,
3, 3,
     3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3,
3,
 3,
 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3,
     3,
3, 3,
3,
     3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3, 3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3,
3, 3,
3, 3,
3, 3,
3,
```

```
2, 2, 2, 2, 2, 2, 2, 2]
36, 36]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
2,
2,
  2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2,
2,
2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2,
2, 2, 2, 2, 2,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
[37, 37,
```

```
2, 2, 2, 2, 2, 2, 2]
40, 40, 40, 40, 40, 40, 40]
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2]
```

```
6, 6, 6, 6
2, 2, 2, 2]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3,
2, 2,
     3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3,
3,
 3,
 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3,
     3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3, 3,
     3, 3, 3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3,
3, 3,
3, 3,
7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
43, 43, 43, 43, 43]
1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5,
5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 6, 6, 6, 6, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
80, 80, 80, 80, 80, 80, 80]
1, 1, 1,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3,
   3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3, 3, 3,
3, 3, 3, 3, 3,
3,
3,
3,
3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3,
   3,
   3,
   3,
3, 3,
3,
   3, 3,
3, 3,
3, 3,
3, 3,
```

```
80, 80, 80, 80, 80, 80, 80]
46, 46, 46, 46, 46, 46, 46]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4,
5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2, 2]
```

```
1,
2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3,
3, 3,
 3, 3, 3,
3, 3,
4, 5,
```

```
6, 6, 6
2, 2, 2]
```

```
7, 7, 7]
2, 2, 2, 2, 2, 2, 2]
50, 50, 50, 50, 50, 50, 50]
1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5,
5, 5, 5, 5, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
```

```
6, 6, 6, 6, 6, 7, 7, 7]
1, 1,
2, 2, 2, 2, 2,
```

```
3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
53, 53, 53, 53, 53, 53]
```

```
4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5,
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
```

```
7, 7, 7, 7, 7, 7, 7]
80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2]
56, 56, 56]
5, 5, 5, 5, 5, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
4, 6, 6, 6, 6, 6, 6, 7, 7]
```

```
80, 80, 80, 80, 80, 80]
```

```
3, 3,
2, 2,
2, 2, 2, 2, 2,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
2, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
2, 2, 2,
2, 2, 2, 2,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2,
2, 2, 2, 2, 3,
3, 3,
3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
3, 3,
2, 2,
    2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
3, 3,
    3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3,
3, 3,
    3, 3, 3, 3, 3,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6]
```

```
2, 2,
4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
6, 6, 6, 6]
2, 2, 2, 2]
3, 3,
7, 7, 7, 7]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
2, 2,
2, 2, 2, 2,
2, 2, 2, 2, 2, 2]
80]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6,
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
2, 3,
7, 7, 7, 7, 7, 7]
80]
2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 6, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3,
```

```
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
80, 80, 80, 80, 80, 80, 80, 80, 80]
4, 4, 4, 5, 5, 5, 5, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 6, 6, 6, 7, 7, 7, 7, 7, 7]
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2]
```

```
73]
4, 4, 4, 5, 5, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
4, 4, 4, 6, 7, 7, 7, 7]
2, 2, 2, 2]
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
5, 5, 6, 6]
2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
6, 6, 7, 7
```

```
1, 1, 1, 1, 1,
2, 2,
2,
  2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2]
```

```
80, 80]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
```

```
2]
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
```

```
80, 80]
```

```
22, 22, 22, 22, 22, 22, 22, 22, 22, 22]
5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
```

```
2,
 2, 2, 2, 2,
2, 2,
2,
2,
 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2]
```

```
72, 72, 72, 72, 72, 72, 72, 72, 72, 73, 73, 73, 73, 73, 73, 73, 73, 73,
2, 2,
3,
    3, 3, 3, 3,
3, 3,
   3, 3, 3, 3, 3,
5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
  5, 5, 5, 5, 5, 5,
   5,
   5,
    5, 5,
5, 5,
5,
5, 5, 5, 5, 5, 5,
 5, 5, 5, 5, 5, 5,
  5, 5,
   5, 5, 5, 5, 5,
    5, 5, 5,
```

```
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2,
2, 2,
2, 2,
3, 3,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
  50, 50,
72, 72, 72, 72, 72, 72, 72, 72, 72, 73, 73, 73, 73, 73, 73, 73, 73, 73,
```

```
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
```

```
2,
  2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2,
2, 2, 2, 2,
2, 2, 2,
2,
2,
   2,
2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2,
  2,
2, 2, 2, 2, 2,
  2,
  2, 2,
```

```
80]
1, 1,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2, 2,
  2,
   3, 3,
3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
5, 5, 5,
5, 5,
5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
  5, 5,
  5, 5, 5, 5, 5, 5, 5,
```

```
2, 2,
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
3, 3,
3, 3,
3, 3,
3,
  3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3, 3,
```

```
80]
```

```
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
3, 3,
```

```
2, 2,
3, 3,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
31, 31, 31, 31, 31, 31, 31]
5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
2,
   2, 2, 2,
   2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2,
   2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2]
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3,
  3,
   3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
5, 5, 5, 5,
```

```
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
```

```
3, 3, 3, 3, 3,
3, 3,
    3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3,
3, 3,
 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3,
    3,
3, 3,
3,
3, 3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7]
```

```
78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
34, 34, 34, 34, 34]
5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2,
```

```
2,
   2, 2, 2, 2,
2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3,
   3, 3, 3,
2,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3, 3,
   3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
3, 3,
5, 5,
5,
```

```
2, 2,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3, 3, 3,
3,
3, 3, 3, 3, 3,
3,
  3, 3, 3, 3, 3, 3,
3,
 3,
 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3,
3, 3,
3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
```

```
2, 2, 2, 2]
37, 37, 37]
6, 6, 6, 6
2, 2, 2, 2]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7,
7, 7, 7, 7]
```

```
1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2,
    2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2,
2,
2,
    2, 2, 2,
     2, 2,
2,
 2, 2, 2,
 2, 2, 2, 2, 2, 2,
   2,
   2, 2, 2, 2, 2,
    2,
    2,
    2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2,
2,
2, 2, 2,
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50,
```

```
2, 2,
3, 3,
 3, 3, 3, 3, 3,
5, 5,
 5,
  5, 5, 5, 5,
5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
```

```
2, 2,
2, 2,
3, 3,
3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
```

```
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
40, 40, 40, 40, 40, 40]
1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 4, 6, 6, 7, 7, 7, 7]
2, 2, 2, 2]
```

```
1,
3, 3,
3, 3, 3,
```

```
6, 6, 6, 6
2, 2, 2, 2]
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2]
44, 44, 44, 44, 44, 44]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6,
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
6, 6, 6, 7, 7, 7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
2, 2, 2, 2,
```

```
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
2, 2,
3, 3,
3, 3,
```

```
62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 63, 63, 63, 63, 63, 63, 63, 63,
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
```

```
3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3,
2, 2,
2, 2, 2]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
```

```
2, 2,
3, 3,
3, 3, 3, 3, 3,
6, 6, 6
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2, 2]
2, 2,
2, 2,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
7, 7, 7]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
```

```
2, 2, 2, 2, 2]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
51, 51, 51, 51]
1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5,
5, 5, 6, 6, 6]
2, 2, 2, 2, 2]
6, 6, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
2,
 2, 2, 2,
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
3, 3,
3, 3,
```

```
67, 67, 67, 67, 67, 67, 67, 67, 67, 68, 68, 68, 68, 68, 68, 68, 68, 68,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
54, 54, 54, 54, 54, 54, 54]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80. 80. 80. 80. 801
2, 2, 2, 2, 2,
3, 3,
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2]
2, 2,
2, 2, 2, 2, 2,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2, 2, 2, 2]
58, 58, 58, 58]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 6, 6, 6, 6]
2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3,
6, 6, 7, 7, 7]
2, 2, 2, 2, 2]
```

```
80, 80]
2, 2,
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
```

```
7, 7, 7, 7, 7]
80, 80]
2, 2, 2]
61, 61]
2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6,
6, 6, 6]
2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3,
6, 7, 7]
```

```
2, 2]
80]
6, 6]
```

```
2, 2]
2, 2, 2, 2, 2,
3, 3,
7, 7]
80]
3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2]
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6,
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3,
```

```
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7,
7, 7, 7, 7, 7, 7, 7, 7]
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
67, 67, 67, 67, 67, 67, 67, 67, 67]
2, 3, 3, 3, 3, 3, 4, 5, 5, 5, 6, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 7, 7]
80, 80]
```

```
3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
80, 80]
6]
2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 6, 6, 6, 7, 7, 7, 7,
7]
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2,
  2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2, 2,
2]
```

```
80, 80]
```

```
2, 2,
5, 5,
6]
```

```
2]
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
 3, 3, 3,
7]
```

```
80, 80]
22, 22, 22, 22, 22, 22, 22, 22, 22, 22]
5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2]
```

```
48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
  50, 50,
72, 72, 72, 72, 72, 72, 72, 72, 72, 73, 73, 73, 73, 73, 73, 73, 73, 73,
```

```
6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2,
    2, 2, 2, 2, 2,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3, 3,
3,
3, 3,
3, 3,
3, 3, 3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
    3,
    3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50, 50, 50, 50,
72, 72, 72, 72, 72, 72, 72, 72, 72, 73, 73, 73, 73, 73, 73, 73, 73, 73,
```

```
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
```

```
801
2, 2,
2, 3, 3, 3, 3,
```

```
2, 2,
```

```
3, 3,
   3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3, 3,
```

```
80]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
```

```
2, 2,
2, 2, 2, 2, 2,
```

```
60, 60, 60, 60, 60, 60, 60, 60, 60, 61, 61, 61, 61, 61, 61, 61, 61, 61,
80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
3, 3,
5, 5,
```

```
2, 2,
3, 3,
3, 3,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2]
32, 32, 32, 32, 32, 32, 32]
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
6, 6, 6, 6, 6, 7, 7, 7, 7]
```

```
2,
   2, 2, 2, 2,
2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2,
   2, 2, 2, 2, 2,
2, 2,
2,
2,
   2, 2, 2,
    2,
2,
  2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2,
   2, 2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80]
2, 2, 2, 2, 2,
3, 3,
 3, 3, 3, 3, 3,
3, 3,
5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80]
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2,
  2, 2, 2, 2, 2,
2, 2,
2,
2,
  2, 2, 2,
   2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2,
3, 3,
3, 3, 3, 3, 3,
4, 5,
```

```
2, 2,
```

```
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
80, 80, 80, 80, 80, 80, 80]
```

```
2, 2, 2, 2, 2, 2, 2, 2]
39, 39, 39, 39, 39, 39, 39]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6,
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 3, 3,
    3,
    3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7]
```

```
4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
2,
   2, 2, 2,
   2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
  2,
   2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
80, 80, 80, 80, 80, 80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
2, 2,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2]
46, 46, 46, 46, 46, 46]
3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6,
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
2,
 2, 2, 2, 2, 2,
2,
 2, 2, 2,
2, 2, 2]
```

```
6, 6, 6
2, 2, 2]
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
3,
    3,
     3, 3,
3, 3,
3, 3, 3, 3, 3,
3,
  3, 3, 3, 3, 3, 3,
3,
 3,
 3, 3, 3,
   3,
   3, 3, 3, 3, 3,
    3,
     3,
     3,
3, 3,
3,
3, 3,
3, 3,
6, 6, 6,
7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2]
50, 50, 50, 50, 50, 50, 50]
1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5,
5, 5, 5, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
6, 6, 6, 6, 6, 7, 7, 7]
```

```
2, 2,
2,
3, 3,
3, 3, 3, 3, 3,
3, 3,
```

```
2, 2,
2, 2,
2, 2,
  3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
54, 54, 54, 54, 54, 54, 54, 54, 54, 54]
1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5,
5, 5, 5, 5, 5, 5, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2,
2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2, 2, 2, 2,
2, 2,
3, 3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
```

```
80, 80, 80, 80, 80]
2, 2, 2, 2, 2]
58, 58, 58, 58]
5, 6, 6, 6, 6]
2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3,
6, 6, 7, 7, 7]
2, 2, 2, 2, 2]
```

```
80, 80]
3, 3,
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
3, 3,
```

```
7, 7, 7, 7, 7]
80, 80]
2, 2, 2]
61, 61]
2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6,
6, 6, 6
2, 2, 2]
6, 7, 7
```

```
2, 2]
80]
6, 6]
2, 2]
```

```
3, 3,
7, 7]
80]
3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2]
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6,
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7,
7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2]
68, 68, 68, 68, 68, 68]
1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5,
5, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 7, 7]
2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
```

```
6, 6]
2, 2]
7, 7]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 6, 6, 7, 7, 7, 7, 7]
2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
5, 5, 6, 6]
2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
6, 6, 7, 7]
```

```
2,
  2, 2, 2, 2,
2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
2,
  2, 2, 2,
   2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2, 2,
2, 2, 2,
2]
```

```
80, 80]
```

```
3, 3,
5, 5,
6]
```

```
2]
2, 2,
3, 3,
```

```
7]
```

```
80, 80]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
2, 2,
3, 3,
5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
```

```
3, 3,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
    3, 3,
3, 3,
3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
1, 1, 1,
```

```
2, 2,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
```

```
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
30, 30, 30, 30, 30, 30, 30, 30, 30, 31, 31, 31, 31, 31, 31, 31, 31, 31,
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
30, 30, 30, 30, 30, 30, 30, 30, 30, 30]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 7, 7]
```

```
1, 1, 1, 1, 1,
2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2]
```

```
80, 80]
1, 1, 1,
2, 2, 2, 2,
```

```
2]
```

```
2, 2,
3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
7]
```

```
80, 80]
2, 2, 2, 2, 2, 2]
34, 34, 34, 34, 34]
6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2]
[6, 7, 7, 7, 7, 7]
```

```
69, 69, 69, 69, 69, 69, 69, 69, 69, 70, 70, 70, 70, 70, 70, 70, 70, 70,
```

```
2, 2, 2, 2, 2,
1, 1,
2, 2,
 2, 2, 2, 2, 2,
2, 2,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
```

```
2, 2, 2, 2]
37, 37, 37]
6, 6, 6, 6]
2, 2, 2, 2]
```

```
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7,
7, 7, 7, 7]
2, 2,
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50,
```

```
2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3,
```

```
2, 2,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
     3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
     3, 3, 3, 3, 3, 3,
       3, 3, 3,
 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3, 3, 3,
3,
 3,
 3,
     3, 3, 3, 3, 3, 3,
       3,
       3, 3,
 3, 3,
3, 3,
 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50,
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2,
2, 2, 2]
80, 80, 80, 80, 80, 80]
```

```
6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2]
2, 2,
2, 2,
3, 3,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
2, 2, 2]
45, 45]
3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6,
6, 6, 6]
2, 2, 2]
7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
2,
   2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2,
   2, 2,
2, 2,
2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
```

```
6, 6]
2, 2]
```

```
2, 2,
2, 2, 2, 2, 2,
2,
2,
3,
3, 3,
 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3,
    3,
3, 3,
3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3, 3, 3,
3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3,
7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2]
49, 49, 49, 49, 49]
5, 5, 5, 6, 6, 6]
2, 2, 2, 2, 2, 2]
6, 7, 7, 7, 7, 7]
```

