

Unit 5 Programming Assignment 1 [50 points]

Inverted File

An inverted file or inverted index is a useful data structure for maintaining the index of a book. Any book *B*, which can be viewed as an unordered list of words, has an associated index, *I*, which maintains a lexicographically ordered list of words such that each word, *w*, in *I*, is paired with the associated page number(s) in *B* where *w* appears. Implement an efficient algorithm for constructing *I* from *B* using the built-in (i.e., standard library) map or dictionary data structure from your preferred programming language (i.e., python, C++, or Java).

First, we must consider how we will transform *B* into *I*. *I* is a map or dictionary such that the keys are the distinct words in *B* and the values will be a set of the associated page number(s) where the words appear. Observe the following page from the index of an algorithms textbook (Figure 1) and the respective key, value pairings where each key is a unique word and the set of values associated with the word are the page number(s) where the word appears (e.g., double hashing, 419; CPU, 111; edge, 302, 620). We will use a set data structure to store the page numbers. Since a set does not repeat replicates, it would be the best option for storing the page numbers. Use the built-in set data structure from your preferred programming language to store the values of the associated keys for *I*.

Index	
Cormen, Thomas, 535, 696	greedy method, 603
Counter class, 450	position, 279–281
CPU, 111	prune-and-search, 571–573
CRC cards, 63	template method, 93, 342, 406, 448, 478
CreditCard class, 63, 69–73, 73, 83–86	DFS, <i>see</i> depth-first search
Crochemore, Maxime, 618	Di Battista, Giuseppe, 361, 696
cryptography, 216–218	diameter, 358
ctypes module, 191, 195	dict class, 7, 11, 402
cubic function, 119	dictionary, 11, 16, 402–408, <i>see also</i> map
cyber-dollar, 197–199, 497–500, 682	dictionary comprehension, 43
cycle, 623	Dijkstra's algorithm, 661–669
directed, 623	Dijkstra, Edsger, 182, 696
cyclic-shift hash code, 413–414	dir function, 46
DAG, <i>see</i> directed acyclic graph	directed acyclic graph, 655–657
data packets, 227	disk usage, 157–160, 163–164, 340
data structure, 110	divide-and-conquer, 538–542, 550–551
Dawson, Michael, 55	division method for hash codes, 416
debugging, 62	documentation, 66
decision tree, 311, 463, 562	double hashing, 419
decorate-sort-undecorate design pattern, 570	double-ended queue, <i>see</i> deque
decrease-and-conquer, 571	doubly linked list, 260, 270–276
decryption, 216	„DoublyLinkedBase class, 273–275, 278
deep copy, 102, 188	down-heap bubbling, 374
deepcopy function, 102, 188	duck typing, 60, 306
def keyword, 23	dynamic array, 192–201, 246
degree of a vertex, 621	shrinking, 200, 246
del operator, 15, 75	DynamicArray class, 195–196, 204, 206, 224, 225, 245
DeMorgan's Law, 137	dynamic binding, 100
Demurjian, Steven, 108, 254	dynamic dispatch, 100
depth of a tree, 308–310	dynamic programming, 594–600
depth-first search (DFS), 639–647	dynamically typed, 5
deque, 247–249	Eades, Peter, 361, 696
abstract data type, 247–248	edge, 302, 620
linked-list implementation, 249, 275	destination, 621
deque class, 249, 251	endpoint, 621
descendant, 302	incident, 621
design patterns, v, 61	multiple, 622
adapter, 231	origin, 621
amortization, 197–200	outgoing, 621
brute force, 584	parallel, 622
composition, 287, 365, 407	self-loop, 622
divide-and-conquer, 538–542, 550–551	edge list, 627–629
dynamic programming, 594–600	edge relaxation, 661
factory method, 479	edit distance, 616

Figure 1

For practical purposes let's use the file "*Orwell 1984.txt*" and assume each page has 250 words.

Write a program that will create the index, I, for this book and write the index to a file named "*Orwell 1984 Index.rtf*" so that the format is similar to the example index above. **[40 points]**

Include functionality testing of the index, I, which includes a lookup function for a desired key/word which returns the set of pages associated with that word. If the word is not found, an appropriate message will be returned. **[10 points]**