Miguel Amigot
November 22, 2013

# AN ANALYSIS AND COMPARISON OF THE EFFICIENCY OF BINARY SEARCH TREES AND SKIP-LISTS

## ABSTRACT

The objective of this report is to investigate the performance of the binary search tree (BST) and skip-list classes. Though both classes' put(), get() and remove() methods are characterized by *O(logN)* efficiencies –which essentially divide the data to be processed by two– their performance varied for text-processing tasks.

Most notably, the binary search tree data structure performed look-up tasks more quickly and thus more efficiently than the skip-lists for all instances that entailed an unordered set of elements; namely, WarAndPeace.txt and RomeoJuliet.txt. On the other hand, an ordered set such as DICT_ZOnly.txt caused the BST to behave like a linked-list due to the fact that all elements were inserted in order. Inevitably, this reduced the BST to *O(N)* and the skip-list was found to be more efficient in this case.

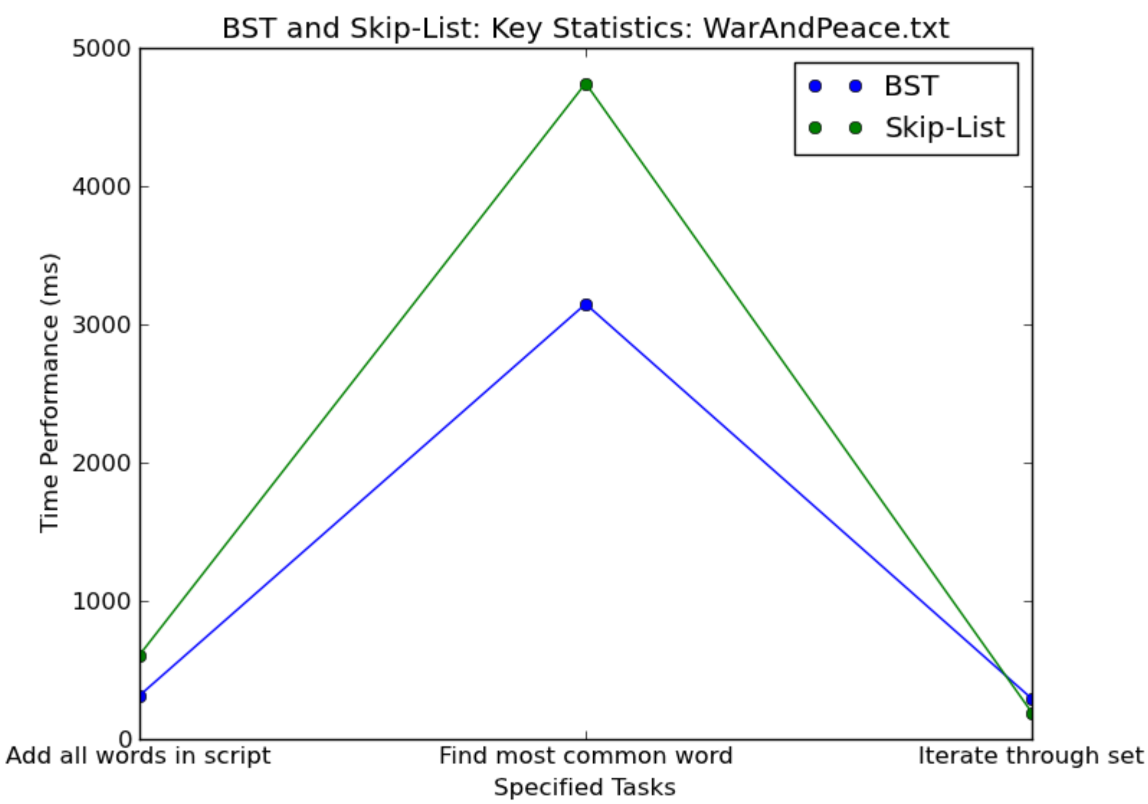
## DATA and RELEVANT GRAPHS

<u>WarAndPeace.txt</u>

*Size value: 18403 words*

| | BST | Skip-List |
|---|---|---|
| Time to add all words in script (ms) | 315 | 608 |
| Time to find most common word (ms) | 3148 | 4744 |
| Time to iterate through the set (ms) | 292 | 192 |