```
80]
2, 2, 2,
3, 3,
3, 3, 3,
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
2, 2,
3, 3,
```

```
80]
1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4,
4, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2]
80, 80, 80, 80, 80, 80, 80, 80]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5,
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2]
57, 57, 57, 57, 57]
5, 5, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7]
```

```
2]
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
2, 2, 2,
2, 2,
6]
```

```
1, 1,
2, 2,
2,
2]
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
   3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
7]
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
```

```
80, 80, 80, 80]
2, 2, 2, 2, 2, 2]
60, 60, 60, 60, 60]
2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3,
6, 6, 6, 7, 7, 7]
```

```
3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5,
3, 3,
2, 2, 2, 2, 2,
2, 2,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
64, 64, 64, 64, 64, 64, 64, 64, 64, 64]
4, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 6, 6, 6, 6, 7, 7]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2]
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6,
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7,
7, 7, 7, 7, 7, 7, 7, 7]
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2]
68, 68, 68, 68, 68, 68]
```

```
1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5,
5, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 7, 7]
2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
6, 6]
2, 2]
7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 6, 6, 7, 7, 7, 7, 7, 7]
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
5, 5, 6, 6]
2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
6, 6, 7, 7
```

```
49, 49,
```

```
80, 80]
2, 2,
3, 3,
   3, 3, 3, 3, 3,
3, 3,
3, 3,
5, 5,
5, 5, 5,
5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
  5, 5,
  5, 5, 5, 5,
   5,
   5, 5, 5,
5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
  5, 5, 5, 5, 5, 5,
  5,
   5,
   5, 5, 5,
```

```
6]
```

```
2]
2, 2,
3, 3,
3, 3,
7]
```

```
80, 80]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1,
2, 2, 2, 2, 2,
2, 2,
   2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2,
2, 2,
2, 2, 2,
2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2, 2,
2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
3, 3,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
27, 27, 27]
```

```
1, 1, 1, 1, 1,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80]
```

```
2, 2, 2, 2, 2, 2]
31, 31, 31, 31, 31]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
6, 6, 6, 6, 6, 7]
```

```
2,
 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2,
 2, 2, 2,
 2, 2,
2, 2, 2, 2, 2]
2, 2,
2,
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
3, 3,
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2]
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7]
```

```
78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2,
    2, 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
    2, 2, 2, 2, 2,
2, 2,
2, 2,
2,
2,
    2, 2, 2,
     2, 2,
2,
 2, 2, 2,
 2, 2, 2, 2, 2, 2,
   2,
   2, 2, 2, 2, 2,
    2,
    2,
    2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2,
3, 3,
5, 5, 5, 5, 5,
```

```
2, 2,
2, 2,
3, 3,
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2]
39, 39, 39, 39, 39, 39, 39]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6,
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2]
2, 2,
2, 2,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7]
```

```
3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
3, 3,
4, 4, 4, 4, 4, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2,
 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2, 2, 2,
2, 2, 2,
2, 2,
```

```
80, 80, 80, 80, 80, 80, 80]
3, 3,
3, 3,
4, 5,
```

```
2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
```

```
80, 80, 80, 80, 80, 80, 80]
5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 2,
2, 2, 2, 2, 2,
2, 2,
2,
2,
   2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2, 2, 2,
2, 2,
2,
2, 2, 2,
2, 2, 2]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
```

```
6, 6, 6
2, 2, 2]
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
  3, 3, 3, 3, 3, 3,
   3,
    3, 3,
3, 3,
3, 3,
7, 7, 7]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
2, 2, 2, 2, 2]
```

```
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
51, 51, 51, 51]
1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5,
5, 5, 6, 6, 6]
2, 2, 2, 2, 2]
6, 6, 7, 7, 7]
```

```
2, 2,
3, 3,
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
       2, 2, 2, 2, 2,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
  2, 2, 2, 2, 2, 2, 2,
    2, 2, 2, 2, 2,
      2,
       2,
       2, 2,
2, 2,
3, 3, 3,
3, 3,
3, 3,
3,
3,
3, 3,
 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3, 3,
       3,
       3, 3,
3, 3,
3,
3, 3, 3,
3, 3,
3,
       3,
       3, 3,
3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3, 3,
       3,
3, 3,
3, 3,
3, 3, 4, 4, 4,
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
55, 55, 55, 55, 55, 55, 55, 55, 55, 55]
5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80]
2, 2,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 7]
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
1, 1,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
61, 61, 61, 61, 61, 61, 61, 61, 61, 62, 62, 62, 62, 62, 62, 62, 62, 62,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 7, 7, 7, 7]
2, 2, 2, 2, 2]
```

```
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7]
80, 80]
3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80]
3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 6, 6, 6, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3,
4, 4, 4, 4, 4, 6, 6, 6, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2]
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
5, 5, 6, 6]
2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
```

```
6, 6, 7, 7
2, 2,
```

```
2]
```

```
80, 80]
1, 1,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
  3,
  3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
5,
  5, 5, 5,
5, 5,
5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
  5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
6]
```

```
2]
2, 2,
3, 3,
3, 3,
3, 3,
```

```
6,
7,
7, 7, 7, 7,
21,
```

```
80, 80]
4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1,
2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
3, 3,
5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
```

```
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
27, 27, 27]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80]
```

```
2, 2,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
3, 3,
```

```
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80]
2]
32, 32, 32, 32, 32, 32]
6]
2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2,
2, 2,
2,
2,
   2, 2, 2,
    2, 2,
2, 2,
2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2,
  2, 2, 2, 2, 2,
   2,
   2,
    2, 2,
2, 2,
2,
2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80]
2, 2,
   2, 2, 2, 2, 2,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
3, 3,
5, 5, 5,
5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3,
3,
3,
 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80]
36, 36, 36, 36, 36, 36]
1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5,
6]
2]
4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7,
7]
```

```
1, 1,
2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2,
3,
 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
3, 3,
```

```
3, 3, 3, 3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2]
40, 40, 40, 40, 40, 40, 40]
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2,
2, 2, 2, 2]
```

```
6, 6, 6, 6]
```

```
2, 2, 2, 2]
2, 2,
2, 2, 2, 2, 2,
2, 2,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2]
44, 44, 44, 44, 44, 44]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6,
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
6, 6, 6, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1,
2, 2, 2, 2, 2,
2, 2,
2,
2,
   2, 2, 2,
    2, 2,
2,
2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2,
  2, 2,
  2, 2, 2,
   2,
   2,
   2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2,
80, 80, 80, 80, 80, 80]
```

```
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
```

```
2, 2,
2,
2,
2,
2, 2,
 2, 2, 2, 2, 2, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3,
3, 3, 3, 3,
3, 3,
3, 3,
```

```
80, 80, 80, 80, 80, 80]
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
2, 2, 2, 2,
```

```
2, 2, 2, 2]
1, 1, 1,
```

```
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
```

```
7, 7, 7, 7, 7, 7]
53, 53, 53, 53]
3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
```

```
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7]
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2]
80, 80, 80, 80, 80, 80, 80, 80]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5,
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
```

```
7, 7, 7, 7, 7, 7, 7]
80, 80, 80, 80, 80, 80, 80, 80]
2, 2]
58, 58, 58, 58, 58, 58, 58]
1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
6, 6]
2, 2]
```

```
7, 7]
2, 2,
2, 2,
2, 2, 2, 2, 2]
80, 80]
```

```
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
2, 2, 2, 2, 2,
3, 3,
3, 3,
7, 7, 7, 7, 7]
```

```
80, 80]
62, 62, 62, 62, 62, 62]
3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6,
6]
2]
7]
2, 2, 2, 2]
```

```
2, 2,
4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
6, 6, 6, 6]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2,
2, 2, 2, 2]
3, 3,
3, 3,
7, 7, 7, 7]
```

```
1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 4, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
80]
```

```
5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
80]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
```

```
3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 7, 7, 7, 7, 7]
78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 6, 6]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 7, 7, 7]
```

```
2, 2,
  2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2,
2,
  2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2, 2, 2,
2]
```

```
80, 80]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
5, 5,
6]
```

```
2]
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3,
  3,
  3,
  3,
3, 3,
  3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
7]
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
2,
2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2,
2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2,
3, 3,
5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
 5, 5, 5, 5, 5, 5, 5,
 5,
  5, 5, 5,
5, 5,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2,
2,
  2, 2, 2, 2,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3, 3,
3, 3, 3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
27, 27, 27]
```

```
2,
  2, 2, 2, 2,
2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2,
2, 2, 2, 2,
2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2,
2, 2,
  3,
  3, 3, 3, 3,
3, 3,
3, 3, 3,
5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
  5, 5, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
3, 3,
    3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3,
3, 3,
 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3, 3,
3, 3,
3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80]
32, 32, 32, 32, 32, 32]
6]
2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
```

```
7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80]
2, 2, 2,
2]
```

```
36, 36, 36, 36, 36, 36]
1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5,
6]
4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7,
7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
48, 48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
48, 48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50,
2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
```

```
7, 7, 7, 7, 7, 7, 7]
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
   3,
3, 3, 3, 3, 3,
3, 3,
5, 5, 5,
6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2]
```

```
7, 7, 7, 7, 7]
62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 63, 63, 63, 63, 63, 63, 63, 63,
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
2, 2]
```

```
46, 46, 46, 46, 46, 46]
6, 6]
2, 2]
7, 7]
2, 2, 2]
```

```
2, 2, 2, 2, 2,
3, 3, 3,
2,
3, 3,
3,
5,
5, 5,
6, 6, 6]
```

```
2, 2, 2]
7, 7, 7]
```

```
2, 2, 2, 2, 2]
51, 51, 51, 51, 51, 51, 51, 51, 51, 51]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
```

```
6, 6, 7, 7, 7]
1, 1,
1, 1,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2,
```

```
2, 2,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
55, 55, 55, 55, 55, 55, 55, 55, 55, 55]
```

```
5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 7]
```

```
2, 2,
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3, 3, 3, 3,
```

```
2,
    2, 2, 2, 2,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
```

```
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2]
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3,
3, 3, 4, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7]
```

```
80, 80]
3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
3, 3,
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
80, 80]
```

```
3, 3, 4, 4, 4, 4, 5, 5, 5, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 6, 6, 6, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2]
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
5, 5, 6, 6]
2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
6, 6, 7, 7
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
```

```
80, 80]
```

```
2, 2,
6]
```

```
2]
```

```
2, 2,
2, 2,
2,
 2, 2, 2, 2,
3, 3,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
7]
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
6, 6, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
  2, 2, 2,
   2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
  2,
  2,
   2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
```

```
61, 61, 61, 61, 61, 61, 61, 61, 61, 62, 62, 62, 62, 62, 62, 62, 62, 62,
75, 75, 75, 75, 75, 75, 75, 75, 75, 76, 76, 76, 76, 76, 76, 76, 76, 76, 77,
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
5, 5,
```

```
2, 2,
3, 3,
3, 3,
3, 3,
```

```
30, 30, 30, 30, 30, 30, 30, 30, 30, 31, 31, 31, 31, 31, 31, 31, 31, 31,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
28, 28]
```

```
2, 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2,
2, 2,
2,
2,
   2, 2, 2,
    2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2,
    2, 2,
2, 2,
2, 2, 2,
2,
2,
    2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2, 2, 2,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2,
3, 3,
3, 3, 3,
```

```
2, 2,
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 3, 3, 3,
   3,
   3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
3,
   3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2,
2, 2]
33, 33, 33, 33, 33, 33]
```

```
6, 6]
2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3,
7, 7]
```

```
63, 63, 63, 63, 63, 63, 63, 63, 63, 64, 64, 64, 64, 64, 64, 64, 64, 64,
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
```

```
2, 2,
3, 3,
3, 3,
3, 3,
3, 3,
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
2, 2, 2,
2, 2,
```

```
37, 37, 37, 37, 37, 37, 37, 37]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 4, 5, 5, 5, 5, 5, 5,
6, 6, 6
2, 2, 2]
4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7,
7, 7, 7]
```

```
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50,
```

```
2, 2,
2, 2,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50,
```

```
2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2,
2, 2,
3, 3,
  3, 3, 3, 3, 3,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 7, 7, 7, 7, 7]
2, 2,
2, 2, 2]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
6, 6, 6,
6, 6, 6
```

```
2, 2,
2, 2, 2]
2, 2,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
7, 7, 7]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
```

```
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
52, 52, 52, 52]
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
2, 2, 2, 2,
```

```
2, 2, 2, 2]
3, 3,
 3, 3, 3,
3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
6, 6, 6, 6
2, 2, 2, 2]
3, 3,
7, 7, 7, 7]
```

```
67, 67, 67, 67, 67, 67, 67, 67, 67, 68, 68, 68, 68, 68, 68, 68, 68, 68,
2, 2, 2]
57, 57, 57, 57, 57, 57, 57, 57]
1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6,
6, 6, 6
2, 2, 2]
6, 7, 7
```

```
2]
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
2, 2, 2,
2, 2,
6]
```

```
1, 1,
2, 2,
2,
2]
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
   3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
7]
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
```

```
80, 80, 80, 80]
61, 61, 61, 61]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
2, 2,
  2, 2, 2, 2, 2,
2, 2]
```

```
80]
6, 6]
2, 2]
7, 7]
```

```
80]
65]
1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2]
80]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6,
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
3, 3,
7, 7, 7, 7, 7, 7]
80]
71, 71, 71]
```

```
3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
3, 3, 4, 4, 5, 5, 6, 6]
2, 2, 2, 2, 2, 2, 2]
4, 6, 6, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
```

```
80, 80]
```

```
3, 3,
5, 5,
6]
```

```
2]
```

```
2, 2,
   2, 2, 2, 2, 2,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3, 3, 3,
3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3,
   3, 3, 3, 3,
3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
7]
```

```
80, 80]
```

```
2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
6, 6, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2,
2,
   2, 2, 2,
   2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2,
   2, 2,
2, 2,
2, 2, 2,
2,
2,
   2, 2,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2,
 2, 2, 2,
3, 3, 3, 3, 3,
```

```
5, 5,
2, 2,
```

```
2, 2,
3, 3,
3, 3,
3, 3,
3, 3,
```

```
30, 30, 30, 30, 30, 30, 30, 30, 30, 31, 31, 31, 31, 31, 31, 31, 31, 31,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
28, 28]
```

```
2,
    2, 2, 2, 2,
2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
    2, 2, 2, 2, 2,
2, 2,
2,
2,
    2, 2, 2,
    2,
     2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2,
   2, 2, 2, 2, 2,
    2,
    2,
    2, 2,
2, 2,
2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2, 2,
    2,
    2, 2,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3,
5, 5,
5,
```

```
2, 2,
    3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2,
2, 2]
33, 33, 33, 33, 33, 33]
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
5, 5,
6, 6]
```

```
2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3,
7, 7]
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
```

```
2, 2,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
2, 2,
2]
37, 37, 37, 37, 37, 37, 37, 37]
```

```
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 4, 5, 5, 5, 5, 5, 5,
6, 6, 6
2, 2, 2]
4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7,
7, 7, 7]
```

