```c
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#include"dicegame.h"

int main(int argc, char* argv[])
{
        // Initialize the srand() to start the random generator
        // Initialize the player-1 and player-2 scores to 0
        // Initialize all other required variables
        // Ask the user for the number of rounds to run the game
        // Dynamically create an array of DiceRound struct objects, based on the user input
        // Check if array created. If not end program, by calling return 1;

        // Fill the DiceRound array by calling the fillRounds() function
        // Call printPlayerInfo() function to print the initial player scores which will be 0

        for(i=0; i<nRounds; i++)
        {
                // Call getRoundPoints() to get the actual points to be used for this round
                printf("\nROUND %d\n--------\n", i+1);
                printf("Player\t: %d\n", curPlayer); // Print current player
            // Call printRoundInfo() to print the round information

                // MAIN GAME LOGIC
                // Write code here to get the main game logic using branches
                // IF Player-1 (even player) is the current player and the dice rolled is even
                // OR
                // IF Player-2 (odd player) is the current player and the dice rolled is odd
                        // THEN Current player gains the points, based on the type of the round. The playerâ€™s turn continues
in the next round.
                // IF Player-1 (even player) is the current player and the dice rolled is odd
                // OR
                // IF Player-2 (odd player) is the current player and the dice rolled is even
                        // THEN Current player incurs penalty of the same amount of points, based on the type of the round
(see above). In the next round, the current player is changed to the other player.

                // Call printPlayerInfo() to print the player information at the end of the round
        }
        printf("\nGAME OVER!!\n");
        // Compare the final points for player-1 and player-2
        // Print the winner as the one with higher points

        // Free the dynamically allocated array memory

        return 0;
}
```