**Figure 1**

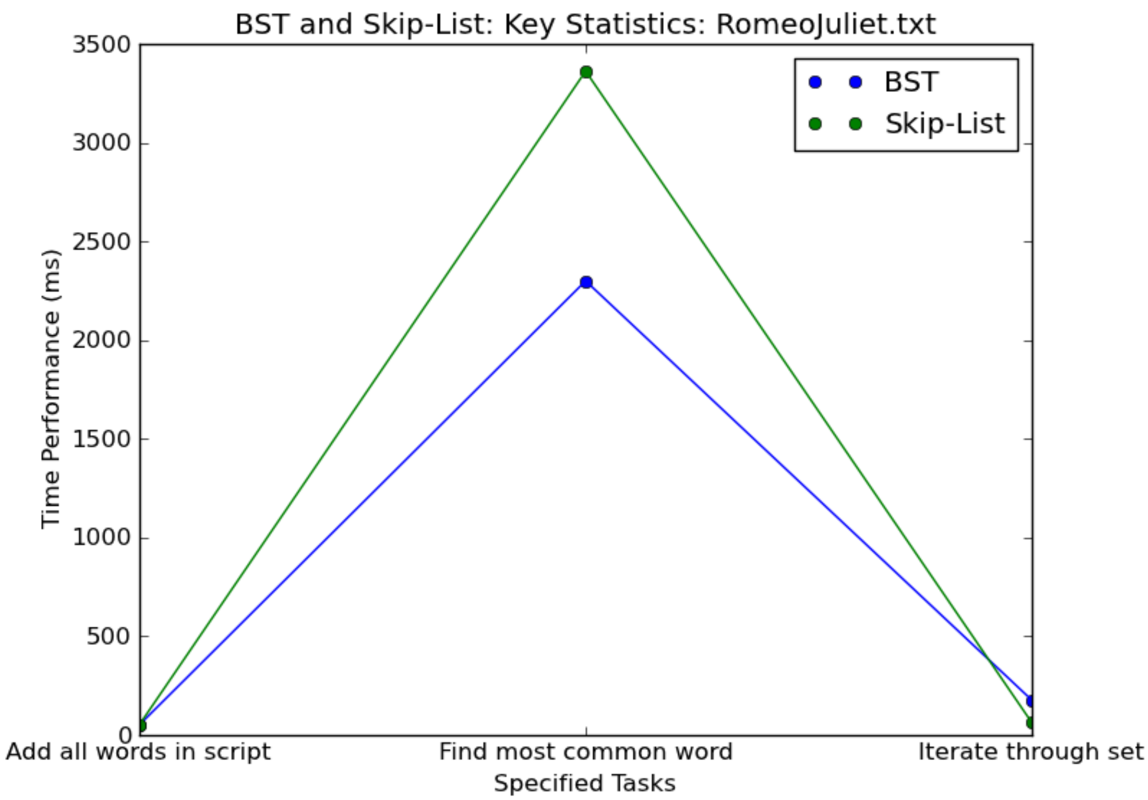


__RomeoJuliet.txt__

*Size value: 3763 words*

| | BST | Skip-List |
|---|---|---|
| Time to add all words in script (ms) | 52 | 53 |
| Time to find most common word (ms) | 2300 | 3362 |
| Time to iterate through the set (ms) | 177 | 63 |

Figure 2

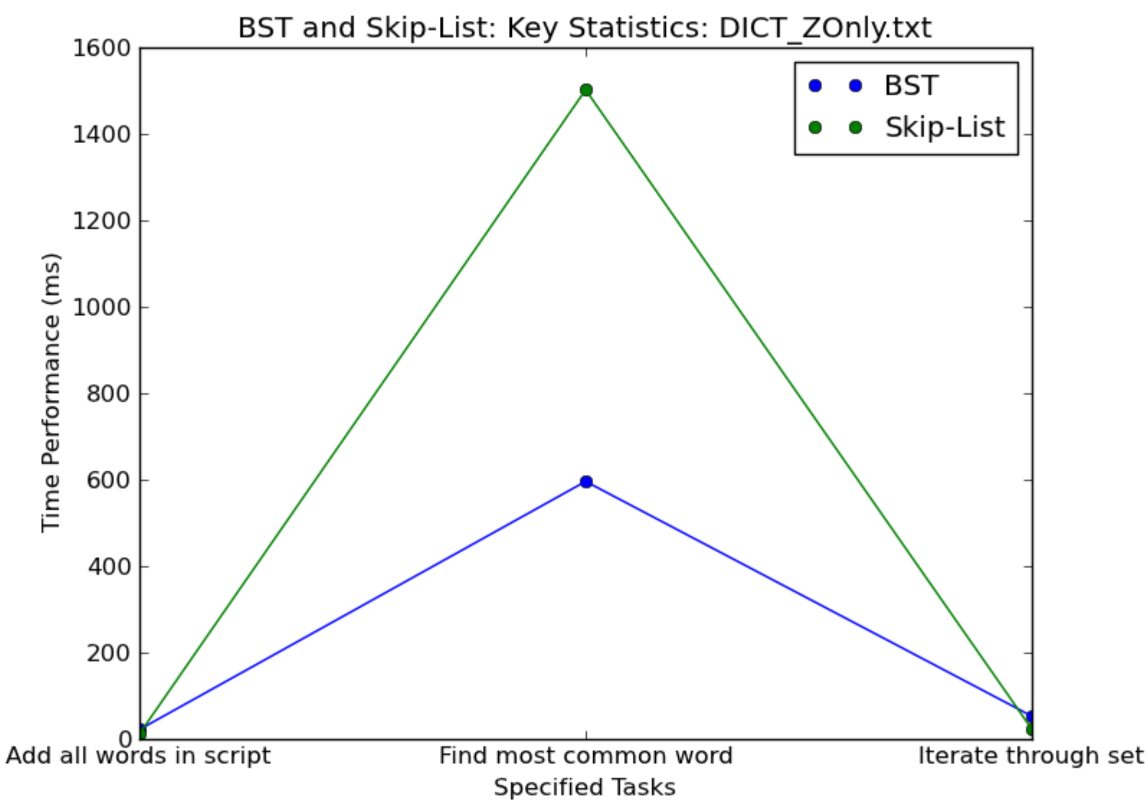


**DICT_ZOnly.txt**

*Size value: 601 words*

| | BST | Skip-List |
|---|---|---|
| Time to add all words in script (ms) | 22 | 12 |
| Time to find most common word (ms) | 596 | 1502 |
| Time to iterate through the set (ms) | 53 | 23 |