```
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50,
```

```
2, 2,
```

```
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2,
2, 2,
2,
3,
       3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
     3, 3, 3, 3, 3, 3,
       3,
       3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3, 3, 3,
       3, 3, 3, 3,
3, 3,
3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50,
```

```
2, 2, 2, 2, 2, 2]
42, 42, 42, 42, 42, 42, 42, 42, 42, 42]
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1,
1, 1,
2,
2, 2, 2, 2, 2,
2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2,
2, 2,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
2, 2,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 7, 7, 7, 7, 7]
2, 2,
2, 2,
2, 2, 2]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
2, 2,
3, 3,
6, 6, 6
```

```
2, 2, 2]
2, 2,
2, 2,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
7, 7, 7]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
```

```
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
52, 52, 52, 52]
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
2, 2, 2, 2]
```

```
2, 2, 2, 2, 2,
3, 3, 3,
3,
3, 3,
3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5,
6, 6, 6, 6]
```

```
2, 2, 2, 2]
3, 3,
3, 3,
3, 3,
7, 7, 7, 7]
```

```
2, 2, 2]
57, 57, 57, 57, 57, 57, 57, 57]
1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6,
6, 6, 6
2, 2, 2]
6, 7, 7]
```

```
2]
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
6]
```

```
2]
2, 2, 2, 2, 2,
3,
3, 3,
3, 3,
3, 3,
7]
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
```

```
61, 61, 61, 61]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
2, 2, 2, 2, 2,
2, 2]
80]
```

```
6, 6]
2, 2]
2, 2, 2, 2, 2,
3, 3,
7, 7]
```

```
80]
65]
1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2]
80]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6,
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7]
80]
71, 71, 71]
3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2]
3, 3, 4, 4, 5, 5, 6, 6]
2, 2, 2, 2, 2, 2, 2]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
4, 6, 6, 7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
  2, 2, 2, 2,
2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2,
2]
```

```
80, 80]
```

```
5, 5,
6]
```

```
2]
2, 2,
```

```
3, 3,
    3, 3, 3, 3, 3,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
     3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
     3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
7]
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
6, 6, 7, 7, 7, 7, 7, 7
```

```
2,
  2, 2, 2, 2,
2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
2,
  2, 2, 2,
  2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2,
  2, 2, 2,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
```

```
2, 2,
```

```
2, 2,
2, 2,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
29, 29, 29, 29, 29, 29]
```

```
2, 2,
2,
    2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2, 2,
   2,
    2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2,
   2, 2, 2, 2,
2, 2,
2,
2,
   2, 2, 2,
    2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2,
  2, 2, 2, 2, 2,
   2,
   2,
    2, 2,
2, 2,
2,
2,
```

```
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
1, 1,
2, 2,
3, 3,
3,
  3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
5,
  5,
  5, 5,
```

```
2, 2,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3,
      3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3, 3,
3, 3,
      3,
      3, 3, 3, 3,
3, 3, 3, 3, 3,
 3,
3,
 3,
 3, 3, 3,
  3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3,
     3,
      3,
      3,
      3,
3, 3,
3,
      3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3, 3,
      3,
      3, 3, 3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3, 3,
      3,
      3, 3,
3, 3,
3, 3,
```

```
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
2, 2,
2, 2]
34, 34, 34, 34, 34, 34, 34]
6, 6]
2, 2]
7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
```

```
69, 69, 69, 69, 69, 69, 69, 69, 69, 70, 70, 70, 70, 70, 70, 70, 70, 70,
```

```
2, 2,
2, 2,
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3,
     3,
     3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3, 3,
     3,
     3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3, 3, 3,
     3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
1, 1,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2]
```

```
2, 2, 2, 2, 2,
3, 3, 3, 3, 3,
3, 3,
3, 3,
5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2]
3, 3,
```

```
7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7, 7, 7, 7]
```

```
62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 63, 63, 63, 63, 63, 63, 63, 63,
80, 80, 80, 80, 80, 80]
```

```
2, 2,
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 63, 63, 63, 63, 63, 63, 63, 63,
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
```

```
2]
49, 49, 49, 49, 49, 49]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6,
6]
2]
7]
2, 2,
```

```
80]
1, 1,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3,
 3,
 3, 3, 3,
3, 3,
3, 3,
```

```
2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
  3, 3, 3, 3,
3, 3,
3, 3,
  3, 3, 3, 3, 3,
```

```
80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
1, 1, 1,
2, 2, 2, 2, 2,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5,
3, 3,
```

```
2, 2, 2, 2, 2,
2, 2,
3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
  3,
  3, 3,
3, 3,
3, 3,
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
64]
5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
```

```
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2]
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6,
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
```

```
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7,
7, 7, 7, 7, 7, 7, 7, 7]
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 6, 6, 6]
3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2,
  2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2, 2,
2]
```

```
80, 80]
```

```
5, 5,
6]
```

```
2]
2, 2,
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3,
 3, 3, 3, 3, 3, 3,
 3,
  3, 3,
3, 3,
3, 3,
3, 3,
7]
```

```
80, 80]
2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
6, 6, 7, 7, 7, 7, 7, 7]
```

```
2,
  2, 2, 2, 2,
2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
1, 1,
   2, 2, 2, 2, 2,
2, 2,
3, 3,
3, 3, 3, 3, 3,
3, 3,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
  5, 5, 5, 5, 5, 5,
5, 5,
   5,
   5,
   5, 5,
5, 5,
5,
```

```
2, 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 3, 3, 3, 3,
    3,
    3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3,
    3, 3, 3, 3,
3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3,
3, 3,
```

```
49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50, 50, 50, 50,
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
2,
 2, 2, 2,
```

```
29, 29, 29, 29, 29, 29]
```

```
2,
 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2,
2, 2, 2, 2,
2,
2,
```

```
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
2, 2,
  2, 2, 2, 2, 2,
3, 3,
  3, 3, 3, 3, 3,
5, 5, 5, 5, 5,
```

```
2, 2,
2, 2,
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3, 3, 3,
3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3, 3,
3,
3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3, 3,
3, 3,
```

```
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
```

```
80, 80, 80, 80, 80, 80, 80]
1, 1, 1,
2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
```

```
2, 2, 2, 2, 2,
2, 2,
3, 3,
     3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3, 3, 3, 3,
3,
3,
 3,
 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3,
     3,
     3,
     3,
     3,
3, 3,
3,
     3, 3, 3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3, 3, 3,
3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3,
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
1, 1,
2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2,
2, 2, 2, 2]
48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50, 50, 50, 50,
```

```
2, 2,
2, 2, 2, 2, 2,
3, 3,
6, 6, 6, 6]
```

```
2, 2, 2, 2]
3, 3,
3, 3,
7, 7, 7, 7]
```

```
2, 2, 2, 2,
2]
```

```
6]
2]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7,
7]
2, 2, 2]
```

```
2, 2, 2, 2, 2,
3, 3, 3,
2,
3, 3,
3,
5,
5, 5,
6, 6, 6]
```

```
2, 2, 2]
7, 7, 7]
```

```
52, 52, 52, 52, 52, 52, 52, 52, 52, 52]
3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
```

```
3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
1, 1,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2,
2, 2]
```

```
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6]
2, 2, 2, 2]
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
7, 7, 7, 7]
58, 58, 58, 58, 58, 58, 58, 58, 58]
```

```
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2]
80, 80]
```

```
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
7, 7, 7, 7, 7]
```

```
80, 80]
63, 63, 63, 63, 63, 63, 63, 63]
4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2]
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
3, 3,
7, 7, 7, 7, 7, 7]
```

```
69, 69, 69, 69, 69, 69]
1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
4, 4, 4, 5, 5, 5, 5, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 6, 6, 6, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2,
  2, 2, 2,
   2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2, 2, 2,
2,
2,
  2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2, 2,
2]
```

```
80, 80]
```

```
6]
```

```
2]
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
7]
```

```
80, 80]
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
25, 25]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
2,
  2, 2, 2, 2,
2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2,
  2, 2, 2,
  2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
26, 26, 26, 26, 26, 26, 26, 26, 27, 27, 27, 27, 27, 27, 27,
```

```
801
1, 1,
 2, 2, 2,
2, 2,
3, 3, 3, 3, 3,
3, 3,
5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
 5, 5, 5, 5, 5, 5,
 5,
 5,
 5, 5,
```

```
2, 2, 2, 2, 2,
2, 2,
2,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3,
3, 3,
3,
3, 3,
3, 3,
3, 3, 3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3, 3,
     3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
```

```
80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
```

```
2]
```

```
80, 80]
2, 2,
```

```
2]
3, 3,
```

```
3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
7]
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 3, 3, 3, 3, 3, 4, 5, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
 2,
 2,
 2, 2,
2, 2,
2, 2, 2,
2,
2,
 2, 2, 2, 2,
2,
2, 2, 2,
2, 2,
2, 2, 2,
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2,
3, 3,
3, 3, 3, 3, 3,
3, 3,
```

```
2, 2,
2, 2,
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
2,
  2, 2, 2,
  2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2, 2,
2,
2, 2,
2, 2]
```

```
6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2]
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
7, 7, 7, 7]
```

```
2]
6]
2]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7,
7]
2, 2,
2, 2, 2]
```

```
2, 2,
2, 2,
3, 3, 3, 3, 3,
3, 3,
3, 3,
6, 6, 6
```

```
2, 2, 2]
2, 2,
2, 2, 2, 3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
3, 3,
7, 7, 7]
```

```
52, 52, 52, 52, 52, 52, 52, 52, 52]
3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2]
```

```
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6
2, 2, 2, 2]
```

```
7, 7, 7, 7]
2, 2, 2, 2, 2,
58, 58, 58, 58, 58, 58, 58, 58, 58]
```

```
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
2, 2, 2, 2, 2]
80, 80]
```

```
6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2]
3, 3,
3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7]
```

```
80, 80]
63, 63, 63, 63, 63, 63, 63, 63]
4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2]
```

```
2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3,
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2]
3, 3,
   3, 3, 3,
7, 7, 7, 7, 7, 7, 7]
```

```
69, 69, 69, 69, 69]
1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
80, 80, 80, 80, 80, 80, 80, 80, 80]
4, 4, 4, 5, 5, 5, 5, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 6, 6, 6, 7, 7, 7, 7, 7, 7]
```

```
2,
   2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2,
   2, 2, 2, 2, 2,
2,
   2, 2, 2,
    2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
   2,
   2,
   2, 2,
2]
```

```
80, 80]
2, 2,
 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3,
```

```
5, 5,
6]
```

```
2]
3, 3,
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
25, 25]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
2, 2,
```

```
80]
```

```
2, 2,
5, 5,
```

```
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3, 3, 3,
3, 3, 3, 3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3,
3,
3,
  3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
   3, 3, 3, 3,
3, 3,
3, 3,
7, 7,
```

```
80]
2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2]
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
1, 1, 1,
```

```
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2]
```

```
6,
7,
 7, 7, 7, 7,
7, 7, 7, 7, 7, 7, 7]
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
```

```
80, 80, 80]
36, 36, 36, 36, 36, 36, 36]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2,
    2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2, 2,
    2, 2, 2, 2, 2,
2,
    2, 2, 2, 2, 2,
2,
2,
    2, 2, 2,
     2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
    2,
    2,
    2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
48, 48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50,
50, 50,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
48, 48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50,
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5,
6]
2]
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7,
7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3,
3, 3,
3, 3, 3,
3, 3,
    3, 3, 3, 4, 4,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 63, 63, 63, 63, 63, 63, 63, 63,
```

```
80, 80, 80, 80, 80, 80]
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
7, 7, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
7, 7, 7, 7, 7, 7]
67, 67, 67, 67, 67, 67, 67, 67, 67, 68, 68, 68, 68, 68, 68, 68, 68, 68,
54, 54, 54, 54, 54, 54, 54]
4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
```

```
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7]
2, 2,
2, 2, 2,
2, 2,
2, 2,
2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
```

```
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
60, 60, 60, 60, 60, 60]
1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
3, 3,
4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
2, 2,
```

```
3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
2, 2,
3, 3,
3, 3, 3,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2]
```

```
5, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
80]
```

```
5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2,
  2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
  2, 2, 2,
  2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2,
2, 2, 2,
2]
```

```
80, 80]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
2, 2,
5, 5,
6]
```

```
2]
2, 2,
3, 3,
3, 3,
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2]
25, 25]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
80]
2, 2,
2, 2, 2, 2, 2,
3, 3,
5, 5, 5, 5, 5,
5, 5, 5,
5, 5,
```

```
3, 3,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80]
2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2]
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
```

```
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
7, 7, 7, 7, 7, 7, 7, 7]
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
36, 36, 36, 36, 36, 36]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
2,
   2, 2, 2,
    2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2,
   2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2, 2,
   2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2,
3, 3,
 3, 3, 3, 3, 3,
5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
```

```
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5,
6]
2]
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7,
7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
80, 80, 80, 80, 80, 80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2, 2, 3, 3,
3, 3,
3, 3,
     3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
     3, 3, 3,
3, 3, 3, 3, 3,
3,
3,
 3,
 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3,
     3,
     3,
     3,
3, 3,
3,
     3, 3,
3, 3,
3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2]
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
2, 2, 2,
2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
7, 7, 7, 7, 7, 7]
54, 54, 54, 54, 54, 54, 54]
```

```
4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7]
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
3, 3,
3, 3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
60, 60, 60, 60, 60, 60]
```

```
1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
    2, 2, 2, 2, 2,
3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
2, 2,
2, 2,
    2, 2, 2, 2, 2,
3, 3,
3, 3, 3, 3, 3,
3, 3,
    3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2]
5, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
80]
```

```
5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2,
  2, 2, 2,
   2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2, 2, 2,
2,
2,
  2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2, 2,
2]
```

```
80, 80]
```

```
6]
```

```
2]
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
7]
```

```
80, 80]
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
25, 25]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
2,
  2, 2, 2, 2,
2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2,
  2, 2, 2,
  2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
26, 26, 26, 26, 26, 26, 26, 26, 27, 27, 27, 27, 27, 27, 27,
```

```
801
1, 1,
 2, 2, 2,
2, 2,
3, 3, 3, 3, 3,
3, 3,
5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
 5, 5, 5, 5, 5, 5,
 5,
 5,
 5, 5,
```

```
2, 2, 2, 2, 2,
2, 2,
2,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3,
3, 3,
3,
3, 3,
3, 3,
3, 3, 3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3, 3,
     3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
```

```
80]
2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 7, 7, 7]
```

```
2,
 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2,
 2, 2, 2,
 2, 2,
2, 2, 2, 2, 2]
2, 2,
2,
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
3, 3,
6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2]
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7]
```

```
78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
37, 37, 37, 37, 37, 37, 37, 37]
1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5,
2, 2,
3, 3,
6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
1, 1,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2,
2, 2, 2, 2,
2, 2,
2, 2,
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50,
```

```
2, 2,
 2, 2, 2, 2, 2,
3, 3, 3, 3, 3,
3, 3,
5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
```

