

```
1  //-----
2  //      Title:MASTER DEGREE THESIS by ANTONIO SCAZZI
3  //
4  //  Description:header file with all write/read functions
5  //-----
6  #pragma once
7  #include "globalvar.h"
8
9  void LetturaConfigs();
10 void StampaConfigs(double daticonfigs[2]);
11 void CreaFile();
12 void ApriFile();
13 void StampaFile();
14 void ChiudiFile();
15
16 int AskYesNo();
17 int SelectTest();
18
19
20
21 //funzione che legge i dati dal file input
22 void LetturaConfigs()
23 {
24     configs_file;
25     err0 = fopen_s(&configs_file, "inputs/configs.txt", "r");
26     if (err0 != 0)
27     {
28         printf("Error opening configs file.\n");
29         system("pause");
30         exit(1);
31     }
32     else {
33         printf("Config file correctly loaded.\n");
```

```
34     int i = 0, max_length = 200, max_data_configs = 3;
35     char buf[200];
36     double x;
37     while (fgets(buf, max_length, configs_file) != NULL && i < max_data_configs)
38     {
39         if (buf[0] == '*')
40         {
41             continue;
42         }
43         else if (buf[0] != '*')
44         {
45             int k = sscanf_s(buf, "%lf", &x);
46             daticonfigs[i] = x;
47             i++;
48         }
49     }
50     quota_crociera = daticonfigs[0];
51     heading_crociera = daticonfigs[1];
52     initial_heading = daticonfigs[2];
53     fclose(configs_file);
54 }
55
56 }
57
58 //funzione che stampa su console i dati di configurazione
59 void StampaConfigs(double daticonfigs[2])
60 {
61     int max_data_configs = 3;
62     printf("\nData configs:\n");
63     for (int i = 0; i < max_data_configs; i++)
64     {
65         printf("%lf\n", daticonfigs[i]);
66     }
```

```
67 }
68
69 //crea due file di output uno per C2 e uno per il drone con la data e ora attuali
70 void CreaFile()
71 {
72     const char* filepath = "outputs/";
73     time_t timestamp;
74     struct tm datetime;
75     time(&timestamp);
76     char output_date[50]{};
77     if (localtime_s(&datetime, &timestamp) == 0)
78     {
79         strftime(output_date, 50, "%F_%T", &datetime);
80     }
81     strcpy_s(fileTitle, filepath);
82     strcat_s(fileTitle, "C2_");
83     strcat_s(fileTitle, output_date);
84     strcat_s(fileTitle, ".txt");
85     for (char& ch : fileTitle)
86     {
87         if (ch == ':')
88         {
89             ch = '-';
90         }
91     }
92     strcpy_s(fileTitle2, filepath);
93     strcat_s(fileTitle2, "DRONE_");
94     strcat_s(fileTitle2, output_date);
95     strcat_s(fileTitle2, ".txt");
96     for (char& ch : fileTitle2)
97     {
98         if (ch == ':')
99         {
```

```
100     ch = '-';
101     }
102     }
103     flag_created = 1;
104 }
105
106 //Apro i due file di output precedentemente creati
107 void ApriFile()
108 {
109     output_file;
110     err = fopen_s(&output_file, fileTitle, "w");
111     if (err == 0)
112     {
113         printf("Creazione del file output: ");
114         printf("%s\n", fileTitle);
115         fprintf(output_file, "*SimTime\tLatitude\tLongitude\tAltitude\tComAltitude\tPitch\t\tComPitch\tBank\t\t\tComBank\t\t\tHeading\t\tComHeading\tElevator\tRudder\t\tAileron\t\tVelocity\tComVelocity\tDOWN\t\tcomDOWN\t\t\tEAST\t\tcomEAST\t\tNORTH\t\tcomNORTH\tthrottle\taccelerationX\taccelerationY\taccelerationZ\tWindX\t\tWindY\tWindZ");
116     }
117     output_file2;
118     err2 = fopen_s(&output_file2, fileTitle2, "w");
119     if (err2 == 0)
120     {
121         printf("Creazione del file output: ");
122         printf("%s\n", fileTitle);
123         fprintf(output_file2, "*SimTime\tLatitude\tLongitude\tAltitude\tComAltitude\tPitch\t\tComPitch\tBank\t\t\tComBank\t\t\tHeading\t\tComHeading\tElevator\tRudder\t\tAileron\t\tVelocity\tComVelocity\tDOWN\t\tcomDOWN\t\t\tEAST\t\tcomEAST\t\tNORTH\t\tcomNORTH\tthrottle\taccelerationX\taccelerationY\taccelerationZ\tWindX\t\tWindY\tWindZ");