**Figure 3**



*See the end of the report for the BST's and skip-list's additional data, detailing the number of steps to reach an element vs. the number of elements reachable in that number of steps for WarAndPeace.txt, RomeoJuliet.txt and DICT_ZOnly.txt. The following graphs correspond to this data.*

**Figure 4**

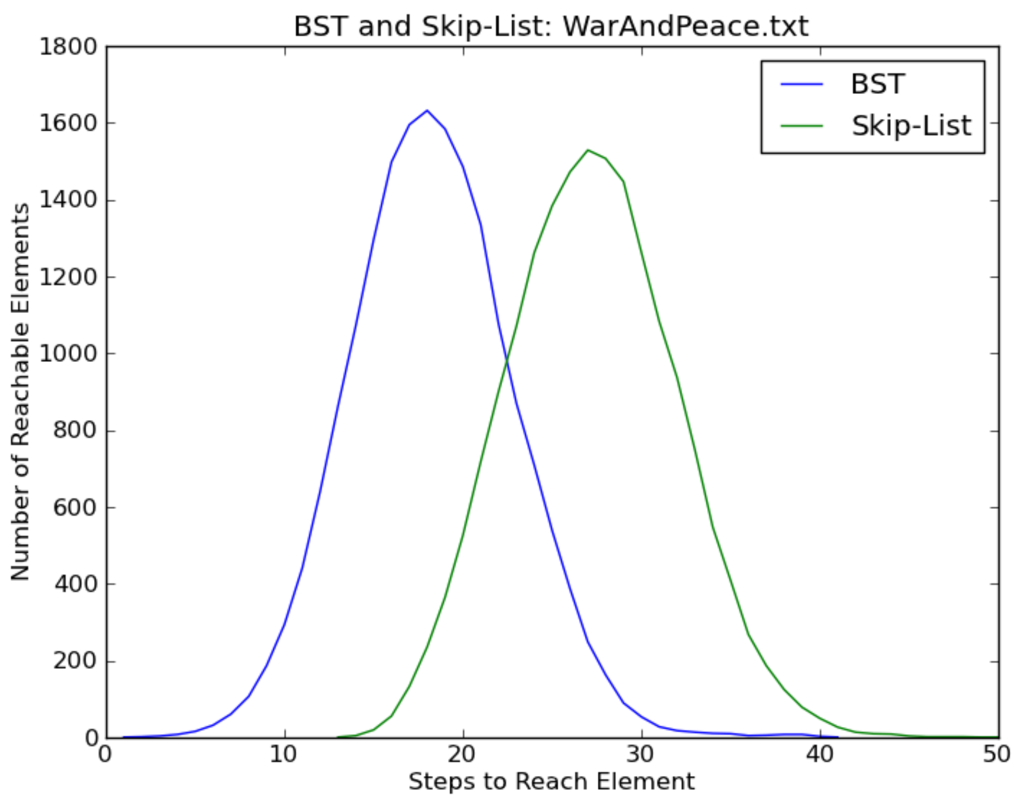


**Figure 5**

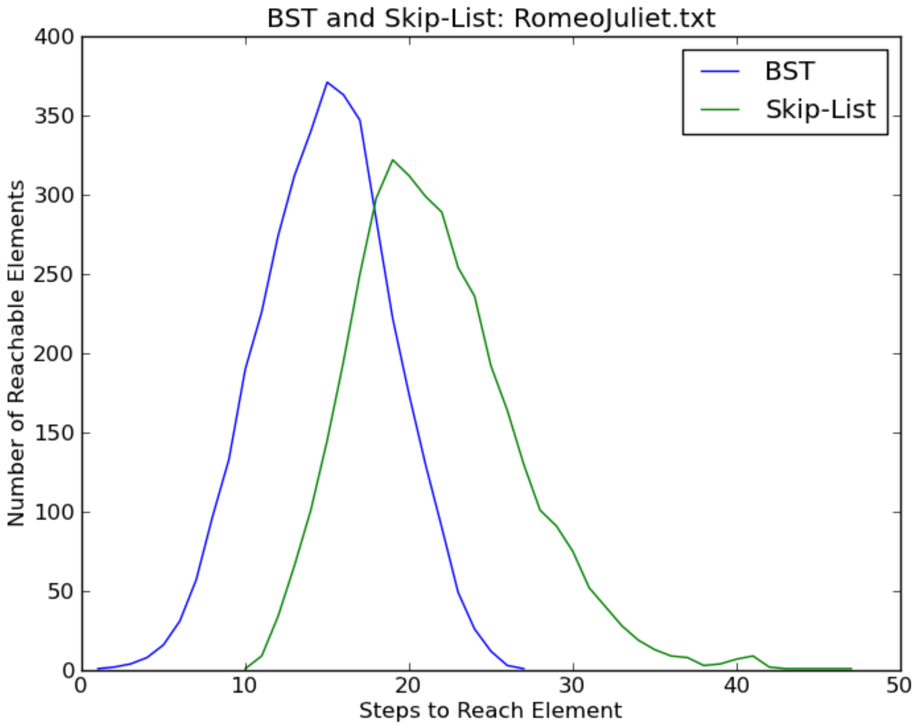
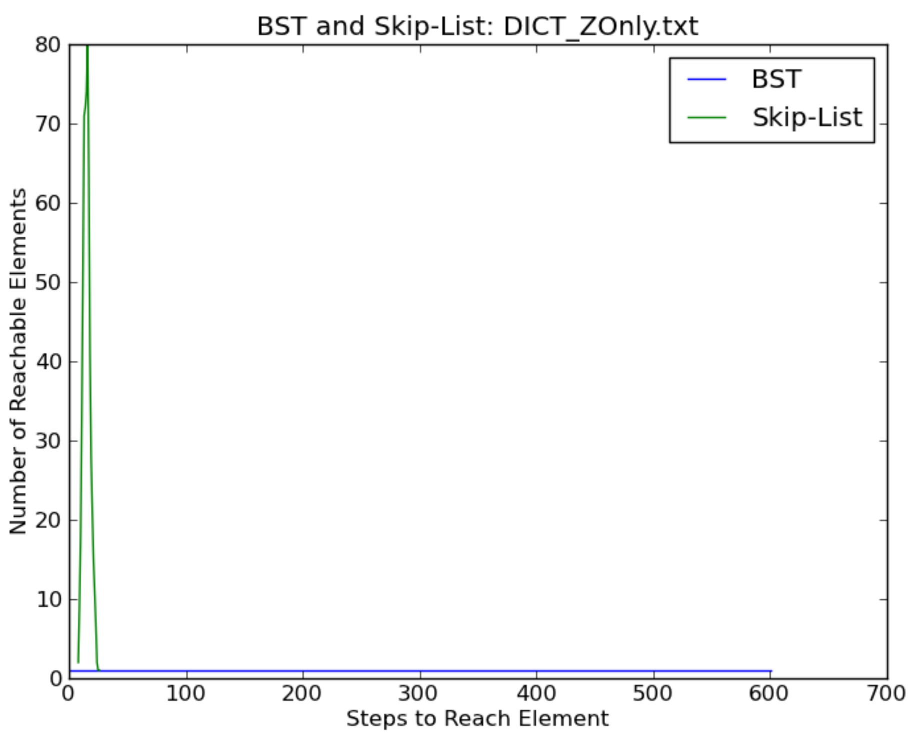



**Figure 6**



## DISCUSSION

The objective of this report was to investigate the respective efficiencies of the binary search tree and skip-list classes. As the data suggests, the former data structure was slightly more efficient than the latter for all cases in which the inserted words were "random" or unordered (for WarAndPeace.txt and RomeoJuliet.txt, but not for DICT_ZOnly.txt). That is because the BST becomes no different from a linked-list of efficiency *O(N)* when elements are inserted in order.

As Fig. 1 highlights, the most drastic difference in performance between the BST and the skip-list comes from the time they took to find the most common word in their respective structures: 3148*ms* for the BST and 4744*ms* for the skip-list. This trend is supported by Fig. 2 –with 2300*ms* for the BST and 3362*ms* for the skip-list– and by Fig. 3 –with 596*ms* for the former and 1502*ms* for the latter. Naturally, the generally-decreasing trend for both data structures comes from the fact that the parsed text files' sizes are in decreasing order.