```
3, 3,
```

```
4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
```

```
2, 2,
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3, 3,
     3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3, 3,
     3,
      3, 3,
3,
3, 3, 3, 3, 3,
 3,
  3, 3, 3, 3, 3, 3,
3,
 3,
 3, 3,
  3,
    3,
    3, 3, 3,
     3,
     3,
     3,
      3,
      3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3,
3, 3,
62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 63, 63, 63, 63, 63, 63, 63, 63,
```

```
80, 80, 80, 80, 80, 80]
2, 2]
4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6,
6, 6]
2, 2]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7,
7, 7]
```

```
2, 2, 2, 2, 2,
3, 3, 3,
```

```
2, 2,
3, 3,
3, 3,
```

```
57, 57, 57, 57, 57, 57, 57, 57, 57, 57]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
```

```
6]
2]
3, 3,
```

```
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
63, 63, 63, 63, 63, 63, 63, 63, 63, 63]
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2]
2, 2,
3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
3, 3,
7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2]
70]
5, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 6, 6, 6]
3, 3,
3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
  2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2, 2,
2, 2, 2,
2]
```

```
80, 80]
```

```
6]
```

```
2]
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
7]
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
25, 25]
5, 5, 5, 5, 5,
5, 5, 5, 5,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
2,
2, 2, 2, 2,
2, 2,
2,
2,
2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
27,
```

```
80]
2, 2,
  2, 3, 3, 3, 3,
3, 3,
3, 3, 3,
3, 3,
5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
  5, 5, 5, 5, 5, 5,
  5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
 5, 5, 5, 5, 5, 5, 5,
  5, 5, 5, 5,
```

```
2, 2,
   2, 2, 2, 2, 2,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
```

```
80]
2, 2, 2, 2,
2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 7, 7, 7]
```

```
2,
   2, 2, 2, 2,
2, 2,
2,
2,
   2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2,
2,
2, 2, 2, 2, 2, 2, 2]
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
```

```
80, 80, 80]
6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2]
3, 3,
```

```
7, 7, 7, 7, 7, 7, 7]
```

```
78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
37, 37, 37, 37, 37, 37, 37, 37]
1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
1, 1, 1, 1, 1,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2,
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50,
```

```
2, 2,
2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
2, 2,
 3, 3, 3, 3, 3,
3, 3,
```

```
2, 2,
2, 2,
3, 3,
```

```
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50,
```

```
44, 44, 44, 44, 44, 44, 44, 44, 44, 44]
4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
```

```
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3, 3, 3,
3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
3,
 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3,
80, 80, 80, 80, 80, 80]
```

```
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7]
2, 2,
```

```
2, 2, 2,
2, 2,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5,
 5, 5, 5, 5, 5, 5, 5,
  5, 5, 5, 5,
```

```
2, 2,
2, 2,
3, 3,
3, 3, 3,
 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
```

```
67, 67, 67, 67, 67, 67, 67, 67, 67, 68, 68, 68, 68, 68, 68, 68, 68, 68,
2, 2,
58, 58, 58, 58, 58, 58, 58, 58, 58, 58]
1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2]
80, 80]
2, 2, 2, 2, 2,
2, 2,
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5,
6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2]
3, 3,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7]
80, 80]
```

```
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 7, 7, 7, 7]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6,
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7,
7, 7, 7, 7, 7, 7, 7, 7]
```

```
2,
 2, 2, 2,
2, 2,
2, 2,
2,
 2, 2, 2, 2,
2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2,
```

```
80, 80]
```

```
6]
```

```
2]
2, 2,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
7]
```

```
80, 80]
1, 1, 1,
2,
2, 2, 2, 2, 2, 2, 2, 2]
```

```
25, 25]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2,
2,
    2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
    2, 2, 2, 2, 2,
2, 2,
2,
2,
    2, 2, 2,
    2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2,
  2, 2, 2, 2, 2,
   2,
    2,
    2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80]
3, 3,
3, 3, 3,
```

```
3, 3,
3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
```

```
80]
```

```
2, 2, 2]
6, 6, 6
2, 2, 2]
7, 7, 7]
```

```
1, 1, 1,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
    2, 2, 2, 2, 2,
2, 2,
2,
2,
    2, 2, 2,
    2, 2,
2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2,
  2, 2,
   2, 2, 2,
   2,
    2,
    2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2,
2,
2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80]
1, 1,
3,
  3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
5,
  5, 5,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3,
     3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
3,
     3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3,
3,
 3,
 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3,
    3,
     3,
     3,
     3,
3, 3,
3,
     3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
38, 38, 38]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2,
2, 2, 2, 2, 2,
```

```
2, 2, 2,
2, 2,
3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3,
3, 3,
 3, 3, 3,
```

```
2]
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
7]
```

```
4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2,
   2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2, 2, 2,
2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
```

```
6, 6]
2, 2]
```

```
2, 2, 2, 2, 2,
2,
2, 2,
2,
    2, 2, 2, 2, 2,
3, 3,
3, 3, 3,
3, 3, 3, 3, 3,
3,
3,
 3,
 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3,
    3,
    3,
     3,
3, 3,
3,
3, 3,
3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3, 3,
3, 3,
7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2, 2, 2]
2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2,
3, 3, 3, 3,
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
6, 6, 6, 6]
2, 2, 2, 2]
7, 7, 7, 7]
```

```
2, 2,
2, 2, 2]
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
```

```
6, 6, 7, 7, 7]
61, 61, 61, 61, 61, 61, 61, 61, 61, 62, 62, 62, 62, 62, 62, 62, 62, 62,
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
2, 2, 2, 2,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3, 3,
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
65, 65, 65, 65, 65, 65, 65, 65, 65]
5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
2, 2, 2, 2]
80]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6,
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2,
  2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
2, 2, 2,
2]
```

```
80, 80]
```

```
2, 2,
5, 5,
6]
```

```
2]
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
 3, 3, 3,
7]
```

```
80, 80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
2,
  2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2,
  2,
  2,
  2,
  2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
1, 1,
3, 3, 3, 3, 3, 3,
2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
   3,
   3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
   5,
   5, 5, 5,
5, 5,
5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
  5, 5,
   5, 5, 5, 5, 5, 5, 5, 5,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2,
3, 3,
3,
  3, 3, 3, 3,
3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3,
  3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
1, 1, 1, 1,
30, 30, 30, 30, 30, 30, 30, 30, 30, 31, 31, 31, 31, 31, 31, 31, 31, 31,
```

```
33, 33, 33, 33, 33, 33, 33, 33]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80]
```

```
3, 3,
```

```
2, 2,
2, 2,
```

```
63, 63, 63, 63, 63, 63, 63, 63, 63, 64, 64, 64, 64, 64, 64, 64, 64, 64,
78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80, 80,
```

```
80, 80, 80]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
1, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2,
2,
2,
   2, 2, 2,
   2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2,
   2, 2,
2,
2, 2, 2,
2,
2, 2, 2, 2]
```

```
6, 6, 6, 6]
```

```
2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7]
```

```
2]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5,
6]
```

```
2]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7,
7]
2, 2, 2]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
2, 2,
 3, 3, 3, 3,
3, 3,
6, 6, 6
```

```
2, 2, 2]
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
3, 3,
7, 7, 7]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
```

```
1, 1,
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
54, 54, 54, 54, 54, 54, 54, 54, 54, 54]
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
60, 60, 60, 60, 60, 60]
1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
2, 2,
3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5,
 5, 5, 5, 5, 5,
```

```
2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2]
```

```
3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6,
6, 6]
2, 2]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7,
7, 7]
80, 80]
3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
```

```
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2,
 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2, 2,
  2, 2,
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
 3, 3, 3,
```

```
5, 5,
6]
```

```
2]
2, 2,
```

```
7]
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
3, 3,
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5,
2, 2, 2, 2, 2]
```

```
62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 63, 63, 63, 63, 63, 63, 63, 63,
80, 80, 80, 80, 80, 80]
```

```
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3,
   3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3, 3, 3,
3, 3, 3, 3, 3,
3,
3,
3,
3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3,
   3,
   3,
   3,
3, 3,
3,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5,
4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2,
 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2,
```

```
55, 55, 55, 55, 55, 55, 55, 55, 55, 55]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80]
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
80]
2,
2, 2, 2, 2, 2, 2, 2]
62, 62, 62, 62, 62]
```

```
5, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 7, 7, 7]
2, 2, 2, 2]
```

```
6, 6, 6, 6]
2, 2, 2, 2]
2, 2,
2, 2, 2, 2, 3,
7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2]
70, 70, 70, 70]
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7, 7]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 5, 5, 5, 6, 6, 6]
```

```
3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 7, 7, 7, 7]
2, 2,
```

```
2,
   2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2, 2,
  2,
   2, 2, 2,
2, 2, 2, 2,
2, 2, 2,
   2,
2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
  2,
  2,
  2,
   2, 2,
2]
```

```
80, 80]
2, 2, 2,
```

```
5, 5,
6]
```

```
2]
```

```
7]
```

```
80, 80]
2, 2,
5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3,
3,
3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2,
2, 2,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5,
2, 2, 2, 2, 2]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
```

```
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3, 3, 3, 3, 3,
3,
3,
3,
 3, 3, 3,
 3, 3, 3, 3, 3, 3,
  3,
   3, 3, 3, 3, 3,
    3,
    3,
    3,
3, 3,
3,
    3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3,
    3, 3, 3,
3, 3,
7, 7, 7, 7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
2,
2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
```

```
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
7, 7, 7, 7, 7, 7]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2]
```

```
56, 56, 56, 56, 56, 56]
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 7, 7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
1, 1,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2, 2,
    2, 2, 2, 2, 2,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
2, 2,
2, 2,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3, 3,
3, 3,
    3,
    3, 3,
3, 3,
62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 63, 63, 63, 63, 63, 63, 63, 63,
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
```

```
1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2]
```

```
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7]
```

```
2, 2,
2,
  2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2, 2, 2,
2, 2,
2,
  2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2,
2]
```

```
80, 80]
```

```
5, 5,
6]
```

```
2]
3, 3,
```

```
3, 3,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
  3,
   3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
7]
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
2,
  2, 2, 2,
   2, 2,
2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
  2,
  2,
   2, 2,
2, 2,
2,
2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2, 2, 2,
2, 2,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3,
3, 3,
 3, 3, 3, 3, 3,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
30, 30, 30, 30, 30, 30, 30, 30, 30, 31, 31, 31, 31, 31, 31, 31, 31, 31,
```

```
2, 2, 2,
```

```
1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2,
2,
  2, 2, 2,
  2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
```

```
2, 2,
3, 3,
5,
5, 5, 5, 5,
```

```
2, 2,
2, 2,
3, 3,
3, 3,
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
1, 1, 1,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2,
2,
2,
    2, 2, 2,
     2, 2,
2, 2,
 2, 2, 2,
 2, 2, 2, 2, 2, 2,
   2,
   2, 2, 2, 2, 2,
    2,
    2,
    2, 2,
2, 2,
2, 2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2, 2,
    2, 2, 2,
2,
2,
2, 2, 2, 2, 2]
62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 63, 63, 63, 63, 63, 63, 63, 63,
```

```
80, 80, 80, 80, 80, 80]
6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2]
49, 49, 49, 49, 49, 49]
6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2]
6, 7, 7, 7, 7, 7, 7, 7]
2, 2,
2, 2, 2, 2, 2,
```

```
80]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
3, 3,
3, 3,
```

```
2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
```

```
80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2]
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
2, 2, 2,
2, 2,
6]
```

```
1, 1,
2, 2,
2,
2]
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
   3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
7]
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
```

```
80, 80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6,
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7,
7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2]
```

```
80, 80]
```

```
2, 2,
3, 3,
5, 5,
6]
```

```
2]
```

```
3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3,
   3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
3, 3,
```

```
80, 80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2,
 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2,
2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2,
   2, 2,
2, 2,
2, 2, 2,
2,
2,
   2, 2, 2, 2,
2, 2,
2, 2, 2,
2, 2,
2,
2, 2, 2, 2,
48, 48, 48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50,
```

```
1, 1,
3,
  3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
5,
  5, 5,
```

```
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
  3, 3, 3,
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
 2, 2, 2, 2, 2,
2, 2, 2, 2, 2]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
2,
3, 3,
   3, 3, 3,
3, 3,
5, 5, 5, 5,
6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2]
7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2]
```

```
49, 49, 49, 49, 49, 49]
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
6, 7, 7, 7, 7, 7, 7, 7]
```

```
80]
1, 1,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3,
 3,
 3, 3, 3,
3, 3,
3, 3,
```

```
2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
  3, 3, 3, 3,
3, 3,
3, 3,
  3, 3, 3, 3, 3,
```

```
80]
2, 2,
1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2,
```

```
2]
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
```

```
6]
2]
3, 3,
7]
```

```
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2]
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6,
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7,
7, 7, 7, 7, 7, 7, 7, 7]
```

```
2]
```

```
80, 80]
2, 2,
     2,
     2, 2,
3, 3,
3, 3, 3,
3, 3,
3,
     3, 3, 3, 3,
3, 3,
4, 4, 4,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5,
5, 5, 5,
 5, 5, 5, 5, 5, 5, 5, 5,
   5, 5,
    5, 5, 5, 5,
     5,
     5, 5, 5,
5, 5,
5, 5, 5, 5, 5,
  5, 5, 5, 5, 5, 5,
   5,
    5, 5, 5, 5, 5,
     5,
     5,
     5, 5,
5, 5,
5, 5,
5,
5, 5, 5, 5, 5, 5,
  5, 5, 5, 5, 5, 5,
   5, 5,
    5, 5, 5, 5, 5, 5, 5, 5,
```

```
6]
```

3, 3, 7]

```
80, 80]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
1, 1, 1,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
2, 2,
6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
```

```
5, 5,
2, 2,
```

```
2, 2, 2, 2, 2,
2, 2,
2,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3, 3,
3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3,
3, 3,
3, 3,
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2]
```

```
62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 63, 63, 63, 63, 63, 63, 63, 63,
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80, 80]
1,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  3, 3, 3, 3, 3, 3,
   3,
    3, 3,
3, 3,
   3, 3, 3,
3, 3,
```

```
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
```

```
7, 7, 7, 7, 7]
80, 80, 80, 80, 80, 80]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
2, 2,
```

```
2, 2,
```

```
2, 2,
3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7]
2, 2,
2, 2, 2, 2,
2, 2, 2, 2, 2]
80, 80]
```

```
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
7, 7, 7, 7, 7]
```

```
80, 80]
65, 65]
1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2]
80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6,
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7]
```

```
2]
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
2, 2,
  3, 3, 3, 3, 3,
3, 3,
5, 5, 5,
5, 5, 5, 5,
5, 5,
5,
  5, 5, 5,
5, 5,
```

```
6]
```

```
2]
7]
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
```

```
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80]
35]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4,
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
```

```
80, 80, 80, 80, 80, 80, 80]
2,
2,
2, 2, 2,
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
```

```
3, 3,
    3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3,
3,
 3,
 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3,
    3,
    3,
    3,
3, 3,
3,
    3, 3, 3, 3,
3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3, 3,
3, 3,
```

```
80, 80, 80, 80, 80, 80, 80]
43]
2, 2, 2,
```

```
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2,
3, 3,
3, 3, 3, 3, 3,
3, 3,
```

