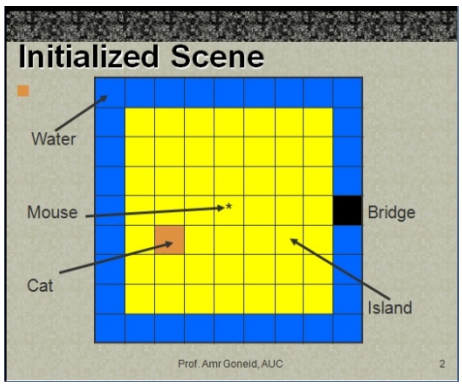
CSCE 1101 Fall 2022: Fundamentals of Computing II
Assignment #1

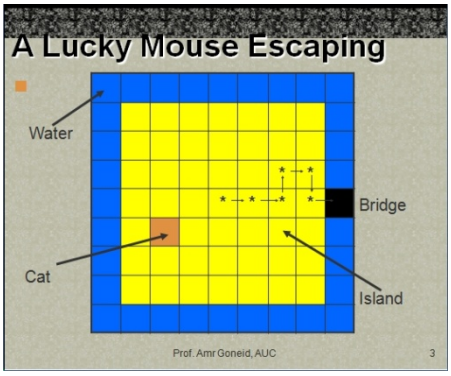
Dr. Amr Goneid ***Date: Thu Sept 15, Due: Thu Sept 22, 2022***

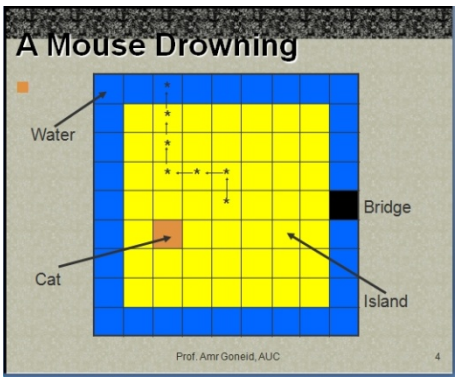
Implement the following program. It will serve as a revision of the basic C++ material provided in parts 1-7 of the slides.

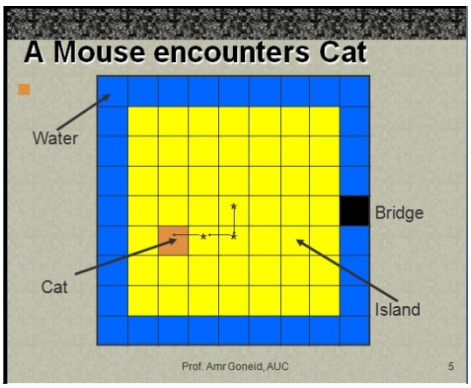
Problem Statement:

The scenes below represent an island (yellow squares) surrounded by water (blue squares). There is a cat on the island (brown square) and a bridge leading to safety (black square). A (drunken) mouse is placed in the center of the island.









Write a program to make the mouse move randomly across the island. The mouse is allowed to move one square at a time, either horizontally or vertically (no diagonal moves). A random number between 0 and 3 inclusive is used to define the direction of the next move (0 for North, 1 for East, 2 for South and 3 for West). The mouse drowns if it hits water, dies of starvation if it makes more than 100 moves, gets eaten if it meets the cat, or hopefully it crosses the bridge. The program should be run from the start several times to count the number of times the mouse drowns, escapes or dies.

Design:

1. Represent the scene as a 2-D array **A** of characters with N rows and N columns, where N is an odd number ≥ 7.
2. Use the following characters to represent contents of scene:
   (*) for mouse, char(219) for water, char(175) for bridge, char(32) for island and any symbol you choose for the cat.
3. **Initialise the scene** as follows (r is the row number, c is the column number):

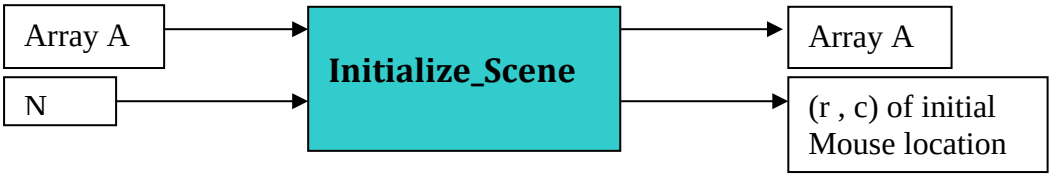
- Put blanks everywhere in array A
- put water in A[0][c] , A[N-1][c] , c = 0 .. N-1
- put water in A[r][0] , A[r][N-1] , r = 0 .. N-1
- Set bridge in A[N/2][N-1]
- Set mouse in A[N/2] [N/2]
- Set initial position of mouse to r = N/2 and c = N/2 (i.e. in the center)
- Set the cat in a random position on the island

