

DAA LAB-5

Task 1:

Algorithm:

csv file format

ST. No , ShelfLife, Cost, Capacity

1/ Input: Taken from csv file with above

file - path = dataset csv

dataset = readcsv (file-path)

Assumption: Max weigh capacity = 200 tonnes

Fractional Knapsack and Jo MA

Readcsv (file dataset)

// Input: CSV fire dataset

11 output: creates a list of dictionaries of the

return detaset. to dict (prient = records")

Fractional Knapsack (items, W)

// Input: D'List of Lictionaries of CSV file (items)
and max capacity (w)

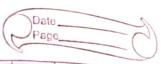
Moutput: Maximum wash of the Total cost of iters

bayed on given condition

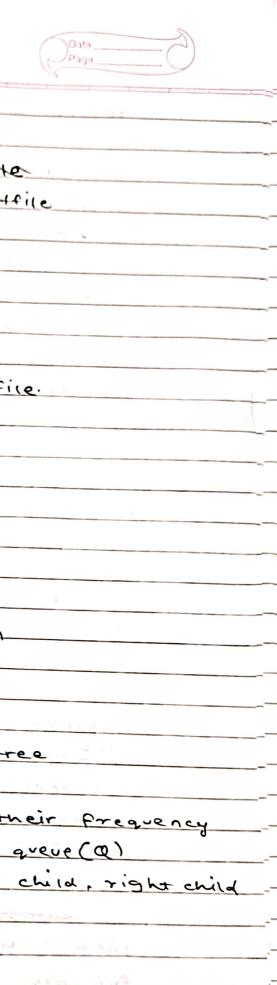
for i Eitems:

i [Value Index] = & i [cost] (i [shelflife] ti [capacity]
\$ 50++ items in descending order of andre Value Index

total cost = 0

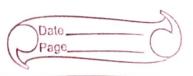


for i E items: if c> aW: break if i [capacity] + c <= W: totalcost += i [cos+] C+=i[capacity] else! fraction = (EW-c)/i[capacity] totalcost += fraction [(0:+] c = capaci W return total cost Test cases: Negative Testcases: · Any of the columns have non positive number OUTPUT: INVALID INPUT · It dataset is empty DUTPUT: EMPTY FILE . It any coronn is unfined: Output: INVALID SIZE OF COLUMNS Positive Test cases: Input: Knapsack data - 1 - csv Output: The Maximum cost in Transport vehicles The such that Them, have man Cost, Min Capacity and min Sheft Life is 14402.86 Input: Knapsack_data = 2, csv shy sole Output: The maximum cost in Transport venicles such that Items have man cost, min capacity and min Shelf Life it 14729



Task - 2 Algorithm: 1/ Imput: Taken from a textite 11 Input: Taken from a textile filename = file.txt data = filename . read() Compress FILE (Separa) 11 Input: Contents or txt fire 11 output: compression Ratio of fice. original length = len (data) * 8 COMPRATIO (Lata) // Input: Contents of tx+ file !/ output: compression Ratio of file original Length & = len (data) * 8 New Length = len (Huffman (data)) return original Length/ NewLength HUFFMANTREE (data) // Input: contents of txt file 11 Output: Root node of Huffmann tree n = len (data) Put all unique characters with their frequency into the a minimum priority queue (Q) Each tree mode has a left child, right child and a frequency. forb) i=11342401000 -11001118 117911 Allocate a new node 2 z-> left = x = dequeve (Q) 2-) right = y = dequeve (a) z -> freq = x-> fre2 + y-> freq

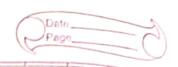
enqueux (Q,z)