```
2, 2,
3, 3,
3, 3,
```

```
80, 80, 80, 80, 80, 80, 80]
```

```
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2,
3, 3,
4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2,
3, 3,
3, 3,
```

```
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2,
   2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2,
   2, 2,
2, 2,
3,
3, 3,
3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3,
3, 3, 3,
3, 3,
3,
3, 3,
```

```
59, 59]
2, 2,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2,
3, 3,
  3, 3, 3, 3, 3,
3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
2, 2, 2, 2]
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2,
2, 2, 2, 2, 2, 2]
67, 67, 67, 67, 67, 67, 67, 67, 67]
```

```
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7]
80, 80]
3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
```

```
4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2,
```

```
2,
2, 2, 2, 2,
2,
2, 2, 2, 2,
2]
21,
```

```
80, 80]
2, 2,
2, 2, 2,
3, 3, 3, 3,
3, 3,
3, 3, 3,
```

```
2]
2, 2,
3, 3,
3, 3,
3, 3,
```

```
7]
```

```
80, 80]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80]
```

```
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2,
35]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4,
```

```
2, 2, 2, 2, 2,
2, 2,
```

```
80, 80, 80, 80, 80, 80, 80]
```

```
2, 2,
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2,
   2, 3,
3, 3,
3, 3,
3, 3,
3, 3,
3,
   3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
```

```
80, 80, 80, 80, 80, 80, 80]
43]
```

```
2, 2,
2, 2, 2, 2, 2,
```

```
80, 80, 80, 80, 80, 80, 80]
1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
2, 3, 3,
2,
  2,
3, 3,
3, 3,
3, 3,
4, 5,
```

```
2, 2,
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2]
```

```
2, 2,
2, 2,
3,
 3, 3, 3, 3,
3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5,
 5,
 5, 5, 5,
5, 5,
6, 6, 6, 6]
```

```
1, 1,
2, 2, 2, 2]
3, 3,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
2,
  2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
  2, 2, 2,
  2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2,
  2, 2,
2, 2,
2,
2, 2, 2,
2]
```

```
80, 80]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
2, 2,
5, 5,
6]
```

```
2]
2, 2,
3, 3,
3, 3,
```

```
80, 80]
2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
  5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2,
  2, 2, 2, 2,
2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
2,
  2, 2, 2,
  2, 2,
2, 2, 2, 2, 2, 2]
2, 2, 2, 2, 2,
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
3, 3,
   3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
    3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
    3, 3,
3, 3,
3, 3,
3, 3,
3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2]
36, 36, 36, 36, 36, 36, 36, 36, 36, 36]
```

```
1, 1, 1, 1, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5,
6, 6]
2, 2]
7, 7]
```

```
2,
  2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2,
2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7,
7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
```

```
6, 6]
2, 2]
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
6, 6,
7, 7]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
```

```
54]
3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2, 2,
2, 2,
6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2, 2, 2, 2,
3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3, 3,
  3, 3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2]
62, 62, 62, 62, 62, 62, 62, 62]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3,
6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2]
6, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2]
6, 6, 6, 6
2, 2, 2, 2]
```

```
7, 7, 7, 7]
2, 2,
```

```
2,
 2, 2, 2, 2,
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
2, 2, 2, 2, 2,
2, 2, 2,
  2, 2,
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
 3, 3, 3,
```

```
5, 5,
6]
```

```
2]
2, 2,
```

```
7]
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3, 3, 3,
     3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
    3, 3, 3, 3, 3, 3,
     3,
      3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2, 2,
2, 2]
36, 36, 36, 36, 36, 36, 36, 36, 36, 36]
1, 1, 1,
1, 1, 1, 1, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5,
6, 6]
```

```
2, 2,
2, 2]
7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
48, 48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3,
3, 3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7,
7, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2,
  2,
  2, 2, 3,
```

```
6, 6]
2, 2]
```

```
7, 7]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
```

```
54]
3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80, 80]
2, 2,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2,
2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2]
2, 2, 2, 2, 2,
2, 2,
3, 3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3,
3, 3,
3, 3,
3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
77, 77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80,
```

```
80, 80, 80, 80, 80]
2, 2, 2]
6, 6, 6
2, 2, 2]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7,
7, 7, 7]
```

```
2, 2, 2, 2, 2, 2]
3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6,
3, 3,
6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2]
3, 3, 3, 3, 3,
```

7, 7, 7, 7, 7, 7] 2]

```
80, 80]
2, 2,
3,
  3, 3, 3, 3,
3, 3,
  3, 3, 3, 3, 3,
5, 5,
  5,
  5,
  5, 5,
5, 5,
5, 5,
5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
 5, 5,
 5, 5, 5, 5, 5, 5, 5, 5,
```

```
6]
```

```
2, 2,
3, 3,
3, 3,
3, 3,
```

```
80, 80]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80]
```

```
3, 3,
5, 5,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
3, 3,
```

```
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
80, 80, 80, 80, 80, 80, 80]
2, 2]
36, 36, 36, 36, 36, 36, 36, 36, 36, 36]
1, 1, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5,
6, 6]
```

```
2, 2]
7, 7]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
48, 48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50,
```

```
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
2, 2,
2,
2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3,
    3,
    3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
    3,
    3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3, 3,
    3, 3, 3, 3,
3, 3,
3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
2, 2,
2, 2, 2, 2, 2, 2, 2, 2]
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3,
6, 6, 6, 6, 6, 6, 6, 6]
```

```
2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7,
7, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2]
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2,
 2,
 2, 2,
3, 3, 3,
3,
3, 3,
5,
6, 6, 6
```

```
2, 2, 2]
7, 7, 7]
```

```
2, 2, 2,
```

```
3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
3, 3,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
80]
3, 3,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
80]
2, 2, 2,
2, 2,
2, 2, 2]
```

```
1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
6, 6, 6
2, 2, 2]
3, 3,
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7,
7, 7, 7]
2, 2, 2, 2, 2, 2, 2, 2]
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80, 80]
3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6,
6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2]
4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7,
7, 7, 7, 7, 7, 7, 7, 7]
```

```
2,
 2, 2, 2,
2, 2,
2, 2,
2,
 2, 2, 2, 2,
2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2,
```

```
80, 80]
```

```
6]
```

```
2]
2, 2,
```

```
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
  3, 3, 3, 3, 3, 3,
   3,
   3, 3, 3, 3,
3, 3,
3,
3, 3,
3, 3,
3,
7]
```

```
80, 80]
1, 1, 1,
```

```
2, 2, 2, 2, 2]
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 1,
2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
2, 2,
2, 2, 2, 2, 2,
2, 2,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
2, 2,
```

```
2, 2,
 2, 2, 2, 2, 2,
2, 2,
3, 3,
3,
3, 3,
3, 3, 3,
3, 3, 3, 3,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3, 3,
 3, 3, 3, 3,
3, 3,
```

```
80, 80, 80, 80, 80, 80, 80, 80, 80]
```

```
2, 2, 2, 2, 2]
6, 6, 6, 6, 6]
2, 2, 2, 2, 2]
7, 7, 7, 7, 7]
```

```
2,
2,
   2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
  2,
   2,
   2, 2,
2, 2,
2,
2, 2, 2, 2,
2,
2, 2, 2,
2, 2,
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50,
```

```
2, 2,
 2, 2, 2, 2, 2,
3, 3,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
```

```
2, 2,
```

```
48, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50,
```

```
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
47, 47, 47]
6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]
```

```
1, 1, 1, 1, 2,
2, 2, 2, 2, 2,
2, 2,
2,
2,
   2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
  2, 2, 2, 2, 2,
   2,
   2, 2, 2,
2, 2,
2,
2, 2, 2,
2, 2, 2]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
```

```
6, 6, 6
2, 2, 2]
```

```
3, 3, 3, 3, 3,
3, 3,
3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3, 3, 3, 3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3, 3, 3,
3, 3, 3, 3, 3,
 3, 3, 3, 3, 3, 3, 3, 3,
3,
3, 3, 3,
   3, 3, 3, 3, 3, 3,
     3,
     3, 3,
3, 3,
3, 3,
7, 7, 7]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
```

```
2, 2]
8, 48, 48, 48, 49, 49, 49, 49, 49, 49, 49, 49, 49, 50, 50, 50, 50, 50, 50,
57, 57, 57, 57, 57, 57, 57, 57, 57, 57]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5,
6, 6]
2, 2]
7, 7]
```

```
2]
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
3, 3,
6]
```

```
2, 2, 2, 2, 2,
2]
1, 2,
3, 3,
3, 3, 3, 3, 3,
3, 3,
3, 3,
3, 3,
7]
77, 78, 78, 78, 78, 78, 78, 78, 79, 79, 79, 79, 80, 80, 80, 80, 80, 80, 80, 80,
80, 80, 80, 80]
```

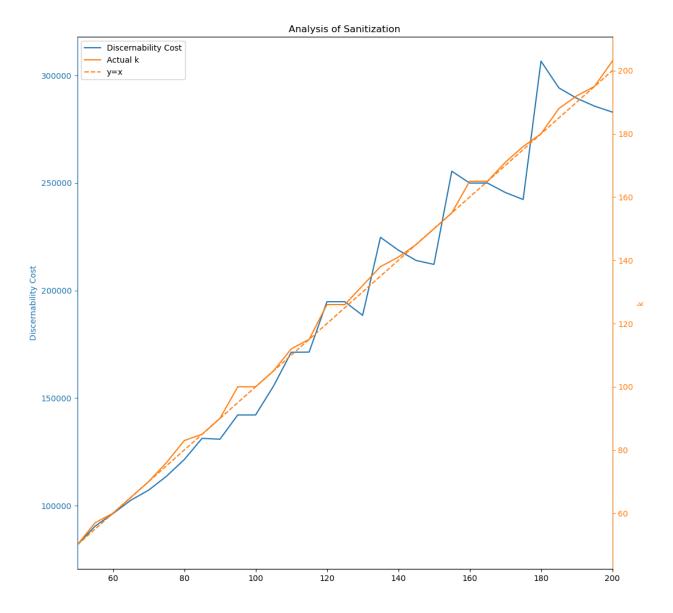
```
65, 65, 65, 65]
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]
4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7]
2, 2, 2, 2, 2, 2]
```

```
801
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6,
6, 6, 6, 6, 6, 6]
2, 2, 2, 2, 2, 2]
7, 7, 7, 7, 7, 7]
```

Run the following cell to plot the outcomes of your code. The provided code will generate the following \$3\$ plots:

- A plot of the discernability cost (on the y-axis) vs the desired k (on the x-axis).
- A plot of the actual k achieved (on a second y-axis; using yyaxis right) vs the desired k (on the x-axis).
- A plot of the line x=y by plotting the desired k (on the second y-axis) vs the desired k (on the x-axis).

```
problem = '1i'
In [205...
         try:
             qID = ['Gender','Age','Marital Status','Country Birth','Race']
             pli desired ks = range(50,201,5)
             pyplot.figure(figsize=(12,12))
             ax1 = pyplot.gca()
             plt1=pyplot.plot(pli desired ks, pli discernability costs, color='C0')
             pyplot.ylabel('Discernability Cost', color='C0')
             pyplot.tick params(axis='y', color='C0', labelcolor='C0')
             ax2 = ax1.twinx()
             plt2=pyplot.plot(p1i desired ks, p1i actual k values, color='C1')
             # For refeerence
             plt3=pyplot.plot(pli desired ks, pli desired ks, color='C1', linestyle='--
             pyplot.xlim([min(pli_desired_ks), max(pli_desired_ks)])
             pyplot.ylabel('k', color='C1')
             pyplot.xlabel('desired k')
             pyplot.tick_params(axis='y', color='C1', labelcolor='C1')
             ax2.spines['right'].set color('C1')
             ax2.spines['left'].set color('C0')
             pyplot.title('Analysis of Sanitization')
             pyplot.legend(plt1+plt2+plt3,['Discernability Cost','Actual k','y=x'])
             pyplot.show()
         except Exception as e:
             safe print err(e)
```



In this Markdown cell, answer the following question

If you had to choose between the option of desired k=135 vs desired k=150, which one would you choose? Explain.

Place your answer here I would choose desired \$k=150\$ because from that point, the line with desired k starts to align with the line of actual k

Problem 2

In this problem, we will study \$\delta\$-Presence. In this problem, we are simulating a selective publication, whereby the full list of possible individuals is presented in

table Lab3_synthetic_identities (a table of simulated identities). There is a total of \$1301\$ individuals in this table. However, the published data, represented by variable Lab3_Problem2 has the data of \$405\$ individuals out of the full set of individuals.

In this problem, we will calculate the different values of \$\delta\$ for this type of publication, each time assuming a different set of quasi-identifiers (in effect, simulating that only the set of quasi-identifiers in each part is published, and the rest of the columns are not published).

(a) \$\delta\$-Presnece for an Individual

In this part, we will focus on a single individual (i.e., record) from Lab3_synthetic_identities, and calculate the probability that this individual is in the published data Lab3 Problem2.

Complete the function delta_Presence_of_Individual below to achieve this.

The function accepts 4 arguments:

- identities: a table with the identities of the individuals and their quasi-identifiers
- identities_row_id: the row number of the person in the identities table, for which we would like to calculate \$\delta\$
- published table: the published table
- quasi_identifiers : a list containing the quasi-identifiers.

The function outputs one argument:

• delta: the probability that the individual in row identities_row_id is part of the publication published_table (i.e., \$\delta\$, as defined in class).

Complete the following code with your solution.

```
In [206...

def delta_Presence_of_Individual(identities, identities_row_id, published_tabl
    # Your solution here
    #raise Exception("Implementation of <i>delta_Presence_of_Individual</i> is
    cnt_identities = identities[quasi_identifiers][identities[quasi_identifier
    cnt_published_table = published_table[quasi_identifiers][published_table[c
    if cnt_identities > 0:
        delta = cnt_published_table / cnt_identities
    else:
        delta = 0.0
    return delta
```

Run the following cell to print some outcomes based on your code.

