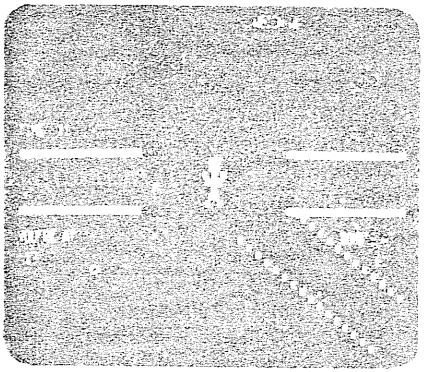
TIME BOMB

Tick ! Tick ! Tick !



by David Bohlke

Seconds count. Somewhere deep inside a towering sky-scraper the time bomb is ticking away. Would you be able to locate and disarm this explosive device in time to save the building and its inhabitants? Maybe, but for everyone's sake, you'd better get in a little practice with this computer simulation first!

ILLUSTRATION

Time Bomb Line Summary:

5-30	Sets boundary of maze
100-190	Loop to generate maze
	Sets A(I) digits to A(X)
200-280	Picks direction from A(I) grid to A(X)
	Variable P is reciprocal direction from A(X) to A(I)
300-360	Sets A(X) digits
390-392	Picks Player and Bomb locations
394	Sub. to set Player and paths
400-408	Player input direction
409-482	Moves player on screen
485-495	Determines won/loss. Sets new intersection
500-582	SETS paths
600-636	SETS man
700-730	RESETS man
800-810	J, K, L are digit numbers of paths in given A(X)
900-955	Won/loss messages

TIME BOMB

'Time Bomb' is a unique new game that utilizes a randomly generated maze with 100 intersections on a 10 x 10 grid (see Figure 1). A bomb is then hidden in one intersection, and the player is put in another. The object of the game is to "diffuse" the bomb by searching through the maze and finding the intersection where the bomb is hidden.

You will not, however, be shown the entire maze. All you will be able to see is the intersection that you are located in; and the paths leading from that grid to the possible neighbors (Figure 2). The computer will ask which of these paths you want to take; and you then enter the direction of the path you want to follow. Figure 3 shows the numbers used to indicate direction.

To aid you in moving through the maze, the program gives you a clue by telling you the number of the grid where the bomb is located; and the grid number of your location. The grid numbers are indicated in Figure 1. The important thing to remember is that if you are in grid 26 and the bomb is in grid 74, then you need to take paths which lead down and/or to the left whenever possible. Going back to Figure 2, the best choice of paths would be straight down; which is a direction of 7.

If you were to enter a 7, the little man would move down along that path, and then grid 36 would appear on the screen (since it is in a direction of 7 from grid 26). You continue picking directions from each intersection, always trying to move towards the bomb's location, until you land on the bomb's grid (hopefully within 15 moves). By landing on the Bomb's grid you diffuse the bomb.

If you succeed, then a safe message is printed; and you and the bomb are re-set in new grids and ready to play another game. If you don't find the bomb's grid in 15 tries a failure message is printed; and new starting positions are re-set as when you win.

As you can see in Figure 1, each intersection usually has three paths leading from it. It is possible to go from the first column to the tenth by taking the dashed paths. The maze in Figure 1 is just a single example — everytime the program is RUN a new maze is computed. This usually takes about 90 seconds. If you play the game several times using the same maze it might be helpful to construct the entire maze on scratch paper as you move from grid to grid. Thus play will be easier; and you will get a better indication of what a complete maze looks like.