Figs. 1-2 additionally show that the time required to add the words in the script to the data structures was higher for the skip-list than for the BST with WarAndPeace.txt (608*ms* and 315*ms*, respectively) and virtually similar with RomeoJuliet.txt (53*ms* and 52*ms*, respectively). Fig. 3 shows that, for an ordered text file such as DICT_ZOnly.txt, the skip-list was almost twice as fast as the BST (12*ms* vs. 22*ms*) –which further evidences the latter's inevitability to act as a linked-list when used to store elements in sorted order.

Figs. 1-3 also show that the skip-list structure was able to traverse more quickly than the binary search tree through its nodes –192*ms* vs. 292*ms* for WarAndPeace.txt, 63*ms* vs. 177*ms* for RomeoJuliet.txt and 23*ms* vs. 53*ms* for DICT_ZOnly.txt. This statistic, which is able to be performed no faster than in *O(N)* time, highlights each class's speed when performing a linear traversal.

The runtime of the data structures, which is additionally portrayed by Figs. 4-6, is once again shown to be more efficient for the BST than for the skip-list when elements are inserted in "random" order (meaning that the words in WarAndPeace.txt and RomeoJuliet.txt follow no particular order that is recognizable by the Java program). This is most likely due to the mathematical nature and imperfection of skip-lists, which certainly do not guarantee even spacing between nodes of different levels. The following image from University of Maryland's CMSC 420's "Lecture 11: Skip Lists" details this reality:

