```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
/** @}*/

///Global Variables
char laneDescription[MAXLANES][MAXSTRLEN] = { 0,0 }; ///< lane description
int laneLength[MAXLANES] = { 0 }; ///< lane length
```

```
53  int lanePar[MAXLANES] = { 0 }; ///< lane Par
54  bool laneOB[MAXLANES] = { 0 }; ///< lane OB
55  int numLanes = 0; // number of lanes
56
57  ///Declarations
58  void Add_Lane(); // Declartion of Add_Lane
59  void Display_Lane(); //Declaration of Display_Lane
60
61  /**
62  * @brief Executes the program
63  * @return int
64  */
65  int main()
66  {
67      // Default values
68      laneLength[0] = 62;
69      lanePar[0] = 3;
70      laneOB[0] = true;
71      strcpy_s(laneDescription[0], MAXSTRLEN,
72          "Lane with a lot of trees and scrub");
73
74      laneLength[1] = 94;
75      lanePar[1] = 3;
76      laneOB[1] = false;
77      strcpy_s(laneDescription[1], MAXSTRLEN,
78          "Flat terrain thourgout the map");
79      numLanes = 2;
80
81      char choice; // User choice for input
82      do // conditional logic in while loop
83      {
84          /**
85          * 1. Printing the menu
86          * 2. Getting user input
87          * 3. Activate switch case
88          * 4. Returning to menu unless Q is pressed
89          */
90          printf("Menu Choices:\n");
91          printf("  A - Add lane:\n");
92          printf("  D - Display alle lanes:\n");
93          printf("  Q - Quit:\n");
94          printf("  Select a choice:");
95          scanf_s(" %c", &choice, 1);
96          choice = toupper(choice);
97          printf("\n");
98
99          switch (choice) // steps into corresponding case
100         {
101         case 'A':
102         {
103             Add_Lane();
104             break;
105         }
```

```
106         case 'D':
107         {
108             Display_Lane();
109             break;
110         }
111         case 'Q':
112         {
113             printf("Quit - selected:\nEXITING PROGRAM");
114             break;
115         }
116         default:
117             printf("Illegal argument");
118         }
119     } while (choice != 'Q');
120 }
121
122 /**
123 * @def Add
124 * @brief Adds a new lane
125 * @return nothing
126 */
127 void Add_Lane()
128 {
129     /**
130         * 1. Takes inputs and generates new lanes
131         * 2. The new lanes are added to the respective arrays
132         * 3. After a new lane is created, increment number of lanes
133     */
134     if (MAXLANES <= numLanes) // is the maximum number of lanes reached?
135     {
136         printf("[LOG]:Max Number of lanes created\n");
137         return;
138     }
139
140     //Utility
141     int currentLane = numLanes;
142
143     // Lane data
144     int qLaneLength = 0;
145     int qLanePar = 0;
146     char qLaneOB;
147     char qLaneDescription[MAXSTRLEN];
148
149     printf("How long is lane %i:", currentLane + 1);
150     scanf_s("%d", &qLaneLength);
151     if (qLaneLength <= 0) // flag if length is less or equal to 0
152     {
153         printf("Illegal argument..\nReturning\n\n");
154         while (getchar() != '\n');
155         return;
156     }
157     // update lane length for current lane
```

```c
    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
}

/**
* @def Display
* @brief Displays all lanes
* @return nothing
*/
```

```c
void Display_Lane()
{
    /**
        * 1. Iterate over all arrays and retieve content
        * 2. 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");
            return; //skip empty lanes
        }
        else if (laneLength[i] == 0) //flag if no data on lane
        {
            printf("No data on lane %d\n\n", i);
            return;
        }
        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);
}
```