```
In [207...
         _{\rm problem} = '2a'
         try:
             qID = ['Marital_Status']
             next row = 99
             next_delta = delta_Presence_of_Individual(Lab3_synthetic_identities, next_
             print mk(f'Assuming qID={qID}, the delta value for row {next row} is $\del
             next row = 101
             next delta = delta Presence of Individual(Lab3 synthetic identities, next
             print mk(f'Assuming qID={qID}, the delta value for row {next row} is $\del
             next row = 104
             next delta = delta Presence of Individual(Lab3 synthetic identities, next
             print mk(f'Assuming qID={qID}, the delta value for row {next row} is $\del
             next row = 92
             next_delta = delta_Presence_of_Individual(Lab3_synthetic_identities, next_
             print_mk(f'Assuming qID={qID}, the delta value for row {next_row} is $\del
             next row = 1000
             next_delta = delta_Presence_of_Individual(Lab3_synthetic_identities, next_
             print mk(f'Assuming qID={qID}, the delta value for row {next row} is $\del
             qID = ['Marital Status', 'Age']
             next row = 99
             next delta = delta Presence of Individual(Lab3 synthetic identities, next
             print mk(f'Assuming qID={qID}, the delta value for row {next row} is $\del
             next row = 7
             next delta = delta Presence of Individual(Lab3 synthetic identities, next
             print_mk(f'Assuming qID={qID}, the delta value for row {next_row} is $\del
             next row = 870
             next_delta = delta_Presence_of_Individual(Lab3_synthetic_identities, next_
             print_mk(f'Assuming qID={qID}, the delta value for row {next_row} is $\del
             next row = 90
             next_delta = delta_Presence_of_Individual(Lab3_synthetic_identities, next_
             print mk(f'Assuming qID={qID}, the delta value for row {next row} is $\del
             next row = 1024
             next delta = delta Presence of Individual(Lab3 synthetic identities, next
             print mk(f'Assuming qID={qID}, the delta value for row {next row} is $\del
         except Exception as e:
             safe print err(e)
```

Assuming qID=['Marital_Status'], the delta value for row 99 is \$\delta=0.3067\$

Assuming qID=['Marital_Status'], the delta value for row 101 is \$\delta=0.3211\$

Assuming qID=['Marital_Status'], the delta value for row 104 is \$\delta=0.3178\$

Assuming qID=['Marital_Status'], the delta value for row 92 is \$\delta=0.2770\$

Assuming qID=['Marital_Status'], the delta value for row 1000 is \$\delta=0.3235\$

Assuming qID=['Marital_Status', 'Age'], the delta value for row 99 is \$\delta=0.5000\$

Assuming qID=['Marital_Status', 'Age'], the delta value for row 7 is \$\delta=0.2222\$

Assuming qID=['Marital_Status', 'Age'], the delta value for row 870 is \$\delta=0.4286\$

Assuming qID=['Marital_Status', 'Age'], the delta value for row 90 is \$\delta=0.2353\$
Assuming qID=['Marital_Status', 'Age'], the delta value for row 1024 is
\$\delta=0.4375\$

(b) Individual Risk

What is the risk of membership disclosure for 'Gregg Maxim Beale' if the data Lab3_Problem2 is published with 'Gender' and 'Race' as the sole quasi-identifiers?

Store the result in the variable p2b delta.

Complete the following code snippet with your code:

```
In [208... qID = ['Gender', 'Race']
    person_name = 'Gregg Maxim Beale'

# Your solution here
# get index of row containing person_name
    row_id = Lab3_synthetic_identities.index[Lab3_synthetic_identities.Full_Name = p2b_delta = delta_Presence_of_Individual(Lab3_synthetic_identities, row_id, Laba_synthetic_identities)
```

Run the following cell to present the outcomes of your code.

(c) Complete \$\delta\$-Presnece

In this part, we will complete the full \$\delta\$-Presence calculation algorithm in the function delta Presence.

delta Presence accepts three arguments:

- identities: a table with the identities of the individuals and their quasi-identifiers
- published table : the published table
- quasi identifiers : a list containing the quasi-identifiers.

The function outputs three arguments:

- delta min : The value of \$\delta {\min}\$ as defined in class.
- delta_max : The value of \$\delta_{\max}\$ as defined in class.
- identities_deltas: A copy of the data frame identities with an additional column delta. Populate the new column delta and assign to it the risk of positive membership disclosure (\$\delta\$) of the corresponding row in identities with respect to published_table and quasi_identifiers (as defined in class).

After implementing the function delta_Presence, use the function to calculate the \$\delta\$-Presence levels of the individiuals in identities with respect to published data Lab3_Problem2, assuming only "Marital_Status" gets published as a quasi-identifier.

Store the results of your calculations in the variables p2c_delta_min , p2c_delta_max and p2c_identities_deltas , respectively.

Complete the following code with your solution.

```
In [210...

def delta_Presence(identities, published_table, quasi_identifiers):
    # Your implementation here
    #raise Exception("Implementation of <i>delta_Presence</i> is missing")
    def delta_func(val_identities, val_published):
        if val_identities > 0:
            delta_val = val_published / val_identities
        else:
            detal_val = 0

        return delta_val
```

```
tmp_identities = identities.groupby(quasi_identifiers).apply(lambda x: ler
tmp_published_table = published_table.groupby(quasi_identifiers).apply(lam

tmp_identities = pandas.merge(tmp_identities, tmp_published_table[quasi_identities['delta'] = tmp_identities['cnt_published_table'].div(tmp_identities_deltas = identities.copy()

identities_deltas = pandas.merge(identities_deltas, tmp_identities[quasi_i

#identities_deltas['delta'] = identities_deltas.groupby(quasi_identifiers)

delta_min = min(identities_deltas.delta)
    delta_max = max(identities_deltas.delta)

return delta_min, delta_max, identities_deltas

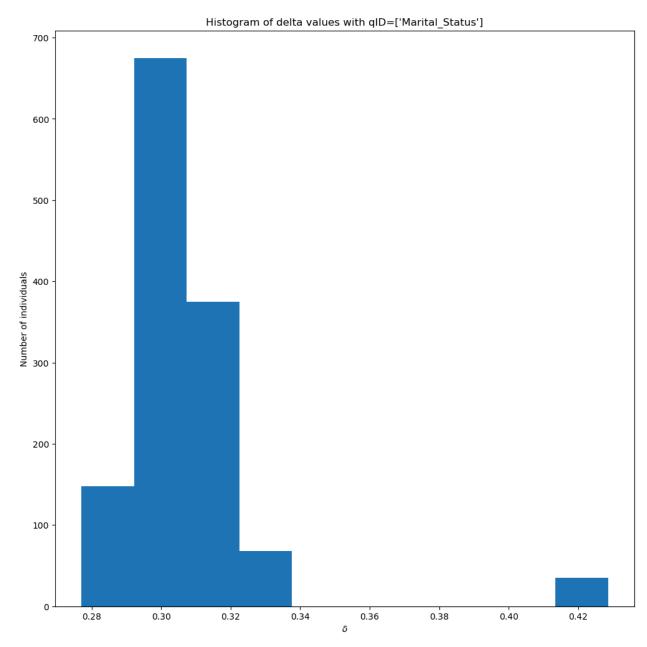
qID = ['Marital_Status']

# Use delta_Presence() to calculate p2c_delta_min, p2c_delta_max and p2c_ident
p2c_delta_min, p2c_delta_max, p2c_identities_deltas = delta_Presence(Lab3_synt)
```

Run the following cell to present the outcomes of your code.

```
In [211... problem = '2c'
         try:
             qID = ['Marital_Status']
             print mk(f'The results of running `delta Presence` with qID={qID} are `del
             pyplot.subplots(figsize=(12,12))
             pyplot.hist(p2c identities deltas.delta, 10)
             pyplot.xlabel("$\delta$")
             pyplot.ylabel("Number of individuals")
             pyplot.title(f"Histogram of delta values with qID={qID}")
             pyplot.show()
             print mk(f'The individuals with delta=delta min=${p2c delta min:.4f}$ are:
             display(p2c identities deltas[p2c identities deltas.delta == p2c delta mir
             print mk(f'The individuals with delta=delta max=${p2c delta max:.4f}$ are:
             display(p2c_identities_deltas[p2c_identities_deltas.delta == p2c_delta_max
         except Exception as e:
             safe_print_err(e)
```

The results of running delta_Presence with qID=['Marital_Status'] are delta_min =\$0.2770\$ and delta_max =\$0.4286\$ with the following distribution of delta values:



The individuals with delta=delta_min=\$0.2770\$ are:

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
1118	Teddy Edouard Barris	1	44	3	2	6	0.277027
1119	Finn Avrom Collin	1	70	3	1	2	0.277027
1120	Marilyn Toinette Bradly	2	50	3	1	4	0.277027
1121	Elwina Sharia Devin	2	64	3	2	2	0.277027
1122	lain Han Bear	1	71	3	1	3	0.277027
1261	Parwane Lenee Aldo	2	75	3	2	3	0.277027
1262	Engelbart Rawley Caspar	1	65	3	1	4	0.277027
1263	Jana Mead Brent	2	69	3	1	3	0.277027
1264	Reeta Loria Drew	2	51	3	1	4	0.277027
1265	Em Phylys Bradley	2	63	3	1	3	0.277027

148 rows × 7 columns

The individuals with delta=delta_max=\$0.4286\$ are:

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
1266	Benoite Prudi Beau	2	46	4	2	1	0.428571
1267	Zondra Nadya Cyrill	2	62	4	1	4	0.428571
1268	Stan Jerri Bennett	1	50	4	1	3	0.428571
1269	Vassili Kenyon Ahmed	1	40	4	1	3	0.428571
1270	Radcliffe Radcliffe Aube	1	51	4	2	1	0.428571
1271	Crysta Darlene Bartlet	2	42	4	1	3	0.428571
1272	Valaree Phylys Benjie	2	33	4	2	2	0.428571
1273	Arielle Angelia Albrecht	2	73	4	1	3	0.428571
1274	Damita Rosario Demetris	2	36	4	1	4	0.428571
1275	Niels Horst Andri	1	51	4	1	3	0.428571
1276	Paolina Elenore Corby	2	66	4	1	3	0.428571
1277	Georgia Charles Carson	1	49	4	1	3	0.428571
1278	Fidelity Adena Earle	2	45	4	1	4	0.428571
1279	Edgar Barde Aylmer	1	52	4	2	1	0.428571
1280	Marianne Prudi Davon	2	49	4	1	4	0.428571
1281	Guthry Praneetf Del	1	50	4	2	2	0.428571
1282	Sandy	1	68	4	1	3	0.428571

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
	Muhammad Antoine						
1283	Ashley Ty Demetrius	1	43	4	1	4	0.428571
1284	Cyndi Harrietta Darin	2	41	4	1	3	0.428571
1285	Tymothy Son Carlin	1	29	4	1	1	0.428571
1286	Lena Heidi Davidde	2	48	4	1	3	0.428571
1287	Tye Nels Aguste	1	73	4	2	2	0.428571
1288	Rosamund Tiffy Dominique	2	41	4	1	4	0.428571
1289	Allie Abbie Andreas	1	28	4	1	3	0.428571
1290	Tailor Trenton Andrus	1	29	4	2	2	0.428571
1291	Taryn Aimee Ahmet	2	70	4	1	4	0.428571
1292	Ramsey Stavros Christofer	1	59	4	1	4	0.428571
1293	Sisely Rhonda Antoni	2	22	4	2	2	0.428571
1294	Ingmar Jere Barny	1	71	4	2	2	0.428571
1295	Jackie Wald Arthur	1	62	4	1	2	0.428571
1296	Gibb Hy Chet	1	53	4	1	4	0.428571
1297	Lynda Noelle Augustus	2	48	4	2	1	0.428571
1298	Stoddard Prunella Dillon	2	57	4	1	2	0.428571

		Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
3	L299	Jessamyn Ernaline Anson	2	40	4	2	1	0.428571
1	L300	Edithe Hephzibah Carlos	2	59	4	1	4	0.428571

(d) What About Gender?

Use the function delta_Presence to calcualte the \$\delta\$-Presence levels of the individiuals in identities with respect to published data Lab3_Problem2, assuming only "Gender" gets published as a quasi-identifier.

Store the results of your calculations in the variables p2d_delta_min, p2d_delta_max and p2d_identities_deltas, respectively.

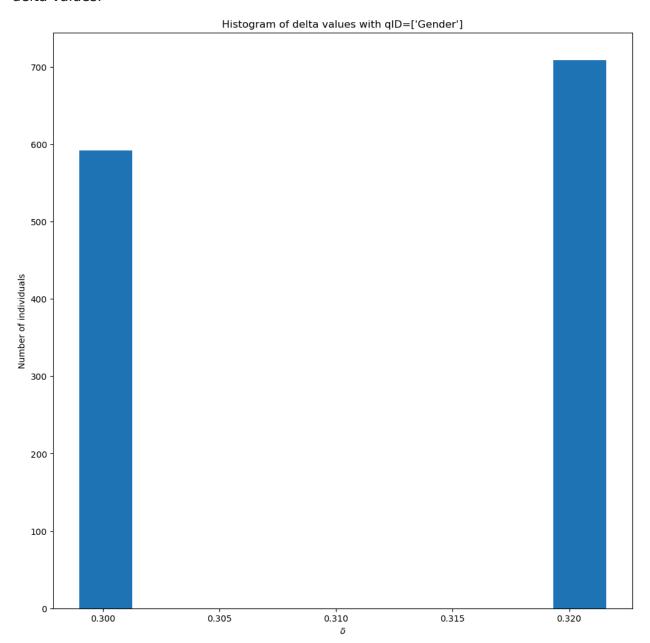
Complete the following code snippet with your code:

```
In [212... qID = ['Gender']
# Your solution here
p2d_delta_min, p2d_delta_max, p2d_identities_deltas = delta_Presence(Lab3_synt)
```

Run the following cell to present the outcomes of your code.

```
In [213...
         problem = '2d'
         try:
             qID = ['Gender']
             print mk(f'The results of running `delta Presence` with qID={qID} are `del
             pyplot.subplots(figsize=(12,12))
             pyplot.hist(p2d identities deltas.delta, 10)
             pyplot.xlabel("$\delta$")
             pyplot.ylabel("Number of individuals")
             pyplot.title(f"Histogram of delta values with qID={qID}")
             pyplot.show()
             print_mk(f'The individuals with delta=delta_min=${p2d_delta_min:.4f}$ are:
             display(p2d identities deltas[p2d identities deltas.delta == p2d delta mir
             print_mk(f'The individuals with delta=delta_max=${p2d_delta_max:.4f}$ are:
             display(p2d_identities_deltas[p2d_identities_deltas.delta == p2d_delta_max
         except Exception as e:
             safe_print_err(e)
```

The results of running delta_Presence with qID=['Gender'] are delta_min =\$0.2990\$ and delta_max =\$0.3216\$ with the following distribution of delta values:



The individuals with delta=delta_min=\$0.2990\$ are:

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
709	Almeria Casandra Artur	2	59	5	1	4	0.298986
710	Mufinella Bebe Aylmer	2	56	2	1	3	0.298986
711	Elsbeth Mandy Berchtold	2	53	1	2	6	0.298986
712	Bessie Clare Claybourne	2	31	5	2	4	0.298986
713	Lebbie Xylia Brant	2	37	1	2	3	0.298986
1296	Diandra Sarah Carlyle	2	47	1	1	4	0.298986
1297	Betsey Jessalyn Davis	2	66	1	1	3	0.298986
1298	Lorita Karena Domenic	2	31	5	1	4	0.298986
1299	Sonya Marrissa Claus	2	55	5	1	1	0.298986
1300	Sydney Ardella Cletus	2	30	5	1	4	0.298986

592 rows × 7 columns

The individuals with $delta=delta_max=\$0.3216\$$ are:

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
0	Jerold Washington Billie	1	65	1	2	2	0.32158
1	Percival Jesse Denis	1	20	6	1	3	0.32158
2	Tim Brinkley Derrol	1	26	5	1	4	0.32158
3	Barth Tonnie Cob	1	50	1	1	4	0.32158
4	Floyd Terrance Corwin	1	66	5	2	2	0.32158
704	Ludvig Isador Boyd	1	26	1	1	4	0.32158
705	Ramon Hilbert Abdullah	1	26	6	1	7	0.32158
706	Chandler Georgie Abram	1	31	6	1	3	0.32158
707	Chaunce Grove Armando	1	25	5	2	6	0.32158
708	Slade Lukas Aloysius	1	70	1	1	4	0.32158

709 rows \times 7 columns

(e) More Options

Use the function delta_Presence to calcualte the \$\delta\$-Presence levels of the individiuals in identities with respect to published data Lab3_Problem2, assuming only "Gender" and "Marital_Status" get published as quasi-identifiers.

Store the results of your calculations in the variables p2e_delta_min , p2e_delta_max and p2e_identities_deltas , respectively.

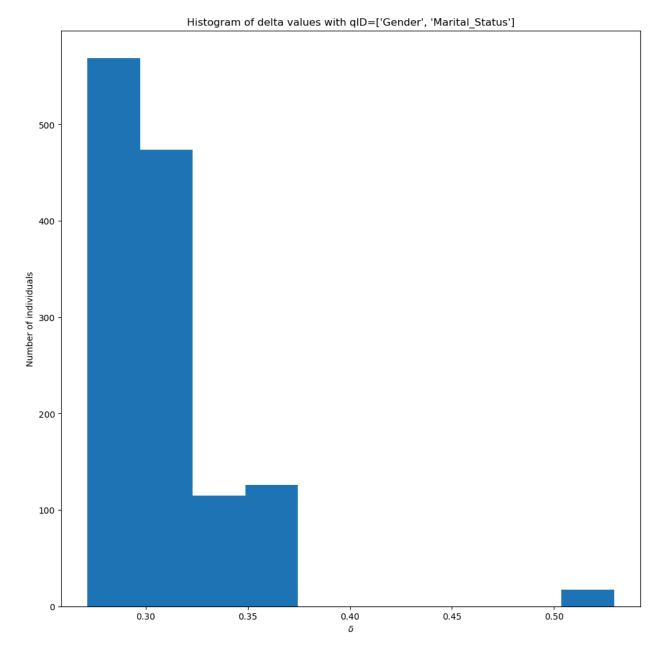
Complete the following code snippet with your code:

```
# Your solution here
p2e_delta_min, p2e_delta_max, p2e_identities_deltas = delta_Presence(Lab3_synt
```

Run the following cell to present the outcomes of your code.

```
In [215...
         problem = '2e'
         try:
             qID = ['Gender','Marital_Status']
             print mk(f'The results of running `delta Presence` with qID={qID} are `del
             pyplot.subplots(figsize=(12,12))
             pyplot.hist(p2e identities deltas.delta, 10)
             pyplot.xlabel("$\delta$")
             pyplot.ylabel("Number of individuals")
             pyplot.title(f"Histogram of delta values with qID={qID}")
             pyplot.show()
             print mk(f'The individuals with delta=delta min=${p2e delta min:.4f}$ are:
             display(p2e identities deltas[p2e identities deltas.delta == p2e delta mir
             print_mk(f'The individuals with delta=delta_max=${p2e_delta_max:.4f}$ are:
             display(p2e identities deltas[p2e identities deltas delta == p2e delta max
         except Exception as e:
             safe_print_err(e)
```

The results of running delta_Presence with qID=['Gender', 'Marital_Status'] are delta_min =\$0.2714\$ and delta_max =\$0.5294\$ with the following distribution of delta values:



The individuals with delta=delta_min=\$0.2714\$ are:

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
1045	Teddy Edouard Barris	1	44	3	2	6	0.271429
1046	Finn Avrom Collin	1	70	3	1	2	0.271429
1047	lain Han Bear	1	71	3	1	3	0.271429
1048	Blaine Lew Dario	1	40	3	2	3	0.271429
1049	Osbourn Sebastiano Chris	1	75	3	1	4	0.271429
1110	Ingelbert Mordecai Bjorne	1	53	3	2	3	0.271429
1111	Cosmo Joachim Aleck	1	38	3	1	3	0.271429
1112	Wilbur Wood Beale	1	63	3	1	4	0.271429
1113	Hodge Randolf Daffy	1	58	3	1	3	0.271429
1114	Engelbart Rawley Caspar	1	65	3	1	4	0.271429

70 rows \times 7 columns

The individuals with delta=delta_max=\$0.5294\$ are:

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
1284	Stan Jerri Bennett	1	50	4	1	3	0.529412
1285	Vassili Kenyon Ahmed	1	40	4	1	3	0.529412
1286	Radcliffe Radcliffe Aube	1	51	4	2	1	0.529412
1287	Niels Horst Andri	1	51	4	1	3	0.529412
1288	Georgia Charles Carson	1	49	4	1	3	0.529412
1289	Edgar Barde Aylmer	1	52	4	2	1	0.529412
1290	Guthry Praneetf Del	1	50	4	2	2	0.529412
1291	Sandy Muhammad Antoine	1	68	4	1	3	0.529412
1292	Ashley Ty Demetrius	1	43	4	1	4	0.529412
1293	Tymothy Son Carlin	1	29	4	1	1	0.529412
1294	Tye Nels Aguste	1	73	4	2	2	0.529412
1295	Allie Abbie Andreas	1	28	4	1	3	0.529412
1296	Tailor Trenton Andrus	1	29	4	2	2	0.529412
1297	Ramsey Stavros Christofer	1	59	4	1	4	0.529412
1298	Ingmar Jere Barny	1	71	4	2	2	0.529412
1299	Jackie Wald Arthur	1	62	4	1	2	0.529412
1300	Gibb Hy Chet	1	53	4	1	4	0.529412

(f) Even More Options

Use the function delta_Presence to calcualte the \$\delta\$-Presence levels of the individiuals in identities with respect to published data Lab3_Problem2, assuming only "Gender", "Race" and "Marital_Status" get published as quasi-identifiers.

Store the results of your calculations in the variables p2f_delta_min, p2f_delta_max and p2f_identities_deltas, repsectively.

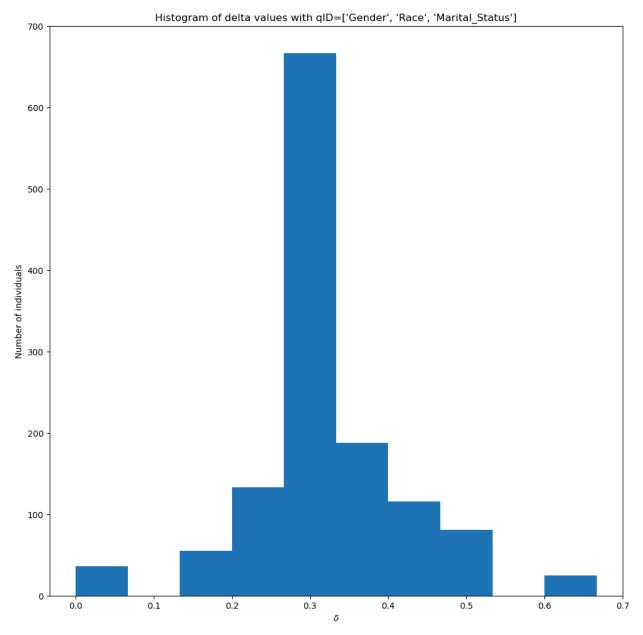
Complete the following code snippet with your code:

```
In [216... qID = ['Gender','Race','Marital_Status']
# Your solution here
p2f_delta_min, p2f_delta_max, p2f_identities_deltas = delta_Presence(Lab3_synt)
```

Run the following cell to present the outcomes of your code.

```
In [217... problem = '2f'
         try:
             qID = ['Gender', 'Race', 'Marital Status']
             print mk(f'The results of running `delta Presence` with qID={qID} are `del
             pyplot.subplots(figsize=(12,12))
             pyplot.hist(p2f identities deltas.delta, 10)
             pyplot.xlabel("$\delta$")
             pyplot.ylabel("Number of individuals")
             pyplot.title(f"Histogram of delta values with qID={qID}")
             pyplot.show()
             print mk(f'The individuals with delta=delta min=${p2f delta min:.4f}$ are:
             display(p2f identities deltas[p2f identities deltas.delta == p2f delta min
             print_mk(f'The individuals with delta=delta_max=${p2f_delta_max:.4f}$ are:
             display(p2f identities deltas[p2f identities deltas.delta == p2f delta max
         except Exception as e:
             safe print err(e)
```

The results of running delta_Presence with qID=['Gender', 'Race', 'Marital_Status'] are delta_min =\$0.0000\$ and delta_max =\$0.6667\$ with the following distribution of delta values:



The individuals with delta=delta_min=\$0.0000\$ are:

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
604	Teddy Edouard Barris	1	44	3	2	6	0.0
605	Abbie Baillie Antonius	1	55	3	2	6	0.0
606	Francis Jerri Brandy	1	66	3	2	6	0.0
1182	Pablo Eduard Baillie	1	69	3	1	7	0.0
1183	Inigo Alphonso Aldrich	1	35	3	1	7	0.0
1184	Hamel Rodd Christoph	1	72	3	1	7	0.0
1185	Niccolo Raymond Alejandro	1	72	3	1	7	0.0
1186	Bear Arel Carlie	1	38	3	1	7	0.0
1187	Corby Sumner Darcy	1	33	3	1	7	0.0
1188	Iggie Jamey Apostolos	1	74	3	1	7	0.0
1244	Valaree Phylys Benjie	2	33	4	2	2	0.0
1245	Sisely Rhonda Antoni	2	22	4	2	2	0.0
1246	Stoddard Prunella Dillon	2	57	4	1	2	0.0
1254	Georgiamay Pollyanna Alan	2	56	3	2	6	0.0
1255	Ealasaid Cristy Amadeus	2	64	2	1	4	0.0
1256	Kathe Harriett Derrol	2	44	2	1	4	0.0
1257	Coreen Ebony Cass	2	52	2	1	4	0.0
1258	Tony Shannon Benedict	2	62	2	1	4	0.0
1259	Ann Yolanda Andri	2	60	2	1	4	0.0
1260	Marga Anne- Marie Arron	2	59	2	2	4	0.0

Parkas Palarita		
Barbee Dolorita 2 21 5 1 Andreas	7	0.0
1262 Harrietta Albina 2 36 5 1	7	0.0
Shayna Edwina 2 25 5 1	7	0.0
Cassandra 1264 Mellicent 2 37 5 1 Delbert	7	0.0
1265 Chanda Ninette 2 34 6 2	6	0.0
1266 Ag Jonie Alonzo 2 25 6 2	6	0.0
Pollyanna 1267 Lyndsey 2 23 6 2 Clarence	6	0.0
1268 Perla Ardyth 2 26 6 1	6	0.0
1280 Aaron Augusto 1 21 5 1	7	0.0
Shelden Tuckie 1 45 5 1 Demetrius	7	0.0
Shayne Josef 1 23 5 1 Bartolomei	7	0.0
1283 Kaiser Nico	7	0.0
1284 Esau Gordon 1 48 5 1	7	0.0
1289 Kiley LeeAnn 2 80 2 1	6	0.0
1299 Bunny Kirstyn 2 65 2 1	7	0.0
1300 Patin Vito Chev 1 59 2 2	2	0.0

The individuals with delta=delta_max=\$0.6667\$ are:

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
1036	Benoite Prudi Beau	2	46	4	2	1	0.666667
1037	Lynda Noelle Augustus	2	48	4	2	1	0.666667
1038	Jessamyn Ernaline Anson	2	40	4	2	1	0.666667
1269	Gilles Ulric Cam	1	71	3	2	1	0.666667
1270	Dalton Loren Andie	1	49	3	1	1	0.666667
1271	Brooks Gale Bartie	1	62	3	1	1	0.666667
1272	Lind Laryssa Burke	2	27	6	1	7	0.666667
1273	Kaye Sibylla Diego	2	40	6	1	7	0.666667
1274	Ola Brigitte Barnabas	2	70	6	1	7	0.666667
1296	Ashley Ty Demetrius	1	43	4	1	4	0.666667
1297	Ramsey Stavros Christofer	1	59	4	1	4	0.666667
1298	Gibb Hy Chet	1	53	4	1	4	0.666667

(g) Even More Options (2)

Use the function delta_Presence to calcualte the \$\delta\$-Presence levels of the individiuals in identities with respect to published data Lab3_Problem2, assuming only "Gender", "Race", "Country_Birth" and "Marital_Status" get published as quasi-identifiers.

Store the results of your calculations in the variables p2g_delta_min , p2g_delta_max and p2g_identities_deltas , respectively.

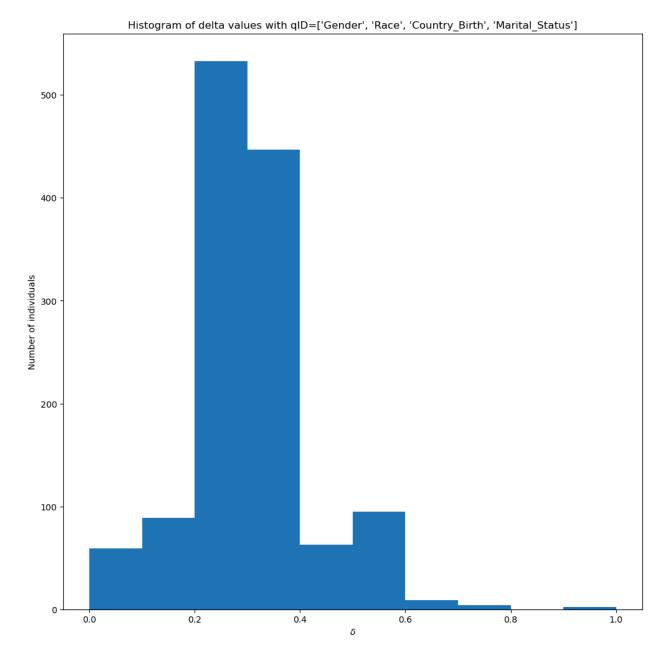
Complete the following code snippet with your code:

```
In [218... qID = ['Gender', 'Race', 'Country_Birth', 'Marital_Status']
# Your solution here
p2g_delta_min, p2g_delta_max, p2g_identities_deltas = delta_Presence(Lab3_synt)
```

Run the following cell to present the outcomes of your code.

