Page	No.		
Date			

HAME: AFNAN ATTAR PRH: F19112003 CLASS - BE COMP2 SUBJECT: - BOAA ASSIGNMENT HO .- 04 ai) write time & space complexity of 0/1 knapsack algorithm using dynamic programming. Ans Since our memorization array using dynamic programming stores the result for all the subproblems we can conclude that we will not have more than NTC supproblems (where 'N' is the number of items and 'c' is the knapsuck capacity? Therefore, Time Complexity = O(N*C) And, space complexity = O(N * c) But we will also use O(N) space for call-stack hence, space Complexity = O (N * C+ N) ei o el el el el el el + + el or write realistic application of this experiment in brief. The energy end of F = 0 Am I Optimal load shedding model for microgrid 1-- The branch and bound algorithm to solve 0-1 knapsack problem one of the most widely used combinatorial optimization algorithms, is used to capture the customer values and the discrete characteristics of loads. II Download Managers: - One of the application is in Internet Download Manager (IDM) The data is broken into chunks, as per the maximum size of data that can be retrived in one go, the server uses this algorithm and packs the chunks so as to utilize the full size limit.

											Annual Control of the
The	wei'	ght	• 11	mit	- 4	by.	thi	s K	nap	sac	is 10 find
solutio	ni	wir	na .	du.	י מממ	nic	Dr	oar	am	min	ng
							ž.	3	Linear Specimens and Assessment of the	and the second second second second second	
		1	1		1			5			it of secondar
Renefit		3	40	7	,	. 2	1, 6	3	£	M1	in the will
We will create a table of 4 rows and 10											
		_							-		A MILLOR
											•
Itemw	eigr	t	ठ	-4	.5	6.1	7	8	'g	10	Sun full
	1			-	1				1		
		_	_	-		The Real Property lies		-	THE RESERVE AND ADDRESS OF THE PERSON NAMED IN	Section in which the Party of t	mar stirle se
3	0										Me Aryod Marie
- 2	enci i	o'th	. 30	1	7.5		0.11	cha	(12	Line	i loughter to both
						•					
		_									
								- 4			
3°140155		1115	1	1				1			
						0	e de condesse de la conse				A No. of the second
. 44		. / .		4411	4 4	otci,	(4)				77.4.7. 2 1/2/20
The second section of the sect							-				
	solution Hem Weigh- Benefit We colun We cell C tet u Item 3 Solution achie	solution in Hem Weight Benefit we will columns. We will cell [1][1 et us s 1 et us s 1 et us s 3 o 3 o 3 o 3 o	solution using Hem Weight: 3 Benefit: 3 We will condumns. We will weight Tet us solv Item 20 3 1 0 7 3 0 7 3 0 7 3 0 7	solution wing Item 0 Weight 2 Benefit 3 We will creat columns. We will use cell [ITCj] = n 1et us solve = Item 2 3 1 0 7 7 3 0 7 7 Solution: Maximal achieved by st	solution wing dy Item 0 1 Weight 2 2 Benefit 3 7 We will create columns. We will use th cell [I][]] = max 1et us solve = 1tem weight 3 4 0 7 7 10 3 0 7 7 10 Solution: Maximu achieved by steal	Hem 0 1 Weight 2 2 Benefit 3 7 We will create a columns. We will use the cell [ITCj] = max (1 1 et us solve = 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	solution wing dynamic Hem 0 1 2 Weight 2 2 4 Benefit 3 7 2 We will create a take columns. We will use the form cell [ITG] = max (cell item item item 1 2 3 4 5 6 1 0 7 7 10 10 10 3 0 7 7 10 10 10 Sourtion: Maximum progachieved by stealing. It	Hem 0 1 2 Weight 2 2 4 Benefit 3 7 2 6 We will create a table columns. We will use the formul cell [17[j] = max (cell [i-1tem + 1et us solve = 12 4 5 6 7 1 0 7 7 10 10 10 10 10 3 0 7 7 10 10 10 10 16 Solution: Maximum profit, achieved by stealing Item	Solution Wing dynamic progrittem O 1 2 3 Weight 2 2 4 5 Benefit 3 7 2 3 We will create a table of columns. We will use the formula: cell [I][] = max (cell [i-1] item + cell tem weight 1 tem 2 3 3 3 3 3 3 3 3 1 0 7 7 10 10 10 10 10 12 3 0 7 7 10 10 10 10 16 16 Solution: Maximum profit=19 achieved by stealing Item 3,	Solution wing dynamic program Hem 0 1 2 3 Weight 2 2 4 5 Benefit 3 7 2 9 We will create a table of 4 columns. We will use the formula: cell [17[j] = max (cell [i-1] [j] 1 tem + cell [1 et us solve: 1 tem 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3	Weight 2 2 4 5 Benefit 3 7 2 9 We will create a table of 4 ro columns. We will use the formula: cell [i][j] = max (cell [i-1][j], v item + cell [i-1. 1et us solve: 1tem 2 5 4 5 6 7 8 9 10 0 0 3 3 3 3 3 3 3 3 3 3 3 1 0 7 7 10 10 10 10 10 10 10 2 0 7 7 10 10 10 10 10 12 12 12

