```
1 /**
 2 * OBLIG 2
 3 *
 4 * THE PROGRAM:
 5 * -Set up variables and constants
 6 * -Create methods and functions
 7 * -Optimize code
 8 * -Use comments and documentation
9 *
10 * Ofile main.c
11 * @date 13.10.25
12 * @version 2.0
13 * @author Daniel AG
14 *
15 * Includes:
16 * - stdio.h for displaying and feeding data
17 * - stdbool.h for using booleans
18 * - ctype.h for toupper
19 */
20
21 // Internal Includes :
22 #include <stdio.h> // printf,scanf
23 #include <stdbool.h> // bool
24 #include <ctype.h> // toupper
25 #include <string.h> // strcpy,strcspn
26
27 // Constants:
28 #define MAXLANES 18 ///< Maximum number of lanes
29 #define MAXLANELENGTH 100 ///< Maximum lane length
30 #define MAXPARS 8 ///< Maximum Pars
31 #define MAXSTRLEN 100 ///< Maximum string length
32
33 //Declarations
34 void Add_Lane(int* in_numLanes, int* in_laneLengt,
       int* in_lanePar, bool* in_laneOB,
35
       char (*in_laneDescription)[MAXSTRLEN]);
36
37
38 void Display_Lane(int* in_numLanes, int* in_laneLengt,
       int* in_lanePar, bool* in_laneOB,
40
       char (*in_laneDescription)[MAXSTRLEN]);
41
42 /**
43 * main program:
44 */
45 int main()
46 {
       // Internal Includes :
47
48
       char laneDescription[MAXLANES][MAXSTRLEN] = { 0,0 }; // lane
49
                                                            //-description
```

```
...ering_H2025\PROG1001_H25\PROG1001_Oblig\Oblig2\main.c
```

```
2
```

```
int laneLength[MAXLANES] = { 0 }; // lane length
51
        int lanePar[MAXLANES] = { 0 }; // lane Par
52
       bool laneOB[MAXLANES] = { 0 }; // lane OB
53
        int numLanes = 0; // number of lanes
54
55
56
57
       // Default values :
       laneLength[0] = 62;
58
59
       lanePar[0] = 3;
60
       lane0B[0] = true;
        strcpy_s(laneDescription[0], MAXSTRLEN,
61
            "Lane with a lot of trees and scrub");
62
63
        laneLength[1] = 94;
64
65
       lanePar[1] = 3;
66
       laneOB[1] = false;
67
        strcpy_s(laneDescription[1], MAXSTRLEN,
68
            "Flat terrain thourgout the map");
69
       numLanes = 2;
70
71
        char choice; // User choice for input
72
        do /// conditional logic in while loop
73
        {
74
            /**
75
            * 1. Printing the menu
            * 2. Getting user input
76
77
            * 3. Activate switch case
            * 4. Returning to menu unless Q is pressed
78
79
            */
80
            printf("Menu Choices:\n");
            printf(" A - Add lane:\n");
81
            printf(" D - Display alle lanes:\n");
82
83
            printf(" Q - Quit:\n");
            printf(" Select a choice:");
84
85
            scanf_s(" %c", &choice, 1);
            choice = toupper(choice);
86
87
            printf("\n");
88
89
            switch (choice) /// steps into corresponding case
90
            case 'A':
91
92
93
                Add_Lane(&numLanes, laneLength, lanePar, laneOB,
                  laneDescription);
94
                break;
95
96
            case 'D':
97
            {
```

```
...ering_H2025\PROG1001_H25\PROG1001_Oblig\Oblig2\main.c
```

```
3
```

```
98
                 Display_Lane(&numLanes, laneLength, lanePar, laneOB,
                   laneDescription);
 99
                 break;
100
             }
101
            case '0':
102
                 printf("Quit - selected:\nEXITING PROGRAM");
103
104
                 break;
105
            default:
106
107
                 printf("Illegal argument");
108
109
         } while (choice != 'Q');
110 }
111
112 /**
113 *
114 * Creates a new lane by taking user input.
115 * In the function, the user is prompted to enter details,
116 * if alle details are filled correctly a new lane is created.
117 * Else it fails and returns to the menu.
118 *
119 * @param in_numLanes Pointer to number of lanes integer
120 * @param in_laneLengt Pointer to lane lengths array of integers
121 * @param in_lanePar Pointer to lane Pars array of integers
122 * @param in_laneOB Pointer to lane OB array of booleans
123 * @param in_laneDescription Pointer to lane descriptions of array of
124 *
                                 -strings
125 */
126 void Add_Lane(int* in_numLanes, int* in_laneLengt,
127
        int* in_lanePar, bool* in_laneOB,
128
        char (*in_laneDescription)[MAXSTRLEN])
129 {
130
        if (MAXLANES <= *in_numLanes) /// is the maximum</pre>
                                       /// number of lanes reached?