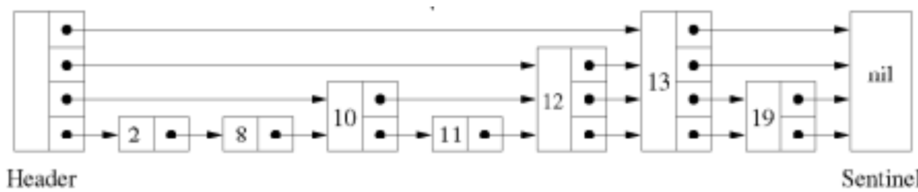



**Image 1: Randomized Skip-List**

Since the chances of new nodes' levels incrementing by one are always 50 percent, the likelihood that a skip-list of *n* elements will have its highest-leveled node –which is as high as the size of the header array of size *s* in the picture above– at exactly the middle is $\left(\frac{1}{n}\right)\left(\frac{1}{2}\right)^{s}$. If the skip-list contains 18403 items as it did for WarAndPeace.txt and the size of the header array is 32, the chances of the highest-possible node being at exactly the center would be equal to $\left(\frac{1}{18403}\right)\left(\frac{1}{2}\right)^{32} \approx 1.27\times10^{-14}$ or $1.27\times10^{-12}$ %. Naturally, the size and arrangement of the rest of the inner nodes must also adhere to the ideal conditions, which would evidently perform all operations in *O(logN)* time. Given that it is virtually impossible for a randomized skip-list of less than infinity number of elements to meet ideal conditions, this trait undoubtedly led to a performance that was poorer than expected. Of course we cannot expect a binary search tree to be perfectly balanced either; therefore this adds to our uncertainty and the expected deviation from *O(logN)* behavior.

The way in which the binary search tree takes advantage of *O(logN)* is clear –each search or deletion task's complexity is virtually reduced by a factor of two for every iteration, as the relevant method is called on a tree that is half as large. The complexity analysis is much more subtle for skip-lists, which ultimately owe their *O(logN)* nature to the fact that the probability that their nodes' levels will increment is 50 percent. Greater or lower probabilities would approximate *O(N)* behavior as too many –or too few– links would be established across nodes through their arrays.

Aside from the data gathered through DICT_ZOnly.txt, I did not expect the skip-list to be less efficient by requiring more steps than the binary search trees to find specified elements (see Figs. 4-5). Nevertheless, this is reasonable due to the mentioned lack of "ideal behavior" that the skip-list displays.

**ADDITIONAL DATA**

***Number of steps to reach an element (left column) vs. number of elements reachable in that number of steps (right column).***

**<u>Binary Search Tree: WarAndPeace.txt</u>**

| | |
|---|---|
| 1 | 1 |
| 2 | 2 |
| 3 | 4 |
| 4 | 8 |
| 5 | 16 |
| 6 | 32 |
| 7 | 61 |
| 8 | 107 |
| 9 | 187 |
| 10 | 294 |
| 11 | 441 |
| 12 | 640 |
| 13 | 862 |
| 14 | 1072 |
| 15 | 1296 |
| 16 | 1498 |
| 17 | 1595 |
| 18 | 1632 |
| 19 | 1584 |
| 20 | 1486 |
| 21 | 1335 |
| 22 | 1076 |
| 23 | 870 |
| 24 | 709 |
| 25 | 539 |
| 26 | 388 |
| 27 | 249 |
| 28 | 163 |
| 29 | 90 |
| 30 | 54 |
| 31 | 28 |
| 32 | 18 |
| 33 | 14 |
| 34 | 11 |
| 35 | 10 |
| 36 | 5 |
| 37 | 6 |
| 38 | 8 |
| 39 | 8 |

40  3
41  1

**Binary Search Tree: RomeoJuliet.txt**

1  1
2  2
3  4
4  8
5  16
6  31
7  57
8  97
9  133
10  190
11  226
12  274
13  312
14  340
15  371
16  363
17  347
18  284
19  222
20  174
21  130
22  90
23  49
24  26
25  12
26  3
27  1

**Binary Search Tree: DICT_ZOnly.txt**

1  1
2  1
3  1
4  1
5  1
6  1
7  1
8  1
9  1
10  1

| | |
|---|---|
| 11 | 1 |
| 12 | 1 |
| 13 | 1 |
| 14 | 1 |
| 15 | 1 |
| 16 | 1 |
| 17 | 1 |
| 18 | 1 |
| 19 | 1 |
| 20 | 1 |
| 21 | 1 |
| 22 | 1 |
| 23 | 1 |
| 24 | 1 |
| 25 | 1 |
| 26 | 1 |
| 27 | 1 |
| 28 | 1 |
| 29 | 1 |
| 30 | 1 |
| 31 | 1 |
| 32 | 1 |
| 33 | 1 |
| 34 | 1 |
| 35 | 1 |
| 36 | 1 |
| 37 | 1 |
| 38 | 1 |
| 39 | 1 |
| 40 | 1 |
| 41 | 1 |
| 42 | 1 |
| 43 | 1 |
| 44 | 1 |
| 45 | 1 |
| 46 | 1 |
| 47 | 1 |
| 48 | 1 |
| 49 | 1 |
| 50 | 1 |
| 51 | 1 |
| 52 | 1 |
| 53 | 1 |
| 54 | 1 |

| | |
|---|---|
| 55 | 1 |
| 56 | 1 |
| 57 | 1 |
| 58 | 1 |
| 59 | 1 |
| 60 | 1 |
| 61 | 1 |
| 62 | 1 |
| 63 | 1 |
| 64 | 1 |
| 65 | 1 |
| 66 | 1 |
| 67 | 1 |
| 68 | 1 |
| 69 | 1 |
| 70 | 1 |
| 71 | 1 |
| 72 | 1 |
| 73 | 1 |
| 74 | 1 |
| 75 | 1 |
| 76 | 1 |
| 77 | 1 |
| 78 | 1 |
| 79 | 1 |
| 80 | 1 |
| 81 | 1 |
| 82 | 1 |
| 83 | 1 |
| 84 | 1 |
| 85 | 1 |
| 86 | 1 |
| 87 | 1 |
| 88 | 1 |
| 89 | 1 |
| 90 | 1 |
| 91 | 1 |
| 92 | 1 |
| 93 | 1 |
| 94 | 1 |
| 95 | 1 |
| 96 | 1 |
| 97 | 1 |
| 98 | 1 |