```
In [219...
         __problem = '2g'
         try:
             qID = ['Gender','Race','Country_Birth','Marital_Status']
             print_mk(f'The results of running `delta_Presence` with qID={qID} are `del
             pyplot.subplots(figsize=(12,12))
             pyplot.hist(p2g identities deltas.delta, 10)
             pyplot.xlabel("$\delta$")
             pyplot.ylabel("Number of individuals")
             pyplot.title(f"Histogram of delta values with qID={qID}")
             pyplot.show()
             print mk(f'The individuals with delta=delta min=${p2g delta min:.4f}$ are:
             display(p2g identities deltas[p2g identities deltas.delta == p2g delta min
             print mk(f'The individuals with delta=delta max=${p2g delta max:.4f}$ are:
             display(p2g_identities_deltas[p2g_identities_deltas.delta == p2g_delta_max
         except Exception as e:
             safe_print_err(e)
```

The results of running delta_Presence with qID=['Gender', 'Race', 'Country_Birth', 'Marital_Status'] are delta_min =\$0.0000\$ and delta_max =\$1.0000\$ with the following distribution of delta values:



The individuals with delta=delta_min=\$0.0000\$ are:

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
428	Teddy Edouard Barris	1	44	3	2	6	0.0
429	Abbie Baillie Antonius	1	55	3	2	6	0.0
430	Francis Jerri Brandy	1	66	3	2	6	0.0
627	Tia Ilana Brandy	2	29	1	1	6	0.0
628	Glennis Blondy Douglis	2	28	1	1	6	0.0
629	Ranique Vivien Alister	2	70	1	1	6	0.0
1090	Pablo Eduard Baillie	1	69	3	1	7	0.0
1091	Inigo Alphonso Aldrich	1	35	3	1	7	0.0
1092	Hamel Rodd Christoph	1	72	3	1	7	0.0
1093	Niccolo Raymond Alejandro	1	72	3	1	7	0.0
1094	Bear Arel Carlie	1	38	3	1	7	0.0
1095	Corby Sumner Darcy	1	33	3	1	7	0.0
1096	lggie Jamey Apostolos	1	74	3	1	7	0.0
1215	Valaree Phylys Benjie	2	33	4	2	2	0.0
1216	Sisely Rhonda Antoni	2	22	4	2	2	0.0
1224	Dorene Chelsea Andrej	2	35	1	2	7	0.0
1225	Georgiamay Pollyanna Alan	2	56	3	2	6	0.0
1229	Ealasaid Cristy Amadeus	2	64	2	1	4	0.0
1230	Kathe Harriett Derrol	2	44	2	1	4	0.0
1231	Coreen Ebony Cass	2	52	2	1	4	0.0

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
1232	Tony Shannon Benedict	2	62	2	1	4	0.0
1233	Ann Yolanda Andri	2	60	2	1	4	0.0
1234	Ettie Ronica Dyson	2	46	3	2	4	0.0
1235	Romain Swen Chris	1	33	6	2	3	0.0
1236	Pennie Emery Beaufort	1	80	2	2	3	0.0
1237	Barbee Dolorita Andreas	2	21	5	1	7	0.0
1238	Harrietta Albina Carson	2	36	5	1	7	0.0
1239	Shayna Edwina Deryl	2	25	5	1	7	0.0
1240	Cassandra Mellicent Delbert	2	37	5	1	7	0.0
1241	Chanda Ninette Andre	2	34	6	2	6	0.0
1242	Ag Jonie Alonzo	2	25	6	2	6	0.0
1243	Pollyanna Lyndsey Clarence	2	23	6	2	6	0.0
1254	Maud Vania Easton	2	52	1	2	4	0.0
1255	Misti Janetta Ashton	2	56	1	2	4	0.0
1256	Aaron Augusto Darrell	1	21	5	1	7	0.0
1257	Shelden Tuckie Demetrius	1	45	5	1	7	0.0
1258	Shayne Josef Bartolomei	1	23	5	1	7	0.0
1259	Kaiser Nico Dickie	1	21	5	1	7	0.0
1260	Esau Gordon Artur	1	48	5	1	7	0.0
1261	Cris Chet Burgess	1	35	6	2	4	0.0

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
1266	Kiley LeeAnn Demetris	2	80	2	1	6	0.0
1267	Ambrosio Salvador David	1	24	6	1	6	0.0
1268	Orin Willis Alex	1	26	6	1	6	0.0
1273	Vera Beverly Chen	2	60	3	2	1	0.0
1277	Bunny Kirstyn Beck	2	65	2	1	7	0.0
1278	Tymothy Son Carlin	1	29	4	1	1	0.0
1283	Bobbi Georgina Barnabe	2	24	6	2	4	0.0
1284	Patin Vito Chev	1	59	2	2	2	0.0
1289	Sonia Genevra Adrien	2	80	2	1	1	0.0
1290	Carlisle Judi Andre	2	80	2	1	1	0.0
1291	Dannie Forester Archie	1	26	6	1	1	0.0
1292	Joab Alasdair Brewer	1	29	5	2	4	0.0
1293	Perla Ardyth Bela	2	26	6	1	6	0.0
1295	Simonette Beatrisa Brooke	2	80	2	2	3	0.0
1296	Seamus Merill Don	1	42	2	2	1	0.0
1297	Tyson Dom Al	1	39	2	2	1	0.0
1298	Puff Marko Donnie	1	25	5	2	1	0.0
1299	Stoddard Prunella Dillon	2	57	4	1	2	0.0
1300	Marga Anne- Marie Arron	2	59	2	2	4	0.0

The individuals with delta=delta_max=\$1.0000\$ are:

	Full_Name	Gender	Age	Marital_Status	Country_Birth	Race	delta
1244	Gilles Ulric Cam	1	71	3	2	1	1.0
1294	Jackie Wald Arthur	1	62	4	1	2	1.0

Problem 3

In this problem we will briefly investigate differential privacy.

(a) Visualize Laplace

Throughout this problem, we will use the Laplace mechanism to achieve Differential Privacy. We will use the function numpy.random.laplace(mu, b, n) that generates a column vector of n random numbers independently distributed according to the distribution \$Laplace(\mu = mu, b)\$.

Write code that generates the following variables:

- p3a_laplace_samples : Generate \$100,000\$ (i.e., \$100k\$) samples distributed according to the distribution \$Laplace(0,2)\$.
- p3a_laplace_pdf : Using the x values p3a_x =
 numpy.arange(-25.0, 25.1, 0.1), generate the Laplace PDF
 function values of the same distribution \$Laplace(0,2)\$: \$\$ f(x) =
 \frac{1}{2\cdot b} \cdot exp\left(-\frac{\mid x-mu \mid}{b}\right) \$\$
 with \$\mu=0\$ and \$b=2\$. You may either evaluate this function
 directly, or use scipy.stats.laplace.pdf().

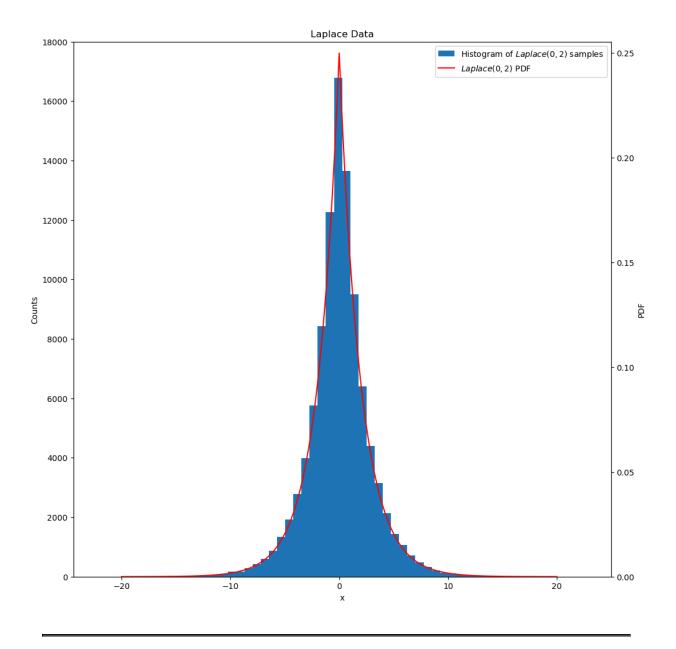
Complete the following code snippet with your code:

```
import scipy.stats
p3a_x = numpy.arange(-20.0, 20.1, 0.1)

# Your solution here
n = 100000
mu = 0
b = 2
p3a_laplace_samples = numpy.random.laplace(mu, b, n)
p3a_laplace_pdf = scipy.stats.laplace.pdf(p3a_x, 0, 2)
```

Run the following cell to present the outcomes of your code.

```
In [221... problem = '3a'
         try:
             p3a x = numpy.arange(-20.0, 20.1, 0.1)
              pyplot.subplots(figsize=(12,12))
              p3a lplc hist fh = pyplot.hist(p3a laplace samples, 60)
              pyplot.ylabel('Counts')
              pyplot.xlabel('x');
              ax1 = pyplot.gca()
              curr_ylim = ax1.get_ylim()
              display(curr ylim)
              new ylim = [0, max(numpy.ceil([17/2, curr ylim[1]/2000])*2000)]
              display(new ylim)
              pyplot.ylim(new ylim)
              ax2 = ax1.twinx()
              p3a_lplc_pdf_fh = pyplot.plot(p3a_x,p3a_laplace_pdf,'r')
              new ylim = [0, 0.25/\text{curr ylim}[1]*\text{new ylim}[1]]
              display(new ylim)
              pyplot.ylim(new ylim)
              pyplot.legend([p3a lplc hist fh[2]] + p3a lplc pdf fh, ['Histogram of $Lap
              pyplot.ylabel('PDF')
              pyplot.title('Laplace Data')
              pyplot.show()
         except Exception as e:
              safe print err(e)
        (0.0, 17616.9)
        [0, 18000.0]
        [0, 0.2554365410486521]
```



(b) Adding Laplace Noise to Query Output

In this part, we are concerned with the query:

Query 3b: Calculate the number of people with all of the following features:

- 1. Gender = Male
- 2. LDL + HDL Cholesterol > 140
- 3. Total Cholesterol < 190
- 4. Systolic Blood Pressure > 130
- 5. Average Alcohol Consumption > 1

which can be implemented with the following code (without any Differential Privacy)

```
In [222... def p3b query(data):
              """Calcualte the result of Query 3b
             Parameters
             data : pandas.DataFrame
                 A data frame containing at least the following columns:
                      - Gender
                      - HDL Cholesterol
                     - LDL Cholesterol
                      - Total Cholesterol
                     - Sys Blood Pressure
                     - Alcohol Average
             Returns
              . _ _ _ _ _
             result : list
                 a list with a single number containing the result of the query
             query = (data.Gender == 1) & \
                  (data.HDL Cholesterol + data.LDL Cholesterol > 140) & \
                  (data.Total Cholesterol < 190) & \
                  (data.Sys Blood Pressure > 130) & \
                  (data.Alcohol Average > 1)
              return [sum(query)]
         print mk(f'The true output of the query is: {p3b query(Lab3 Data)}\n')
```

The true output of the query is: [44]

We would like to achieve \$\epsilon\$-Differential Privacy for this query.

Implement the Laplace Mechanism for p3b_query in the function p3b_query_dp . The function p3b_query_dp takes the following arguments:

- data: The dataset from which to calculate the Differentially Private response.
- epsilon: The \$\epsilon\$ level for Differential Privacy

The function outputs:

• diff_priv_query_answer: The \$\epsilon\$-Differentially Private response to the query using the Laplace Mechanism.

In your implementation, you should use the necessary mean \$\mu\$ and the smallest value of the parameter \$b\$ for the Laplace mechanism to ensure that the

output of this query will always be \$\epsilon\$-Differentially Private (regarless of the input database, or the conditions on the query).

Complete the following code snippet with your code:

```
In [236...

def p3b_query_dp(data, epsilon):
    # Sensitivity of the query (maximum change in query output for a single in sensitivity = 1

# Calculate scale parameter (b) for Laplace noise
b = sensitivity / epsilon

# Original query result
true_result = p3b_query(data)[0]

# Add Laplace noise to the result
laplace_noise = numpy.random.laplace(0, b)

# Ensure that the output is e-Differentially Private
diff_priv_query_answer = true_result + laplace_noise

return [diff_priv_query_answer]
```

Run the following cell to demonstrate the outcomes of your code.

The 0.2-Differentially Private output of the query is: [42.4153305032092]

(c) Histogram Queries: Laplace Mechanism

In this part, we are concerned with the query:

Query 3c: Calculate a non-normalized histogram of people with all of the following features

```
    Gender = Male
    LDL + HDL Cholesterol > 140
    Systolic Blood Pressure > 130
    Average Alcohol Consumption > 1
    Using Total Cholesterol bins provided by the user
```

which can be implemented with the following code (without any Differential Privacy)

```
In [225...
         def p3c query(data, bins):
              """Calcualte the result of Query 3c
             Parameters
             data : pandas.DataFrame
                 A data frame containing at least the following columns:
                      - Gender
                      - HDL Cholesterol
                      - LDL Cholesterol
                      - Total Cholesterol
                      - Sys Blood Pressure
                      - Alcohol_Average
             bins : list
                 The list of bins boundaries as described in numpy.histogram
             Returns
             result : list
                 a list containing the result of the query, one count for each bin desc
             query i = (data.Gender == 1) & \
                  (data.HDL_Cholesterol + Lab3_Data.LDL_Cholesterol > 140) & \
                  (data.Sys Blood Pressure > 130) & \
                  (data.Alcohol Average > 1)
             f query = numpy.histogram(data.Total Cholesterol[query i], bins=bins)[0]
             return f_query
         # Example with bins [130, 190, 250, 310, 370]
         bins = range(130,371,60)
         results = p3c query(Lab3 Data, bins)
         to_print_temp = pandas.DataFrame({"Bin Start (inclusive)": bins[:-1], "Bin End
         print_mk(f'The true output of the query on `Lab3_Data` with bins={list(bins)}
         print mk(f'{to print temp.to markdown()}')
```

The true output of the query on Lab3 Data with bins=[130, 190, 250, 310, 370] is:

	Bin Start (inclusive)	Bin End (exlusive)	Count
0	130	190	44
1	190	250	74
2	250	310	21
3	310	370	1

We would like to achieve \$\epsilon\$-Differential Privacy for this guery.

Implement the Laplace Mechanism for p3c_query in the function p3c_query_dp .

The function p3c query dp takes the following arguments:

- data: The dataset from which to calculate the Differentially Private response.
- bins: The list of bins boundaries as described in numpy.histogramThe list of bins boundaries as described in numpy.histogram()
- epsilon: The \$\epsilon\$ level for Differential Privacy

The function outputs:

• diff_priv_query_answer: The \$\epsilon\$-Differentially Private response to the query using the Laplace Mechanism.

In your implementation, you should use the necessary mean \$\mu\$ and the smallest value of the parameter \$b\$ for the Laplace mechanism to ensure that the output of this query will always be \$\epsilon\$-Differentially Private (regarless of the input database, or the conditions on the query).

Complete the following code snippet with your code:

```
In [226...

def p3c_query_dp(data, bins, epsilon):
    # Sensitivity of the query (maximum change in query output for a single in sensitivity = 1

# Calculate scale parameter (b) for Laplace noise
b = sensitivity / epsilon

# Original query result for each bin
true_results = p3c_query(data, bins)

# Add Laplace noise to each bin in the histogram
laplace_noise = numpy.random.laplace(0, b, len(bins) - 1)

# Ensure that the output is e-Differentially Private for each bin
diff_priv_query_answer = true_results + laplace_noise
```

```
return diff_priv_query_answer
```

Run the following cell to demonstrate the outcomes of your code.

The 0.2-differentially private output of the query on Lab3_Data with bins=[130, 190, 250, 310, 370] is:

	Bin Start (inclusive)	Bin End (exlusive)	Count
0	130	190	27.1676
1	190	250	77.1141
2	250	310	24.954
3	310	370	-0.248654

Problem 4

In your report, list all individuals and sources that you consulted with while working on this assignment.