5-way BTree Example

Format of Data Nodes

```
TP[0]  KV[0]  TP[1]  KV[1]  TP[2]  KV[2]  TP[3]  KV[3]  TP[4]
       DRP[0]        DRP[1]        DRP[2]        DRP[3]
```

where TP is a TreePointer (ChildPtr) - pointing to another node in this tree (i.e., that node's RRN)
KV is a KeyValue – the field on which this INDEX is based (e.g., Code, SSN, WIN, Title)
DRP is a DataRecordPointer - pointing to a node in the actual main DATA File (i.e., its RRN)

NOTES:
1. This is a "conceptual view" of the tree/nodes, not the actual physical view
2. Valid KVs (and their DRPs and surrounding TPs) are always **left-justified** within a node, with empty KVs (and their DRPs & TPs) filling the right end of the node
3. `]]]` is used to indicate an "**empty**" (unused) KV with 00 for its corresponding DRP and -1 for the **next** TP. (These would always be on the right end of the node). This value is used (rather than 3 spaces) because the char `']'` is greater than any standard ASCII character code (a-z, A-Z, 0-9, etc.) (based on the standard 7-bit ASCII code table). Thus the `]]]` will thus act as a "MaxValue" in key-comparisons – assuming the program code uses an ASCII-based comparison operator (e.g., in C#, string.CompareOrdinal, but NOT Compare or CompareTo). This then eliminates the "special case" handling when searching a node.
4. BTree nodes are always at least half-full (because of the INSERT algorithm), except for the root node which may be as small as a single KV/DRP pair (with 2 TPs) with empty padding on the right.

Header Node Data: M = 05, RootPtr = 011, N = 022

NOTE: RRNs for B Tree nodes start at 1, not 0

[RRN 11] **ROOT NODE**

```
03 CON 05 IMP 10 SAT 14 ]]] -1
   18     10     06     00
```

[RRN 03] **LEVEL 2 NODE**

```
18 AND 01 ART 09 BEG 02 BUN 06
   09     07     43     44
```

[RRN 05] **LEVEL 2 NODE**

```
17 DVD 04 FIT 12 HAT 22 ]]] -1
   26     11     59     00
```

[RRN 10] **LEVEL 2 NODE**

```
08 JET 07 RAT 21 ]]] -1 ]]] -1
   17     35     00     00
```

[RRN 14] **LEVEL 2 NODE**

```
15 TON 16 USE 20 WAN 13 YOU 19
   65     64     31     30
```

5 LEAF NODES
(L-to-R children of RRN 03 node)

[RRN 18]
```
-1 ALL -1 AMY -1 ]]] -1 ]]] -1
   40     45     00     00
```

[RRN 01]
```
-1 ANN -1 ANT -1 APT -1 ARE -1
   03     04     19     01
```

[RRN 09]
```
-1 BAM -1 BAT -1 ]]] -1 ]]] -1
   42     41     00     00
```

[RRN 02]
```
-1 BOT -1 BUG -1 ]]] -1 ]]] -1
   05     02     00     00
```

[RRN 06]
```
-1 CAM -1 CAT -1 ]]] -1 ]]] -1
   20     21     00     00
```

4 LEAF NODES
(L-to-R children of RRN 05 node)

[RRN 17]
```
-1 CPU -1 DAT -1 DIP -1 DOG -1
   47     23     70     46
```

[RRN 04]
```
-1 EEK -1 EGG -1 FAN -1 FAT -1
   22     69     16     48
```

[RRN 12]
```
-1 GET -1 GIG -1 GOT -1 HAM -1
   15     24     58     68
```

[RRN 22]
```
-1 HIP -1 HOP -1 HOT -1 ICE -1
   25     57     12     67
```

3 LEAF NODES
(L-to-R children of RRN 10 node)

[RRN 08]
```
-1 INK -1 ITS -1 JAM -1 ]]] -1
   37     60     14     00
```

[RRN 07]
```
-1 NOT -1 ONE -1 OWN -1 RAM -1
   49     66     13     27
```

[RRN 21]
```
-1 RED -1 ROM -1 SAM -1 ]]] -1
   50     28     36     00
```

5 LEAF NODES
(L-to-R children of RRN 14 node)

[RRN 15]
```
-1 SUN -1 SYS -1 TAN -1 THE -1
   51     38     39     08
```

[RRN 16]
```
-1 TWO -1 USA -1 ]]] -1 ]]] -1
   34     29     00     00
```

[RRN 20]
```
-1 VAT -1 VEG -1 ]]] -1 ]]] -1
   52     53     00     00
```

[RRN 13]
```
-1 WEB -1 WOW -1 WWW -1 YES -1
   55     54     33     61
```

[RRN 19]
```
-1 YUP -1 ZAK -1 ZEN -1 ZIP -1
   56     63     32     62
```