Project 3

• B+ Trees!
• Given an implementation of a B+ tree, you need to extend its functionality
• The B+ tree applies Alternative (3):
  • Quick reminder:

  $K^*(data\ record)$ is a $<k,\ rid-list>$ pair, where rid-list is a list of rids of data records with search key $k$
B+ Tree Structure

- Root
- Non-Leaf Nodes
- Leaf Nodes
- B-TREE_FILE
- POSTINGSFILE
- TEXTFILE
  - american beauty
  - usual suspect ...
  - american history
  - ...

The diagram illustrates the structure of a B+ tree, showing how keys (e.g., "american", "usual") are stored in non-leaf nodes and how they point to leaf nodes containing actual data (e.g., "textfile").
B+ Tree Functionalities

- With the given implementation, you have the following functionalities:

  "C" to scan the tree
  "i" to insert
  "n" to print the number of all the distinct keys
  "p" to print a btree page
  "s" to search, and print the key
  "S" to search, and print the key, posting list pairs
  "T" to print the btree in inorder format
  "x" to exit
  "f" to print, in alphabetical order, all the distinct keys between (and including) key1 and key2
  "b" to print, in reverse alphabetical order, all the distinct keys between (and including) key1 and key2
Demo!
Starter Files

- **DOC**: useful documentation of the code.
- **SRC**: source code.
- **Datafiles**: sample data to load into the tree.
- **Tests**: sample tests and their solutions.
- **README**
Getting Started

• Run the demo again!

• Study all the important data structures in *def.h*

• Read the documentation provided!