131
132
        {
133
             printf("[LOG]:Max Number of lanes created\n");
134
135
136
        // Temporary variables for user input
137
        int currentLane = *in_numLanes;
138
        int qLaneLength;
139
        int qLanePar;
140
        char qLaneOB;
        char qLaneDescription[MAXSTRLEN];
141
142
143
        // Take input for lane length
144
        do /// Flag if length is less or equal to 0 or input fails
145
```

```
...ering_H2025\PROG1001_H25\PROG1001_Oblig\Oblig2\main.c
```

```
4
```

```
146
             printf("How long is lane %i: ", currentLane + 1);
147
             while (getchar() != '\n');
148
         } while (scanf_s(" %d", &qLaneLength) != 1);
149
150
        if (MAXLANES <= currentLane || qLaneLength >= 0) /// Flag if max
                                                           /// -lanes is
151
                                                           /// -overreached or
152
153
                                                           /// -length is valid
154
         {
             in_laneLengt[currentLane] = qLaneLength; // Update lane length,
155
                                                      // -for current lane
156
157
        }
158
159
        // Take input for lane Pars
        do /// Flag if Pars is less than 2 or input fails
160
161
             printf("Pars on the field. Choose a number bwteen (2-8):");
162
163
             while (getchar() != '\n');
164
         } while (scanf_s(" %d", &qLanePar) != 1);
165
         if (MAXPARS < qLanePar) /// Flag if max Pars is overreached</pre>
166
             printf("Max Pars, set to MAXPARS\n");
167
168
            qLanePar = MAXPARS;
169
        if (qLanePar > 2) /// Flag if Pars is valid
170
171
             in_lanePar[currentLane] = gLanePar; // Update lane Pars for
172
               current lane
        }
173
174
175
         // Take input for lane OB
176
        do /// Flag if input is not y or n
177
             printf("Does the lane have OB (y = yes or n = no):");
178
             scanf_s(" %c", &qLaneOB, 1);
179
             while (getchar() != '\n');
180
181
             gLaneOB = toupper(gLaneOB);
182
         } while (qLaneOB != 'Y' && qLaneOB != 'N');
         in_laneOB[currentLane] = (qLaneOB == 'Y'); // Update lane OB for
183
184
                                                      // -current lane
185
        // Description
186
187
         printf("Write a description:"); // Take input for lane description
188
        fgets(qLaneDescription, MAXSTRLEN, stdin); // Read string with spaces
189
190
        // Remove newline character from string "\n"
191
         qLaneDescription[strcspn(qLaneDescription, "\n")] = 0;
192
193
        // Copy the content of qLaneDescription to laneDescription
```

```
194
        strcpy_s(in_laneDescription[*in_numLanes], MAXSTRLEN,
          qLaneDescription);
195
         (*in_numLanes)++; // Increment number of lanes
196
        printf("Lane %i added\n\n", currentLane + 1); // Confirm lane added
197
198
199 }
200
201 /**
202 *
203 * Retrieving data from all lanes for displaying.
204 * It takes pointers to all the relevant arrays and the number of lanes,
205 * then iterates through each lane to print its details.
206 *
207 *
208 * @param in_numLanes Pointer to number of lanes integer
209 * @param in_laneLengt Pointer to lane lengths array of integers
210 * @param in_lanePar Pointer to lane Pars array of integers
211 * @param in_laneOB Pointer to lane OB array of booleans
212 * @param in_laneDescription Pointer to lane descriptions of array of
213 *
                                 -strings
214 **/
215 void Display_Lane(int* in_numLanes, int* in_laneLengt, int* in_lanePar,
      bool* in_laneOB, char (*in_laneDescription)[MAXSTRLEN])
216 {
217
        int totNumPars = 0; // Total number of Pars
218
        int i = 0;
219
        if (*in_numLanes == 0) /// Flag if no lanes are available
220
221
222
            printf("No lanes available\n\n");
223
224
        if (in_laneLengt[i] == 0) /// Flag if no data on lane
225
            printf("No data on lane %d\n\n", i);
226
227
228
        do /// Iterate through all lanes only once
229
            printf("Lane \033[1;4m\%d\033[0m:\n", i + 1);
230
231
            printf("
                          Length: \033[1;4m%d meters\033[0m\n", in_laneLengt
               [i]);
232
            printf("
                          Pars: \033[1;4m%d\033[0m\n", in_lanePar[i]);
233
             printf("
                          OB: %s\n", in_laneOB[i] ?
                 "\033[1;4m With \033[0m" : "\033[1;4m Without \033[0m");
234
            if (in_laneDescription[i])
235
236
                              Description: \033[1;4m%s\033[0m\n\n",
237
                 printf("
                     in_laneDescription[i]);
238
            }
239
```

```
...ering_H2025\PR0G1001_H25\PR0G1001_Oblig\Oblig2\main.c
```

```
6
```

```
240
            i++;
241
            totNumPars += in_lanePar[i]; // Sum up total number of Pars
        } while (i != *in_numLanes);
242
243
244
        printf("Summary\n");
        printf("Total number of lanes: \033[1;4m%d\033[0m\n", *in_numLanes);
245
246
        printf("To get to Par, it requires number of throws:\033[1;4m%d\033[0m >
          \n\n",
247
            totNumPars);
248 }
249
```