124     }
125 }
126 }
```



```
146     if (err == 0)
147     {
148         fclose(output_file);
149     }
150     if (err2 == 0)
151     {
152         fclose(output_file2);
153     }
154 }
155
156 int AskYesNo()
157 {
158     int flag_true = 0;
159     char input[2] = { 'U', '\0' };
160     printf("(Y/N): ");
161     while (input[0] == 'U') {
162
163         if (scanf_s("%1s", input, (unsigned)_countof(input)) == 1) {
164             // Converti l'input in maiuscolo per gestire diverse maiuscole/minuscole
165             input[0] = toupper(input[0]);
166
167             if (strcmp(input, "Y") == 0) {
168
169                 flag_true = 1;
170             }
171             else if (strcmp(input, "N") == 0) {
172
173                 flag_true = 0;
174             }
175             else {
176                 input[0] = 'U';
177                 printf("Input non valido, (Y/N):");
178             }
179         }
180     }
```

```
179     }
180     else {
181         // Gestisci l'errore se scanf_s non riesce a leggere l'input
182         printf("Errore nella lettura dell'input.\n");
183         // Svuota il buffer di input
184         int c;
185         while ((c = getchar()) != '\n' && c != EOF);
186         input[0] = 'U';
187     }
188 }
189
190 return flag_true;
191 }
192 int SelectTest()
193 {
194     int flag = 0;
195     char input[2] = { 'U', '\0' };
196
197     while (input[0] == 'U') {
198         printf("[1] PITCH \n[2] BANK\n[3] ALTITUDE\n[4] HEADING\n[5] VELOCITY\n[6] NORTH\n[7] EAST\n[8] DOWN\n");
199         if (scanf_s("%1s", input, (unsigned)_countof(input)) == 1) {
200             // Converti l'input in maiuscolo per gestire diverse maiuscole/minuscole
201             input[0] = toupper(input[0]);
202             if (strcmp(input, "1") == 0) {
203                 printf("Avviamento test autopilota di mantenimento del pitch\n");
204                 flag = 101;
205             }
206             else if (strcmp(input, "2") == 0) {
207                 printf("Avviamento test autopilota di mantenimento del bank\n");
208                 flag = 102;
209             }
210             else if (strcmp(input, "3") == 0) {
211                 printf("Avviamento test autopilota di mantenimento della quota\n");
```

```
212         flag = 103;
213     }
214     else if (strcmp(input, "4") == 0) {
215         printf("Avviamento test autopilota di mantenimento della direzione\n");
216         flag = 104;
217     }
218     else if (strcmp(input, "5") == 0) {
219         printf("Avviamento test autopilota di mantenimento della velocita\n");
220         flag = 105;
221     }
222     else if (strcmp(input, "6") == 0) {
223         printf("Avviamento test autopilota di mantenimento della posizione relativa NORD\n");
224         flag = 106;
225     }
226     else if (strcmp(input, "7") == 0) {
227         printf("Avviamento test autopilota di mantenimento della posizione relativa EST\n");
228         flag = 107;
229     }
230     else if (strcmp(input, "8") == 0) {
231         printf("Avviamento test autopilota di mantenimento della posizione relativa DOWN\n");
232         flag = 108;
233     }
234     else {
235         input[0] = 'U';
236         printf("Input non valido, Ripetere la scelta:\n");
237     }
238 }
239 else {
240     // Gestisci l'errore se scanf_s non riesce a leggere l'input
241     printf("Errore nella lettura dell'input.\n");
242     // Svuota il buffer di input
243     int c;
244     while ((c = getchar()) != '\n' && c != EOF);
```

```
245         input[0] = 'U';
246     }
247 }
248
249     return flag;
250 }
```