3-4-5 Natural Partitioning

Importing necessary libraries

In [42]:
```
import numpy as np
import pandas as pd
```

Taking Input

In [43]:
```
X = [32, 38, 48, 91, 46, 37, 22, 69, 78, 82, 33, 49, 55, 66, 84, 86, 67, 80, 79, 44]
```

In [44]:
```
# min = int(input("Enter minimum value: "))
# max = int(input("Enter maximum value: "))

min = 0
max = 100
```

Defining the partitioning algorithm

In [45]:
```
def partitioning(nums,minimum,maximum):

  # Finding the most significant digit
  nums = sorted(nums)
  print()
  print("Numbers = {}".format(nums))
  print("Number of distinct values = {}".format(maximum - minimum))

  most_sig_digit = int(str(maximum - minimum)[0])
  print("Most Significant Digit = {}".format(most_sig_digit))


  # Finding the number of partitions
  num_of_partitions = 0
  if most_sig_digit in [3, 6, 7, 9]:
      num_of_partitions = 3
  elif most_sig_digit in [2, 4, 8]:
      num_of_partitions = 4
  else:
      num_of_partitions = 5
  print("Number of partitions decided : ", num_of_partitions)


  # Initializing the partitions dictionary and intervals list
  temp = (maximum-minimum) / num_of_partitions
  partitions = {}
  intervals = []


  # Adding partitions and intervals
  for i in range(0,num_of_partitions):
    interval_min = round(minimum + (i*temp))
    interval_max = round(minimum + (i+1)*temp)
    intervals.append((i+1,interval_min,interval_max))

    partitions[i+1] = []

  print("Partitions = {}".format(partitions))
  print("Intervals = {}".format(intervals))

  for num in nums:
    for interval in intervals:
      if num > interval[1] and num < interval[2]:
        partitions[interval[0]].append(num)

  print("Partitions = {}".format(partitions))
  print()
  return partitions,intervals
```

Partitioning for 2 Levels

In [46]:
```
print("---------------- LEVEL 1 ----------------")
partitions,intervals = partitioning(X,min,max)

print("---------------- LEVEL 2 ----------------")
final_partitions = {}
i = 1
for interval in intervals:
  partitions,intervals = partitioning(newData[interval[0]],interval[1],interval[2])
  for key in partitions.keys():
    final_partitions[i] = partitions[key]
    i += 1
```

```
---------------- LEVEL 1 ----------------

Numbers = [22, 32, 33, 37, 38, 44, 46, 48, 49, 55, 66, 67, 69, 78, 79, 80, 82, 84, 86, 91]
Number of distinct values = 100
Most Significant Digit = 1
Number of partitions decided :  5
Partitions = {1: [], 2: [], 3: [], 4: [], 5: []}
Intervals = [(1, 0, 20), (2, 20, 40), (3, 40, 60), (4, 60, 80), (5, 80, 100)]
Partitions = {1: [], 2: [22, 32, 33, 37, 38], 3: [44, 46, 48, 49, 55], 4: [66, 67, 69, 78, 79], 5: [82, 84, 86, 91]}

---------------- LEVEL 2 ----------------

Numbers = []
Number of distinct values = 20
Most Significant Digit = 2
Number of partitions decided :  4
Partitions = {1: [], 2: [], 3: [], 4: []}
Intervals = [(1, 0, 5), (2, 5, 10), (3, 10, 15), (4, 15, 20)]
Partitions = {1: [], 2: [], 3: [], 4: []}


Numbers = [22, 32, 33, 37, 38]
Number of distinct values = 20
Most Significant Digit = 2
Number of partitions decided :  4
Partitions = {1: [], 2: [], 3: [], 4: []}
Intervals = [(1, 20, 25), (2, 25, 30), (3, 30, 35), (4, 35, 40)]
Partitions = {1: [22], 2: [], 3: [32, 33], 4: [37, 38]}


Numbers = [44, 46, 48, 49, 55]
Number of distinct values = 20
Most Significant Digit = 2
Number of partitions decided :  4
Partitions = {1: [], 2: [], 3: [], 4: []}
Intervals = [(1, 40, 45), (2, 45, 50), (3, 50, 55), (4, 55, 60)]
Partitions = {1: [44], 2: [46, 48, 49], 3: [], 4: []}


Numbers = [66, 67, 69, 78, 79]
Number of distinct values = 20
Most Significant Digit = 2
Number of partitions decided :  4
Partitions = {1: [], 2: [], 3: [], 4: []}
Intervals = [(1, 60, 65), (2, 65, 70), (3, 70, 75), (4, 75, 80)]
Partitions = {1: [], 2: [66, 67, 69], 3: [], 4: [78, 79]}


Numbers = [82, 84, 86, 91]
Number of distinct values = 20
Most Significant Digit = 2
Number of partitions decided :  4
Partitions = {1: [], 2: [], 3: [], 4: []}
Intervals = [(1, 80, 85), (2, 85, 90), (3, 90, 95), (4, 95, 100)]
Partitions = {1: [82, 84], 2: [86], 3: [91], 4: []}
```

Final Answer

In [47]:
```
print("---------------- FINAL PARTITIONS ----------------")
final_partitions
```

```
---------------- FINAL PARTITIONS ----------------
```

Out[47]:
```
{1: [],
 2: [],
 3: [],
 4: [],
 5: [22],
 6: [],
 7: [32, 33],
 8: [37, 38],
 9: [44],
 10: [46, 48, 49],
 11: [],
 12: [],
 13: [],
 14: [66, 67, 69],
 15: [],
 16: [78, 79],
 17: [82, 84],
 18: [86],
 19: [91],
 20: []}
```

Replacing original values with intervals

In [52]:
```
X = sorted(X)
new_X = []
for key in final_partitions.keys():
  for i in range(len(final_partitions[key])):
    new_X.append(key)
```

In [53]:
```
new_X
```

Out[53]:
```
[5, 7, 7, 8, 8, 9, 10, 10, 10, 14, 14, 14, 16, 16, 17, 17, 18, 19]
```