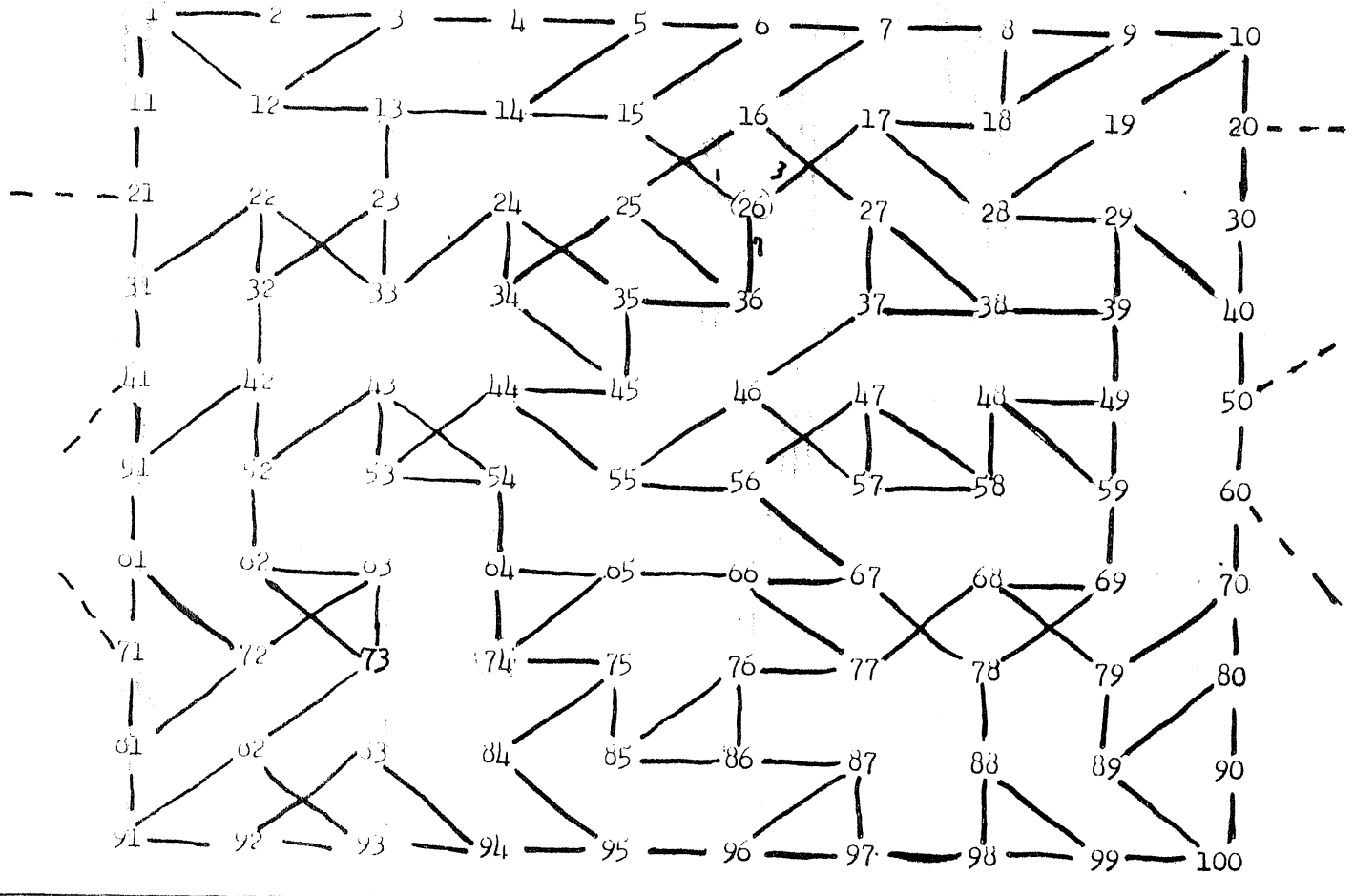

1 2 3 4 5 6 7 8 9 10
11 12 13 14 15 16 17 18 19 20
21 22 23 24 25 26 27 28 29 30
31 32 33 34 35 36 37 38 39 40
41 42 43 44 45 46 47 48 49 50
51 52 53 54 55 56 57 58 59 60
61 62 63 64 65 66 67 68 69 70
71 72 73 74 75 76 77 78 79 80
81 82 83 84 85 86 87 88 89 90
91 92 93 94 95 96 97 98 99 100

FIGURE 3