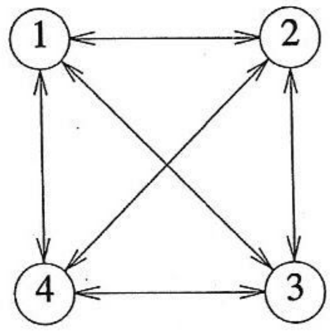
LAB ASSESSMENT

1. A programmer needs to break a certain string into two parts. If the string contains *n* characters, it costs *n* units of time to break it into two parts, as it involves copying the old string. The programmer wants to break the string into more than 2 parts, and the order in which the breaks are made affects the total amount of time used. For example, suppose we wish to break a 20-character string after characters 3, 8, and 10. If the breaks are made in left-right order, then the first break costs 20 units of time, the second break costs 17 units of time, and the third break costs 12 units of time, for a total of 49 steps. If the breaks are made in right-left order, the first break costs 20 units of time, the second break costs 10 units of time, and the third break costs 8 units of time, for a total of only 38 steps. Give a dynamic programming algorithm that takes a list of character positions after which to break and determines the cheapest break cost in O(n^3) time.

2. Suppose the job scanning through a shelf of books is to be split between k workers. To avoid the need to rearrange the books or separate them into piles, we can divide the shelf into k regions and assign each region to one worker. What is the fairest way to divide the shelf up? If each book is the same length, partition the books into equal-sized regions,
   100 100 100 | 100 100 100 | 100 100 100
   But what if the books are not the same length?
   This partition would yield 100 200 300 | 400 500 600 | 700 800 900
   Which part of the job would you volunteer to do? How can we find the fairest possible partition, i.e. 100 200 300 400 500 | 600 700 | 800 900

3. Implement Greedy Fractional Knapsack for the following problems:
   1. N =3, M = 20, (P1, P2, P3) = (25, 24, 15), (W1, W2, W3) = (18, 15, 10)
   2. N =7, M = 15 , (P1, P2, P3, P4, P5, P6, P7) = (10, 5, 15, 7, 6, 18, 3); (W1, W2, W3, W4, W5, W6, W7) = (2,3,5,7,1,4,1)

4. Implement the dynamic programming algorithm for the Traveling Salesman Problem (TSP). Describe the data structure you use for the g(i,S) functions. Run your program for the following graphs:
   Graph 1:

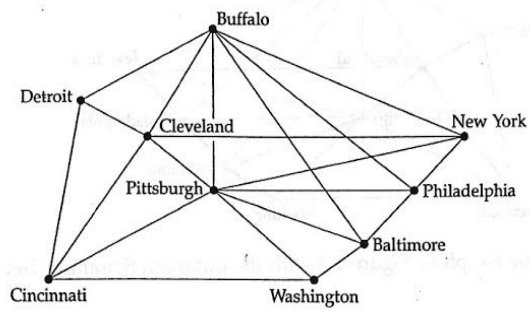


(a)

$$\begin{bmatrix} 0 & 10 & 15 & 20 \\ 5 & 0 & 9 & 10 \\ 6 & 13 & 0 & 12 \\ 8 & 8 & 9 & 0 \end{bmatrix}$$

(b)

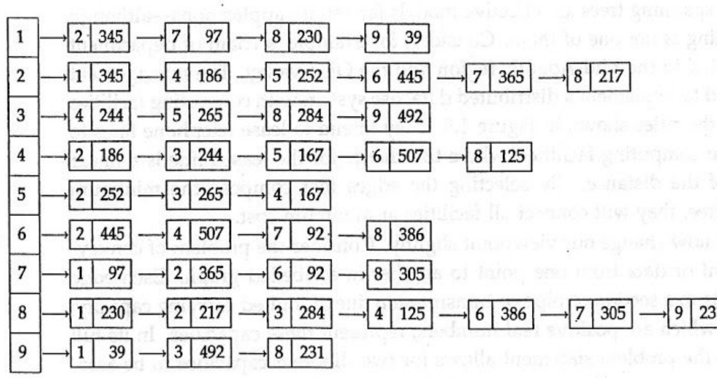
Graph 2:



(a) The graph

	1	2	3	4	5	6	7	8	9	
1		345					97	230	39	Baltimore
2	345			186	252	445	365	217		Buffalo
3				244	265			284	492	Cincinnati
4		186	244		167	507		125		Cleveland
5		252	265	167						Detroit
6		445		507			92	386		New York
7	97	365				92		305		Philadelphia
8	230	217	284	125		386	305		231	Pittsburgh
9	39		492					231		Washington

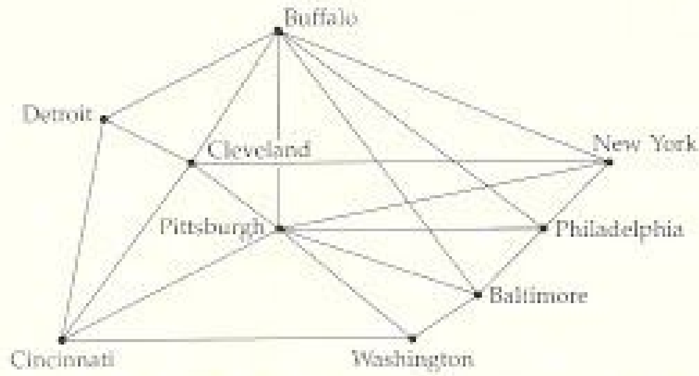
(b) The adjacency matrix representation of the graph



(c) The adjacency list representation of the graph

Figure 4.1: Two Internal Representations of a Graph

Graph 3:



(a) The graph, showing only direct connections

Baltimore	0	345	514	355	522	189	97	230	39
Buffalo	345	0	430	186	252	445	365	217	384
Cincinnati	514	430	0	244	265	670	589	284	492
Cleveland	355	186	244	0	167	507	430	125	356
Detroit	522	252	265	167	0	674	597	292	523
New York	189	445	670	507	674	0	92	386	228
Philadelphia	97	365	589	430	597	92	0	305	136
Pittsburgh	230	217	284	125	292	386	305	0	231
Washington	39	384	492	356	523	228	136	231	0

(b) The distance matrix

Figure 1.3: An Instance of the Symmetric Travelling Salesperson Problem with Triangle Inequality