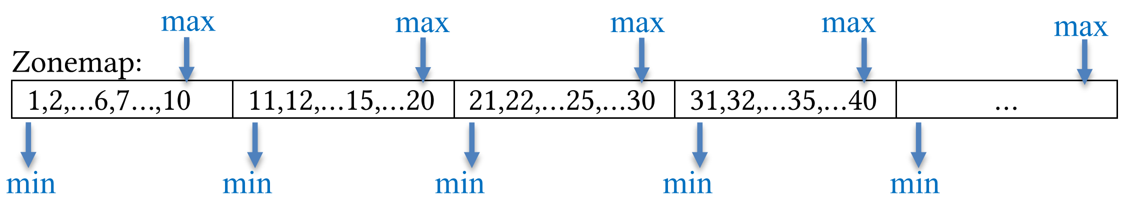
CAS CS 561: Data Systems Architectures
Data-intensive Systems and Computing Lab
Department of Computer Science
College of Arts and Sciences, Boston University
http://bu-disc.github.io/CS561/

CS561 Spring 2021 – Project 0

Title: *Implementation of a simple Zone Map*

Background: A zone map is like a coarse index that maintains minimum/maximum value ranges of one or more specified columns over contiguous sets of data blocks or rows, called zones of a table [1]. A zone map helps in data pruning of both single keys and a range of keys. The queried key/range of keys is first checked with the min/max values of every block/zone before scanning within the block, thereby reducing query latency.
Simple sorted array/vector:

1	2	3	4	5	6	…	…	…	…



Objective: The objective of the project is to implement a simple zone map and evaluate its performance on both point and range queries. The workflow for this is as the following.

(a) Implement a zone map (vanilla implementation). Develop by cloning the API available to you at: https://github.com/BU-DiSC/cs561_templatezonemaps. This API contains a header file with basic functionality definitions for a zone map. You are free to modify certain components to improve performance. Note that you are expected to build a more extensive testing infrastructure.

(b) A simple query generator (point queries) is included in main.cpp file that tests the entire domain along with a few non-existing queries. For range queries, divide the entire domain into 4 batches of 10% elements each (10-20, 30-40, 60-70, 80-90). Perform a range query on each of the batches and report the average over all the results. For testing purposes, a workload generator is also included, that generates integers in a given domain with desirable noise.

(c) Test the zone map with different workloads for both point and range queries. Use the number of elements in the domain to be 1 Million and 5 Million integers. For noise%, generate workloads of 0%, 5% and 25%. Use a standard 5% for the window threshold. The execution time outputs must be written either onto the terminal or to a log file.
Note: Noise(%) is the percentage of total elements out of order. WindowThreshold(%) is the window within which an out of order element can occur from its original position.

Deliverables: Zone map implementation code that runs the test cases. It is **required** to have comments within the implementation, that explains various design decisions.

[1] Mohamed Ziauddin, Andrew Witkowski, You Jung Kim, Dmitry Potapov, Janaki Lahorani, and Murali Krishna. 2017. Dimensions based data clustering and zone maps. Proc. VLDB Endow. 10, 12 (August 2017), 1622–1633. DOI:https://doi.org/10.14778/3137765.3137769