| | |
|---|---|
| 99 | 1 |
| 100 | 1 |
| 101 | 1 |
| 102 | 1 |
| 103 | 1 |
| 104 | 1 |
| 105 | 1 |
| 106 | 1 |
| 107 | 1 |
| 108 | 1 |
| 109 | 1 |
| 110 | 1 |
| 111 | 1 |
| 112 | 1 |
| 113 | 1 |
| 114 | 1 |
| 115 | 1 |
| 116 | 1 |
| 117 | 1 |
| 118 | 1 |
| 119 | 1 |
| 120 | 1 |
| 121 | 1 |
| 122 | 1 |
| 123 | 1 |
| 124 | 1 |
| 125 | 1 |
| 126 | 1 |
| 127 | 1 |
| 128 | 1 |
| 129 | 1 |
| 130 | 1 |
| 131 | 1 |
| 132 | 1 |
| 133 | 1 |
| 134 | 1 |
| 135 | 1 |
| 136 | 1 |
| 137 | 1 |
| 138 | 1 |
| 139 | 1 |
| 140 | 1 |
| 141 | 1 |
| 142 | 1 |

| | |
|---|---|
| 143 | 1 |
| 144 | 1 |
| 145 | 1 |
| 146 | 1 |
| 147 | 1 |
| 148 | 1 |
| 149 | 1 |
| 150 | 1 |
| 151 | 1 |
| 152 | 1 |
| 153 | 1 |
| 154 | 1 |
| 155 | 1 |
| 156 | 1 |
| 157 | 1 |
| 158 | 1 |
| 159 | 1 |
| 160 | 1 |
| 161 | 1 |
| 162 | 1 |
| 163 | 1 |
| 164 | 1 |
| 165 | 1 |
| 166 | 1 |
| 167 | 1 |
| 168 | 1 |
| 169 | 1 |
| 170 | 1 |
| 171 | 1 |
| 172 | 1 |
| 173 | 1 |
| 174 | 1 |
| 175 | 1 |
| 176 | 1 |
| 177 | 1 |
| 178 | 1 |
| 179 | 1 |
| 180 | 1 |
| 181 | 1 |
| 182 | 1 |
| 183 | 1 |
| 184 | 1 |
| 185 | 1 |
| 186 | 1 |

| | |
|---|---|
| 187 | 1 |
| 188 | 1 |
| 189 | 1 |
| 190 | 1 |
| 191 | 1 |
| 192 | 1 |
| 193 | 1 |
| 194 | 1 |
| 195 | 1 |
| 196 | 1 |
| 197 | 1 |
| 198 | 1 |
| 199 | 1 |
| 200 | 1 |
| 201 | 1 |
| 202 | 1 |
| 203 | 1 |
| 204 | 1 |
| 205 | 1 |
| 206 | 1 |
| 207 | 1 |
| 208 | 1 |
| 209 | 1 |
| 210 | 1 |
| 211 | 1 |
| 212 | 1 |
| 213 | 1 |
| 214 | 1 |
| 215 | 1 |
| 216 | 1 |
| 217 | 1 |
| 218 | 1 |
| 219 | 1 |
| 220 | 1 |
| 221 | 1 |
| 222 | 1 |
| 223 | 1 |
| 224 | 1 |
| 225 | 1 |
| 226 | 1 |
| 227 | 1 |
| 228 | 1 |
| 229 | 1 |
| 230 | 1 |

| | |
|---|---|
| 231 | 1 |
| 232 | 1 |
| 233 | 1 |
| 234 | 1 |
| 235 | 1 |
| 236 | 1 |
| 237 | 1 |
| 238 | 1 |
| 239 | 1 |
| 240 | 1 |
| 241 | 1 |
| 242 | 1 |
| 243 | 1 |
| 244 | 1 |
| 245 | 1 |
| 246 | 1 |
| 247 | 1 |
| 248 | 1 |
| 249 | 1 |
| 250 | 1 |
| 251 | 1 |
| 252 | 1 |
| 253 | 1 |
| 254 | 1 |
| 255 | 1 |
| 256 | 1 |
| 257 | 1 |
| 258 | 1 |
| 259 | 1 |
| 260 | 1 |
| 261 | 1 |
| 262 | 1 |
| 263 | 1 |
| 264 | 1 |
| 265 | 1 |
| 266 | 1 |
| 267 | 1 |
| 268 | 1 |
| 269 | 1 |
| 270 | 1 |
| 271 | 1 |
| 272 | 1 |
| 273 | 1 |
| 274 | 1 |

