```python
#functions.py

def num_in_interval(lo, hi, value, needed_increments):
    """
    Maps (normalizes) the given value from the domain of (0, (number_system_base ^ number_of_digits_of_value))
    to a value in the operational domain that coincides to the incremental position that the given
    value had in its original domain.

    Args:
        lo (Union[float,int]): min value of operational domain
        hi (Union[float,int]): max value of operational domain
        value (int): the value that is to be mapped to the operational domain
        needed_increments (int): how many possible values can be represented given the same
                                    number system, and number of digits, as the given value
    Returns:
        (float): value within the operational domain that coincides to the incremental position that
                the given value had in its original domain
    """

    #determine increment size that splits the operational domain into the number needed
    #increments of equal portion
    increment_size = (hi - lo) / needed_increments

    #return value within the operational domain that coincides to the incremental position that
    #the given value had in its original domain
    return lo + value * increment_size


def general_decoder(string, var_length, domain_min, domain_max, number_system_base):
    """
    Takes in binary string and splits it into several string variables of length var_length
    and returns a list of floating point decimal number values representing each within their
    operational domain.

    Args:
        string (str): alphanumeric string representing a number system value
        var_length (int): length of each string variable
        domain_min (Union[float,int]): min value of operational domain
        domain_max (Union[float,int]): max value of operational domain
        number_system_base (int): base of the string variable's utilized numbering system alphanumeric
                                    character set
    Returns:
        (List[float]): list of floating point decimal number values representing each of the string variables
        within their operational domain (domain_min, domain_max)
    """

    #splits string into separate variables of var_length from given string
    str_var_list = [string[i:i + var_length] for i in range(0, len(string), var_length)]

    #convert each variable from original alphanumeric character set to decimal
    dec_list = [(int(num, number_system_base)) for num in str_var_list]

    #map each decimal to a floating point number in their operational domain (domain_min, domain_max)
    max_var_val = number_system_base ** var_length
    xs = [(num_in_interval(domain_min, domain_max, dec_list[i], max_var_val)) for i in range(len(dec_list))]
```

```
56
57          return xs
58
```