11,

Page	No.	
Date		

	Date
	NAME :- AFNAN ATTAR PRN:- F19112003 CLASS :- BE COMP2
Bet 151, 613	SUBJECT: DAA ASSIGNMENT NO: 05
	ix remember our si i processi.
(10	what is backtracking? What are the peculiar
	characteristics & approach?
Ans 1	Backtracking is a general algorithm for solving some
	computational problems, most notably constraint
	satisfaction problems, that incrementally builds
	candidates to the solutions and abandons a
Exting	candidate's backtracks as soon it determines
	that the candidate cannot be completed to a
Signer 1	resonable solution.
2.	In bactracking problem, the algorithm tries to find
	a sequence path to the solution which has some
	small checkpoints from where the problem can
	backtrack if no feasible solution is found,
10731	Applications: exapt 1 (111) or of xelyan score
- ' '	Decision problems used to find a feasible solution
a ili)	Optimisation problems used to find the best solution that can be applied.
	that can be applied a south of the
iii	Enumeration problem used to find the set of all
	feasible solutions of the problem
8-	
Q 2)	Explain Explicit & Implicit constraints with respect to
	8 queen's problem?
Ans 1.	A classic problem in combinatories is to place & queens
	on an 8 by & chessboard so that no two can
	attack each other (From column or diagonal wise).
2 . F. W. Q .	Since each queen (1-8) must be on a different row, we
	can assume queen il is ion move !

Page No.			
Data	7	T	-

B. All solutions to the 8-queens problem can be represented as an 8-tuple (x1, x2, ..., x8) where queen i is on column xi. 4. Explicit Constraints: Bi= 91,2, et, 1=i=8 The solution space consists of 88 8- Tuples. s. implicit Constraints: No two xi's can be the some (as queens must be on different columns) and no two queens can be on the same diagonal. anothered in the morning to the of white or 93) Compare the space & time complexity of recursive & Mon-recursive technique of backtracking. And if Recursive 1-In the worst possible case we may end up trying out of all possible arrangement of H-chusens as His we are using a matrix of size M*H to represent the board had an Andrews Time Complexity: O(H!) I space Complexity: O(HAN) ACTED Hon-recursive: With of page wardon no notioned Hon- recursive algorithm can be implemented using stack data structure, hence space and time complexity remains same as recursive Time Complexity O(N!) 1 space Complexity : O(N2) ou) write realistic applications of this experiment in brief. AnoIl Artificial Intelligence in Video Games: - Backtracking algorithm can be used to devise AI to play tic-tae-toe, chess and various video games. I) Programming languages is - Rogic Programming languages such as Icon, Planner, prolog use backtracking internally to generate answer.