| | |
|---|---|
| 275 | 1 |
| 276 | 1 |
| 277 | 1 |
| 278 | 1 |
| 279 | 1 |
| 280 | 1 |
| 281 | 1 |
| 282 | 1 |
| 283 | 1 |
| 284 | 1 |
| 285 | 1 |
| 286 | 1 |
| 287 | 1 |
| 288 | 1 |
| 289 | 1 |
| 290 | 1 |
| 291 | 1 |
| 292 | 1 |
| 293 | 1 |
| 294 | 1 |
| 295 | 1 |
| 296 | 1 |
| 297 | 1 |
| 298 | 1 |
| 299 | 1 |
| 300 | 1 |
| 301 | 1 |
| 302 | 1 |
| 303 | 1 |
| 304 | 1 |
| 305 | 1 |
| 306 | 1 |
| 307 | 1 |
| 308 | 1 |
| 309 | 1 |
| 310 | 1 |
| 311 | 1 |
| 312 | 1 |
| 313 | 1 |
| 314 | 1 |
| 315 | 1 |
| 316 | 1 |
| 317 | 1 |
| 318 | 1 |

| | |
|---|---|
| 319 | 1 |
| 320 | 1 |
| 321 | 1 |
| 322 | 1 |
| 323 | 1 |
| 324 | 1 |
| 325 | 1 |
| 326 | 1 |
| 327 | 1 |
| 328 | 1 |
| 329 | 1 |
| 330 | 1 |
| 331 | 1 |
| 332 | 1 |
| 333 | 1 |
| 334 | 1 |
| 335 | 1 |
| 336 | 1 |
| 337 | 1 |
| 338 | 1 |
| 339 | 1 |
| 340 | 1 |
| 341 | 1 |
| 342 | 1 |
| 343 | 1 |
| 344 | 1 |
| 345 | 1 |
| 346 | 1 |
| 347 | 1 |
| 348 | 1 |
| 349 | 1 |
| 350 | 1 |
| 351 | 1 |
| 352 | 1 |
| 353 | 1 |
| 354 | 1 |
| 355 | 1 |
| 356 | 1 |
| 357 | 1 |
| 358 | 1 |
| 359 | 1 |
| 360 | 1 |
| 361 | 1 |
| 362 | 1 |

| | |
|---|---|
| 363 | 1 |
| 364 | 1 |
| 365 | 1 |
| 366 | 1 |
| 367 | 1 |
| 368 | 1 |
| 369 | 1 |
| 370 | 1 |
| 371 | 1 |
| 372 | 1 |
| 373 | 1 |
| 374 | 1 |
| 375 | 1 |
| 376 | 1 |
| 377 | 1 |
| 378 | 1 |
| 379 | 1 |
| 380 | 1 |
| 381 | 1 |
| 382 | 1 |
| 383 | 1 |
| 384 | 1 |
| 385 | 1 |
| 386 | 1 |
| 387 | 1 |
| 388 | 1 |
| 389 | 1 |
| 390 | 1 |
| 391 | 1 |
| 392 | 1 |
| 393 | 1 |
| 394 | 1 |
| 395 | 1 |
| 396 | 1 |
| 397 | 1 |
| 398 | 1 |
| 399 | 1 |
| 400 | 1 |
| 401 | 1 |
| 402 | 1 |
| 403 | 1 |
| 404 | 1 |
| 405 | 1 |
| 406 | 1 |

| | |
|---|---|
| 407 | 1 |
| 408 | 1 |
| 409 | 1 |
| 410 | 1 |
| 411 | 1 |
| 412 | 1 |
| 413 | 1 |
| 414 | 1 |
| 415 | 1 |
| 416 | 1 |
| 417 | 1 |
| 418 | 1 |
| 419 | 1 |
| 420 | 1 |
| 421 | 1 |
| 422 | 1 |
| 423 | 1 |
| 424 | 1 |
| 425 | 1 |
| 426 | 1 |
| 427 | 1 |
| 428 | 1 |
| 429 | 1 |
| 430 | 1 |
| 431 | 1 |
| 432 | 1 |
| 433 | 1 |
| 434 | 1 |
| 435 | 1 |
| 436 | 1 |
| 437 | 1 |
| 438 | 1 |
| 439 | 1 |
| 440 | 1 |
| 441 | 1 |
| 442 | 1 |
| 443 | 1 |
| 444 | 1 |
| 445 | 1 |
| 446 | 1 |
| 447 | 1 |
| 448 | 1 |
| 449 | 1 |
| 450 | 1 |

| | |
|---|---|
| 451 | 1 |
| 452 | 1 |
| 453 | 1 |
| 454 | 1 |
| 455 | 1 |
| 456 | 1 |
| 457 | 1 |
| 458 | 1 |
| 459 | 1 |
| 460 | 1 |
| 461 | 1 |
| 462 | 1 |
| 463 | 1 |
| 464 | 1 |
| 465 | 1 |
| 466 | 1 |
| 467 | 1 |
| 468 | 1 |
| 469 | 1 |
| 470 | 1 |
| 471 | 1 |
| 472 | 1 |
| 473 | 1 |
| 474 | 1 |
| 475 | 1 |
| 476 | 1 |
| 477 | 1 |
| 478 | 1 |
| 479 | 1 |
| 480 | 1 |
| 481 | 1 |
| 482 | 1 |
| 483 | 1 |
| 484 | 1 |
| 485 | 1 |
| 486 | 1 |
| 487 | 1 |
| 488 | 1 |
| 489 | 1 |
| 490 | 1 |
| 491 | 1 |
| 492 | 1 |
| 493 | 1 |
| 494 | 1 |