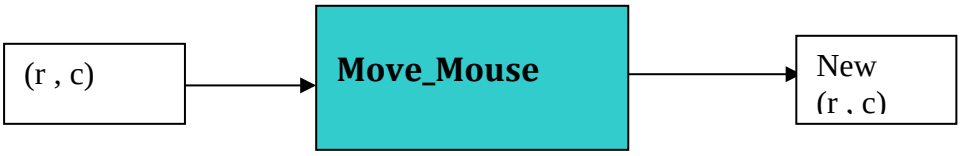
4. **To move mouse one step:**
   - Given the current position (r,c), generate a random number (RN) between 0 and 3.
   - Find the row and column displacements (dr , dc) according to the following table:

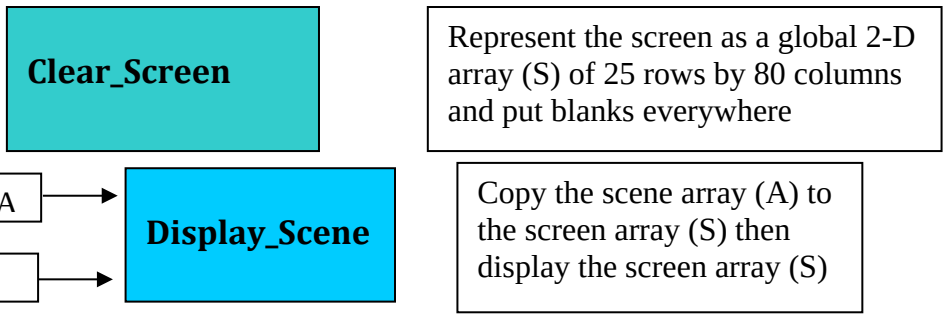
RN	Direction	dr	dc
0	North	-1	0
1	East	0	+1
2	South	+1	0
3	West	0	-1

   - The new position will be r = r + dr and c = c + dc

5. **Implement the following functions:**







Suggested Algorithm

```
Clear_Screen;
Input Island Size N and set the maximum numbers of runs to rmax
Initialize Random Number Generator
Set Starvation Limit = 100;          died = escaped = drowned = 0;      runs = 0;
Do
{
        Initialize_Scene;      runs++ ;
        Display_Scene;
        moves = 0;     finished = false;
        while( ! finished )
        {
                Move_Mouse(r,c);    change contents of A[r][c] to '*' ;   moves ++;
                if mouse hits water  //the mouse has drowned
                        { drowned++;  finished = true; }
                else if mouse reached bridge //the mouse has escaped
                        { escaped++; finished = true; }
                else     //mouse is still in island
                         if ((mouse meets cat) or  (moves > 100))    //The mouse dies
                                { died++; finished = true;}
                Display_Scene;
        }
        Output number of moves done;
 } while (more runs are needed);
Output how many times the mouse drowned, escaped or died.
```

Note on using the C++ Random Number Generator (RNG):

The above program uses random numbers generated by a Random Number Generator (RNG). The RNG in C++ is a function **rand( )** that returns a random integer from 0 to 32,767. To obtain random integers from 1 through n, use **rand( ) % n + 1.** Generally, you may implement a function ***RandInt (i, j)*** that generates an integer between *i* and *j* with equal probability. This is implemented simply as follows:

```
int RandInt (int i , int j)
{
        return rand( ) % (j-i+1) + i ;
}
```

To obtain a random sequence you need to first initialise the RNG using the time of the machine as a seed. This is done so that we do not get the same sequence every time we run the program. The following is an **example** of how to generate a random sequence of pairs of integers with values between 1 and n:

```
#include <time.h>
int x , y, n ;
…….
srand ( (unsigned) time (NULL) );          //Initialize RNG
repeat as needed                           // Loop over the sequence
```

```
{
        x = RandInt (1 , n);          // Generate 1st number
        y = RandInt (1 , n);          // Generate 2nd number
        ……..                          // Do something with x and y
}
```