```c
/**
* OBLIG 2
*
* THE PROGRAM:
* -Set up variables and constants
* -Create methods and functions
* -Optimize code
* -Use comments and documentation
*
* @file main.c
* @author Daniel AG
*
* Includes:
* - stdio.h for displaying and feeding data
* - stdbool.h for using booleans
* - ctype.h for toupper
*/

///Internal Includes
#include <stdio.h> //printf,scanf
#include <stdbool.h> //bool
#include <ctype.h> //toupper
#include <string.h> //strcpy,strcspn


/**
* @def MAXLANES
* @brief Maximum number of lanes
*/
#define MAXLANES 18

/**
* @def MAXLANELENGTH
* @brief Maximum length of a lane
*/
#define MAXLANELENGTH 100

/**
* @def MAXPARS
* @brief Maximum number of Pars
*/
#define MAXPARS 8

/**
* @def STRLEN
* @brief Maximum length of a string (array of char)
*/
#define MAXSTRLEN 100
/** @}*/
```

```c


char laneDescription[MAXLANES][MAXSTRLEN] = { 0,0 }; // lane description
int laneLength[MAXLANES] = { 0 }; // lane length
int lanePar[MAXLANES] = { 0 }; // lane Par
bool laneOB[MAXLANES] = { 0 }; // lane OB
int numLanes = 0; // number of lanes


/**
* @def Run
* @brief Runs the program
* @return nothing
*/
void Run();

/**
* @def Add
* @brief Adds a new lane
* @return true or false
*/
void Add_Lane();

/**
* @def Display
* @brief Displays all lanes
* @return nothing
*/
void Display_Lane();

/**
* @brief Executes the program
* @return int
*/
int main()
{
    // Default values
    laneLength[0] = 62;
    lanePar[0] = 3;
    laneOB[0] = true;
    strcpy_s(laneDescription[0], MAXSTRLEN, "Lane with a lot of trees and 
      scrub");

    laneLength[1] = 94;
    lanePar[1] = 3;
    laneOB[1] = false;
    strcpy_s(laneDescription[1], MAXSTRLEN, "Flat terrain thourgout the 
      map");
    numLanes = 2;

    Run();
}

void Run()
{
    char choice; // User choice for input
    do // conditional logic in while loop
    {
        /**
        * 1. Printing the menu
        * 2. Getting user input
        * 3. Activate switch case
        * 4. Returning to menu unless Q is pressed
        */
        printf("Menu Choices:\n");
        printf("  A - Add lane:\n");
        printf("  D - Display alle lanes:\n");
        printf("  Q - Quit:\n");
        printf("  Select a choice:");
        scanf_s(" %c", &choice, 1);
        choice = toupper(choice);
        printf("\n");

        switch (choice) // steps into corresponding case
        {
        case 'A':
        {
            Add_Lane();
            break;
        }
        case 'D':
        {
            Display_Lane();
            break;
        }
        case 'Q':
        {
            printf("Quit - selected:\nEXITING PROGRAM");
            break;
        }
        default:
            printf("Illegal argument");
        }
    } while (choice != 'Q');
}

void Add_Lane()
{
    /**
        * 1. Takes inputs and generates new lanes
        * 2. The new lanes are added to the respective arrays
        * 3. After a new lane is created, increment number of lanes
    */
    if (MAXLANES <= numLanes) // is the maximum number of lanes reached?
    {
        printf("[LOG]:Max Number of lanes created\n");
        return;
    }

    //Utility
    int currentLane = numLanes;

    // Lane data
    int qLaneLength = 0;
    int qLanePar = 0;
    char qLaneOB;
    char qLaneDescription[MAXSTRLEN];

    printf("How long is lane %i:", currentLane + 1);
    scanf_s("%d", &qLaneLength);
    if (qLaneLength <= 0) // flag if length is less or equal to 0
    {
        printf("Illegal argument..\nReturning\n\n");
        while (getchar() != '\n');
        return;
    }
    // update lane length for current lane
    laneLength[currentLane] = qLaneLength;

    // Take input for lane Pars
    scanf_s("%d", &qLanePar);
    printf("Pars on the field. Choose a number bwteen (2-8):");
    if (MAXPARS < qLanePar) // flag if max Pars is overreached
    {
        printf("Max Pars, set to MAXPARS\n");
        qLanePar = MAXPARS;
    }
    if (qLanePar < 2) //flag if illegal argument
    {
        printf("Illegal argument..\nReturning\n\n");
        while (getchar() != '\n');
        return;
    }
    // update lane Pars for current lane
    lanePar[currentLane] = qLanePar;

    // Take input for lane OB
    printf("Does the lane have OB (y = yes or n = no):");
    while (getchar() != '\n');
    scanf_s(" %c", &qLaneOB, 1);
    qLaneOB = toupper(qLaneOB);
    if (qLaneOB != 'Y' && qLaneOB != 'N') // flag if illegal argument
    {
        printf("Illegal argument..\nReturning\n\n");
        while (getchar() != '\n');
        return;
    }
    // update lane OB for current lane
    laneOB[currentLane] = (qLaneOB == 'Y');

    printf("Write a description:"); // Take input for lane description
    while (getchar() != '\n');
    fgets(qLaneDescription, MAXSTRLEN, stdin); // Read string with spaces

    // Remove newline character from string "\n"
    qLaneDescription[strcspn(qLaneDescription, "\n")] = 0;

    // copy the content of qLaneDescription to laneDescription
    strcpy_s(laneDescription[numLanes], MAXSTRLEN, qLaneDescription);

    numLanes++; //increment number of lanes
    printf("Lane %i added\n\n", currentLane); // confirm lane added
}

void Display_Lane()
{
    /**
        * 1. Iterate over all arrays and retieve content
        * 3. It then displays content
        *
    */

    int totNumPars = 0; // total number of Pars

    for (int i = 0; i <= numLanes; i++) // iterate through all lanes
    {
        if (numLanes == 0) // flag if no lanes are available
        {
            printf("No lanes available\n\n");
            continue; //skip empty lanes
        }
        else if (laneLength[i] == 0) //flag if no data on lane
        {
            printf("No data on lane %d\n\n", i);
            continue;
        }
```

```c
        printf("Lane \033[1;4m%d\033[0m:\n", i + 1);
        printf("     Length: \033[1;4m%d meters\033[0m\n", laneLength[i]);
        printf("     Pars: \033[1;4m%d\033[0m\n", lanePar[i]);
        printf("     OB: %s\n", laneOB[i] ?
                "\033[1;4m With \033[0m" : "\033[1;4m Without \033[0m");
        printf("     Description: \033[1;4m%s\033[0m\n\n",
                     laneDescription[i]);

        totNumPars += lanePar[i]; // sum up total number of Pars
    }

    printf("Summary\n");
    printf("Total number of lanes: \033[1;4m%d\033[0m\n", numLanes);
    printf("To get to Par, it requires number of throws: \033[1;4m%d\033
      [0m\n\n", totNumPars);
}
```