| | |
|---|---|
| 495 | 1 |
| 496 | 1 |
| 497 | 1 |
| 498 | 1 |
| 499 | 1 |
| 500 | 1 |
| 501 | 1 |
| 502 | 1 |
| 503 | 1 |
| 504 | 1 |
| 505 | 1 |
| 506 | 1 |
| 507 | 1 |
| 508 | 1 |
| 509 | 1 |
| 510 | 1 |
| 511 | 1 |
| 512 | 1 |
| 513 | 1 |
| 514 | 1 |
| 515 | 1 |
| 516 | 1 |
| 517 | 1 |
| 518 | 1 |
| 519 | 1 |
| 520 | 1 |
| 521 | 1 |
| 522 | 1 |
| 523 | 1 |
| 524 | 1 |
| 525 | 1 |
| 526 | 1 |
| 527 | 1 |
| 528 | 1 |
| 529 | 1 |
| 530 | 1 |
| 531 | 1 |
| 532 | 1 |
| 533 | 1 |
| 534 | 1 |
| 535 | 1 |
| 536 | 1 |
| 537 | 1 |
| 538 | 1 |

| | |
|---|---|
| 539 | 1 |
| 540 | 1 |
| 541 | 1 |
| 542 | 1 |
| 543 | 1 |
| 544 | 1 |
| 545 | 1 |
| 546 | 1 |
| 547 | 1 |
| 548 | 1 |
| 549 | 1 |
| 550 | 1 |
| 551 | 1 |
| 552 | 1 |
| 553 | 1 |
| 554 | 1 |
| 555 | 1 |
| 556 | 1 |
| 557 | 1 |
| 558 | 1 |
| 559 | 1 |
| 560 | 1 |
| 561 | 1 |
| 562 | 1 |
| 563 | 1 |
| 564 | 1 |
| 565 | 1 |
| 566 | 1 |
| 567 | 1 |
| 568 | 1 |
| 569 | 1 |
| 570 | 1 |
| 571 | 1 |
| 572 | 1 |
| 573 | 1 |
| 574 | 1 |
| 575 | 1 |
| 576 | 1 |
| 577 | 1 |
| 578 | 1 |
| 579 | 1 |
| 580 | 1 |
| 581 | 1 |
| 582 | 1 |

| | |
|---|---|
| 583 | 1 |
| 584 | 1 |
| 585 | 1 |
| 586 | 1 |
| 587 | 1 |
| 588 | 1 |
| 589 | 1 |
| 590 | 1 |
| 591 | 1 |
| 592 | 1 |
| 593 | 1 |
| 594 | 1 |
| 595 | 1 |
| 596 | 1 |
| 597 | 1 |
| 598 | 1 |
| 599 | 1 |
| 600 | 1 |
| 601 | 1 |

**Skip-List: WarAndPeace.txt**

| | |
|---|---|
| 13 | 1 |
| 14 | 5 |
| 15 | 20 |
| 16 | 56 |
| 17 | 133 |
| 18 | 236 |
| 19 | 365 |
| 20 | 526 |
| 21 | 718 |
| 22 | 901 |
| 23 | 1071 |
| 24 | 1262 |
| 25 | 1384 |
| 26 | 1472 |
| 27 | 1529 |
| 28 | 1507 |
| 29 | 1447 |
| 30 | 1263 |
| 31 | 1083 |
| 32 | 937 |

| | |
|---|---|
| 33 | 749 |
| 34 | 548 |
| 35 | 409 |
| 36 | 268 |
| 37 | 187 |
| 38 | 125 |
| 39 | 79 |
| 40 | 50 |
| 41 | 27 |
| 42 | 14 |
| 43 | 10 |
| 44 | 9 |
| 45 | 4 |
| 46 | 2 |
| 47 | 2 |
| 48 | 2 |
| 49 | 1 |
| 50 | 1 |

**Skip-List: RomeoJuliet.txt**

| | |
|---|---|
| 10 | 1 |
| 11 | 9 |
| 12 | 34 |
| 13 | 66 |
| 14 | 101 |
| 15 | 145 |
| 16 | 195 |
| 17 | 250 |
| 18 | 298 |
| 19 | 322 |
| 20 | 312 |
| 21 | 299 |
| 22 | 289 |
| 23 | 254 |
| 24 | 236 |
| 25 | 192 |
| 26 | 164 |
| 27 | 130 |

| | |
|---|---|
| 28 | 101 |
| 29 | 91 |
| 30 | 75 |
| 31 | 52 |
| 32 | 40 |
| 33 | 28 |
| 34 | 19 |
| 35 | 13 |
| 36 | 9 |
| 37 | 8 |
| 38 | 3 |
| 39 | 4 |
| 40 | 7 |
| 41 | 9 |
| 42 | 2 |
| 43 | 1 |
| 44 | 1 |
| 45 | 1 |
| 46 | 1 |
| 47 | 1 |

**Skip-List: DICT_ZOnly.txt**

| | |
|---|---|
| 8 | 2 |
| 9 | 9 |
| 10 | 18 |
| 11 | 33 |
| 12 | 51 |
| 13 | 71 |
| 14 | 72 |
| 15 | 74 |
| 16 | 80 |
| 17 | 64 |
| 18 | 41 |
| 19 | 28 |
| 20 | 21 |
| 21 | 15 |
| 22 | 11 |
| 23 | 7 |

| | |
|---|---|
| 24 | 2 |
| 25 | 1 |
| 26 | 1 |