A B+ tree is a variant of a B tree

- A B tree can store both keys and records in its interior nodes
- A B+ tree stores all the records at the leaf level of the tree and keys are stored in the interior nodes

In a 2-3 tree, each interior node has either two or three children
- All the leaf nodes are at the same level
Searching

• Searching means to find whether a particular value is present in an array or not

• There are two popular methods for searching the array elements: **linear search** and **binary search**
  - The algorithm that should be used depends entirely on how the values are organized in the array
    • For example, if the elements of the array are arranged in ascending order, then binary search should be used
Linear Search

• Linear search, also called as **sequential search**, is a very simple method used for searching an array for a particular value
  
  – It works by comparing the value to be searched with every element of the array one by one in a sequence until a match is found
  
  – It is mostly used to search an unordered list of elements

```
LINEAR_SEARCH(A, N, VAL)
Step 1: [INITIALIZE] SET POS = -1
Step 2: [INITIALIZE] SET I = 1
Step 3: Repeat Step 4 while I<=N
Step 4: IF A[I] = VAL
    SET POS = I
    PRINT POS
    Go to Step 6
  [END OF IF]
  SET I = I + 1
[END OF LOOP]
Step 5: IF POS = -1
    PRINT "VALUE IS NOT PRESENT IN THE ARRAY"
[END OF IF]
Step 6: EXIT
```
Binary Search

- Binary search is a searching algorithm that works efficiently with a sorted list
  - Initially, BEG = lower_bound, END = upper_bound, and POS = MID
  - If VAL is not equal to A[MID], then the values of BEG, END, and MID will be changed depending on whether VAL is smaller or greater than A[MID]
    - If VAL < A[MID], then VAL will be present in the left segment of the array
      The value of END will be changed as END = MID – 1
    - If VAL > A[MID], then VAL will be present in the right segment of the array
      The value of BEG will be changed as BEG = MID + 1
Example

• For a data array, please find 9

\[ \text{int } A[] = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\} \]

  – Step1: BEG = 0, END = 10, MID = \(\frac{0 + 10}{2} = 5\)

  – Step2: BEG = MID + 1 = 6, END = 10, MID = \(\frac{6 + 10}{2} = 8\)

  – Step3: BEG = MID + 1 = 9, END = 10, MID = \(\frac{9 + 10}{2} = 9\)
Binary Search – Algorithm

BINARY_SEARCH(A, lower_bound, upper_bound, VAL)

Step 1: [INITIALIZE] SET BEG = lower_bound
         END = upper_bound, POS = -1

Step 2: Repeat Steps 3 and 4 while BEG <= END

Step 3: SET MID = (BEG + END)/2

Step 4: IF A[MID] = VAL
        SET POS = MID
        PRINT POS
        Go to Step 6
ELSE IF A[MID] > VAL
        SET END = MID - 1
ELSE
        SET BEG = MID + 1

[END OF IF]

[END OF LOOP]

Step 5: IF POS = -1
        PRINT “VALUE IS NOT PRESENT IN THE ARRAY”

[END OF IF]

Step 6: EXIT
Interpolation Search

- Interpolation search, also known as extrapolation search, is a searching technique that finds a specified value in a sorted array.
  - Interpolation search is similar to the binary search technique.
  - The major difference is how to select the middle value.

\[
\text{Middle} = \frac{(\text{low} + \text{high})}{2}
\]

\[
\text{Middle} = \text{low} + (\text{high} - \text{low}) \times \left( \frac{\text{key} - \text{a}[\text{low}]}{\text{a}[\text{high}] - \text{a}[\text{low}]} \right)
\]
Example

- Given a list of numbers, please search for value 19 using interpolation search technique

```plaintext
a[ ] = {1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21}
```

- Low = 0, High = 10, VAL = 19, a[Low] = 1, a[High] = 21
- Middle
  - Middle = Low + (High – Low)×((VAL – a[Low]) / (a[High] – a[Low] ))
  - = 0 + (10 – 0) × ((19 – 1) / (21 – 1) )
  - = 0 + 10 × 0.9 = 9
- a[Middle] = a[9] = 19
Interpolation Search – Algorithm

INTERPOLATION_SEARCH (A, lower_bound, upper_bound, VAL)

Step 1: [INITIALIZE] SET LOW = lower_bound,
         HIGH = upper_bound, POS = -1
Step 2: Repeat Steps 3 to 4 while LOW <= HIGH
Step 3: SET MID = LOW + (HIGH - LOW) ×
Step 4: IF VAL = A[MID]
         POS = MID
         PRINT POS
         Go to Step 6
ELSE IF VAL < A[MID]
         SET HIGH = MID - 1
ELSE
         SET LOW = MID + 1
         [END OF IF]
[END OF LOOP]
Step 5: IF POS = -1
         PRINT "VALUE IS NOT PRESENT IN THE ARRAY"
         [END OF IF]
Step 6: EXIT
Jump Search

- When we have an already sorted list, then the other efficient algorithm to search for a value is jump search or block search
  - Given an array, please find value 8

\[ a[] = \{1,2,3,4,5,6,7,8,9\} \]

Step 1: First three elements are checked. Since 3 is smaller than 8, we will have to make a jump ahead

Step 2: Next three elements are checked. Since 6 is smaller than 8, we will have to make a jump ahead

Step 3: Next three elements are checked. Since 9 is greater than 8, the desired value lies within the current boundary

Step 4: A linear search is now done to find the value in the array.
Questions?

kychen@mail.ntust.edu.tw