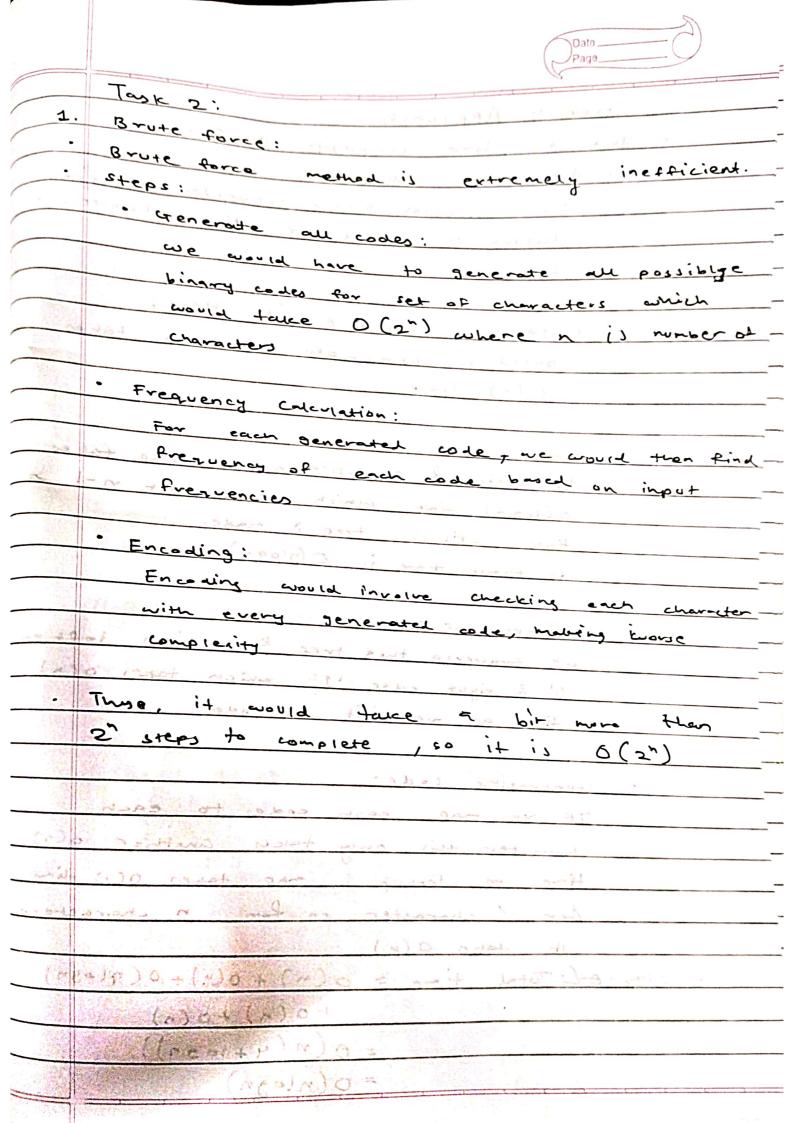
return dequeve (Q) HUFFMAN CODE (data) (100+) It Input: Root node of huffmann tree Lautput. HUFFMANCODE (TOOT, code) 11 Input: Initially root node of Huffmann tree and HEmpty string (code) initially which becomes the code generated so far 110 tput: Dictionary with huffman codes for all characters (least nodes) if me root is leaf node: Store root > character Dict + = [root > character: code] return Dict if root -> left exists: HUFFMANCODE (TOOK -> left, code + 'O') if root > right exists: HUFFMANCODE (TOOL > right, code + '1') ELALMAN LAL P HUPFMAN (database) // Input: Contents of file 11 output: Huffman encoded contents of file filename = filez.txt data2 = filename. write() for all characters i in data: doda 2 = replace = D Dict = HUFFMANCODE (HUFFMANETREE (data), " ") for all characters i in data: data 2 = D data 2. write() = Dict[i] return data 2

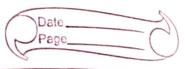


Testcares: Negative test cases: If input file is not txt, html, pdf or doc File: OUTPUT: NOT PROPER FORMAT OF FILE It input tile is empty: Output: EMPTY FILE Positive Testcases: 1. 4 Input = file 1. txt file1.+xt = "abcdaaaaa" Output = Compression Ratio is 24.00 Input = File 2. +x+ file 2. +xt = "the quick brown for jumps over the lazy dog domaraga phonon Output = compression ratio is 13.23 3 - Input = file3.+xt file3. +x+ = "A wonderful serenity has taken possession of my rentire world soul, like these sweet mornings of spring which I enjoy with my whole heart. I am alone, and feel the charm of existence ((neo) of in or this - spot. " Output = compression ratio is 15.13



	Time Complexity:
4.	Task 13
	Davido Force:
-	. It involves checking of every single subset
	of items
	· But for fractional we would also have
	to consider fractio subsets with fractioney
	amounts &
	· so we would have to consider infinitely
	many subsets, so traversing through then
	would be basically 0(00) reso hat
	possible estimate
	· while for non fractional it is O(2") as
	for there are a finite number of subsets
	of count 2" for a set of " n elements.
~ * .	a stance of a sont a state of the
2.	Greedy Approaching Man
	The greedy approach takes o(nlogn) as:
	o we calculate value by weight vatio which
	traverses growy once 121 50 , it takes O(n)
lustine	. Then we sort the array according to it
319-1	which takes O(nlogn) time as that is
ry I	the most efficient sorting time
	ma I musel mode in the
ons of	:. Total Time = 0 (n) + 0 (n1-97)
	((nco1+1) n) 01 = spot.
	81001 & other 200 (nlegn) = , , ,





1	
2	Greedy Approach:
•	This It's time complexity is Ockley to as:
7 × 5	. Frequency Calculation:
	This issurives single traversal of file
	taking o (n) time, n = No. of characters in
17:0	A Line . file - was a line of the same of
ىنى	
) New	· Building Priority Queve/Min Heap:
	building appropriate min heap takes
	6 (n) time.
	Frequence convergence
+ 0	ent emaking Tree:
	Entraction & Insertion in min heap takes
	O(logn) time which we consuct n-1
	times until 1 tree is made.
	so total time is O (nlogn)
4 1-0	Frederic word in a company of the
	Traversal of Trae:
	we traverse then tree & mosion left edge
, = u	O & right edge '11 which takes al. 1
	(1) on time as we wisit auronades
	Generating code:
	If we map each code to each
	time on the contract another o(1)
	time on lookup in map taken 0(1) time
Ţ.	it take O(n)
	· · · · · · · · · · · · · · · · · · ·
	· : · , of total time = o(n) + o(n) + o(n(-9n)
ſ	+o(n)+o(n)
4 = 1	= O(n(4+1001)) $= O(n(001)$
	7 [] [V\] (2) =\ V\]