```
#include "SIgame.h"
#include "mbed.h"

/****************************************************************/
/*                                                              */
/* Game-Related Functions                                       */
/*                                                              */
/****************************************************************/
OBJECT startShip()
{/******
  * startShip
  *     Sets up structure for player's ship
  ******/
    OBJECT ship;

    ship.height = SHIP_HEIGHT;
    ship.width = SHIP_WIDTH;
    ship.y = SCREENHEIGHT - ship.height + 1; // Starting location
    ship.x = SCREENWIDTH >> 1;
    ship.color = GREEN; // Player's ship is green
    ship.killed = false;

    return ship;
}

void summonWave()
{/******
  * summonWave
  *     Initialize 2-D array containing columns of aliens and their starting
  *     indicies
  ******/
    int r, c, i, j;
    j = 1;

    for (c = 0; c < NUM_ALIEN_COLS; c++)
    {
        i = 0;
        for (r = 0; r < NUM_ALIEN_ROWS; r++)
        {   // Within current column, initialize aliens' locations
            wave[r][c].x = j;
            wave[r][c].width = ALIEN_WIDTH;
            wave[r][c].y = i;
            wave[r][c].height = ALIEN_HEIGHT;
            wave[r][c].color = WHITE; // Aliens are white
            wave[r][c].killed = false;
            i += (ALIEN_HEIGHT + 1);
        }

        j += (ALIEN_WIDTH + 3); // Ready gap for next column of aliens
    }
}
```

```
void destroyAlien(int *pals_rem, POINT *pylaser, OBJECT *frontline[])
{/******
  * destroyAlien
  *     Checks if ship's laser hits any alien and kill alien if so
  ******/
    bool hit = false;
    int r, c = 0;

    do
    {   // Go to column from wave of aliens of which laser may hit alien
        if (pylaser->x >= wave[0][c].x && pylaser->x <= wave[0][c].x +
                wave[0][c].width - 1)
        {
            r = 0;
            do // Find alien in column that laser touches if possible
            {
                if (pylaser->y >= wave[r][c].y && pylaser->y <= wave[r][c].y +
                        wave[r][c].height && !(wave[r][c].killed) &&
                        !(pylaser->collide))
                {   // Alien is killed and not be drawn on screen
                    (*pals_rem)--;
                    wave[r][c].killed = pylaser->collide = hit = true;

                    // Adjust pointer to front alien of selected column if killed
                    // alien was one
                    if (frontline[c] == &wave[r][c])
                    {
                        if (!r) // All aliens on that are wiped out
                            frontline[c] = NULL;
                        else // Point to alien behind killed one
                            frontline[c] = &wave[r-1][c];
                    }
                }
                r++;
            } while (!hit && r < NUM_ALIEN_ROWS);
        }
        c++;
    } while (!hit && c < NUM_ALIEN_COLS);
}


bool twoLasersCollide(POINT *pylaser, POINT elaser[])
{/******
  * twoLasersCollide
  *     Checks if player's laser touches any of alien's lasers and removes them if
  *     so
  ******/
    int i;

    for (i = 0; i < ELASER_CAP; i++)
    {   // Find enemy laser in same location as player's laser if so
        if (pylaser->x == elaser[i].x && pylaser->y == elaser[i].y &&
                !(pylaser->collide || elaser[i].collide))
        {   // Both lasers cancel each other out and disappear on screen
            pylaser->collide = elaser[i].collide = true;
            return true;
        }
    }
    return false;
}


bool moveAlienWave(bool *pleft)
{/******
  * moveAlienWave
  *     Changes all aliens' locations to move wave of aliens
  ******/
    int r, c;
    bool reach, old_dir;
    reach = false;
    old_dir = *pleft;

    // Flip direction when wave reaches either side of screen
    if (wave[NUM_ALIEN_ROWS-1][0].x <= 0 && *pleft)
        *pleft = false;
    else if (wave[NUM_ALIEN_ROWS-1][NUM_ALIEN_COLS-1].x + ALIEN_WIDTH >=
            SCREENWIDTH && !(*pleft))
        *pleft = true;

    // Change all aliens' locations in array
    for (c = 0; c < NUM_ALIEN_COLS; c++)
    {
        for (r = 0; r < NUM_ALIEN_ROWS; r++)
        {
            if (old_dir ^ *pleft) // Moves downward if end columns touch borders
                wave[r][c].y += ALIEN_HEIGHT;
            // If not moving down, then moves horizontally
            if (*pleft)
                (wave[r][c].x)--;
            else
                (wave[r][c].x)++;

            // Player loses if any alien reaches screen border at player's side
            if (wave[r][c].y > SCREENHEIGHT - (wave[r][c].height << 1) + 1)
                reach = true;
        }
    }
    return reach;
}
```