Game of Life:

Write an assembly language program that simulates Conway's *Life*. The simulation takes place on a rectangular array of cells, each of which may contain an organism. Except for borders each cell has eight cells immediately adjacent to it, and so each organism may have up to eight neighbors. The survival and reproduction of organisms from generation to generation depend upon the number of neighbors according to four simple rules:

(1) If an organism has no neighbors or only one neighbor, it dies of loneliness.

(2) If an organism has two or three neighbors, it survives to the next generation.

(3) If an organism has four or more neighbors, it dies of overcrowding.

(4) An organism is born in any empty cell that has exactly three neighbors.

All changes occur simultaneously; the fate of an organism depends on the current generation irrespective of what may happen to its neighbors in the next generation. Therefore, the game may be simulated in the program using two $m \infty n$ arrays of bytes, `CURG` and `TEMPG`. Each array component contains the ASCII code for the letter "O" if the cell contains an organism, an ASCII space otherwise.

For each cell in `CURG`, the program examines the neighbors and puts the next generation outcome in the corresponding cell in `TEMPG`. (Border processing may be simplified by using an $m+2 \infty n+2$ array in which the borders have been initialized to always contain spaces.) After processing all cells in `CURG`, the program can copy `TEMPG` into `CURG`, or simply swap the roles of `CURG` and `TEMPG`.

The array $m \infty n$ may correspond to the size of the display screen used in Lab #3 and Lab #4.

Write a subroutine `DISPLAY` that displays the current generation of Life in the `CURG` array by calling another subroutine `COUT` to display each character in the Life array. Write a main program that initializes the Life array to a random population and computes and displays Life one generation at a time.