Asgn1 Design Document
CSE-13S
Despina Patronas
Program: lrc

Variable Declarations:

```
//Scope is global and accessible only to pertaining source file
    DEFINE data type faces
    DEFINE faceim enum                  //4 side possibilities (L, R, C, P)   ie: LEFT, RIGHT, CENTER, PASS
    DEFINE faces type die array                             //6 faces possibilities (L, R, C, P, P, P)
    DECLARE CONST CHAR pointer array of 10 players names
    DECLARE INT players,  pot, pos,  seed = 0               //pos ie current player
```

Function Declarations:

```
//find player to left of pos
INT previous_player (INT pos, INT players)
    return ((pos + players - 1) MOD players

//find player to right of pos
INT next_player (INT pos, INT players)
    return ((pos + players + 1) MOD players

//generate random numbers in range [0,5]
INT generate rand num()
    return rand() MOD 6

//check the number of players in the game
INT check_players (INT pointer to player_balance[], INT num_players)
    set players = 0
    Iterate from 0 to (numplayers – 1)
        IF the pointer player_balance > 0
            increment players
            increment player_balance //next element in array
        ENDIF
    return players

//display and input for seed and players NOTE: there is no input validation occurring!
VOID start_process()
    DISPLAY request random seed
    INPUT seed
    Srand(seed)
    DISPLAY request players count
    INPUT players

INT main ()
    DECLARE INT temp = 0                //temp value to hold random number
    DECLARE INT maxroll = 0             //tracks the amount of times player rolls
    CALL start_process function
    DECLARE INT array for player_balance.  Allocate 10 elements
    Iterate from 0 to (players – 1)
        populate player_balance elements = 3         //the rest of elements will remain 0

    WHILE check_players (player_balance, players) > 0  DO:
        IF (player_balance[pos] > 0)                 //if current player balance is not 0 initiate a turn
            DISPLAY names[pos]                       //current player rolling
            SET maxroll = 0

            // updates player money status checks that player hasn't rolled more than 3x
            // player must have money to roll
            WHILE player_balance[pos] > 0) AND (maxroll < 3 ) DO:
                SET temp = roll()
                CASE based on die[temp]
                    CASE 0 //die lands on L
                        DISPLAY current player money exchanged to previous_player
                        decrement current player_balance
                        increment maxroll
                        BREAK
                    CASE 1 // die lands on R
                        DISPLAY current player exchange of money to next_player
                        decrement current player_balance
                        increment maxroll
                        BREAK
                    CASE 2 //die lands on C
                        DISPLAY names[pos] exchange of money to the pot
                        decrement current player_balance
                        increment maxroll
                        BREAK
                    CASE 3 //die lands on P
                        DISPLAY pass
                        increment maxroll
                        BREAK
                    DEFAULT CASE
                        DISPLAY bad case
                        BREAK
            END OF NESTED WHILE LOOP
        SET pos = call next_player(pos, players)                //update pos of player to the right
        ENDIF
    END OF WHILE LOOP
    DISPLAY the name of the current player position, amount in the pot, and amount in player balance
    RETURN 0
END of main
```