

```

1
2  /*  GENERAL INFORMATION
3  *
4  *  Project created by Sara Marfella IST188316 on May 16, 2017
5  *
6  *  main.c
7  *
8  *  Implemented:
9  *  1. Load the data structures required to represent
10 *      the stations and the trips.
11 *  2. Read parameters of main and 2 files required
12 *      to make the program work.
13 *  3. Build 2 menus and data input.
14 *  4. Build the lists of trips and stations,
15 *      inserting items in an ordered way.
16 *  5. Implementation of trips list mode.
17 *  6. Implementation of data filters.
18 *  7. Implementation of stations list mode.
19 *  8. Implementation of routes list mode.
20 *
21 *  Not Implemented:
22 *  9. Implementation of graphic mode.
23 *  10. Implementation of statistical analysis.
24 *  11. Improvements of application graphics mode,
25 *      with a real map.
26  */
27
28 #include <stdio.h>
29 #include <stdlib.h>
30 #include <time.h>
31 #include <stdbool.h>
32 #include <string.h>
33 #include "dataManager.h"
34 #include "print.h"
35
36 // EVIL GLOBAL VARIABLES
37 int filter_hour_start = -1;
38 int filter_hour_end = -1;
39
40 char trips_file[MAX_SIZE];
41 char stations_file[MAX_SIZE];
42
43 // DECLARATION OF FUNCTIONS
44
45 // command line interface
46 void mainMenu(Trip*, Station*);
47 void listOfStationsMenu(Trip *, Station *);
48 void selectDataMenu();
49 void selectStationMenu(Station*, Trip*, int);
50 void statsMenu();
51 void clearInputBuffer();
52
53 /* Main Function
54  * \param mode and file (trips file  and stations file)
55  */
56 int main (int argc, char *argv[]){
57
58     // check for correct input
59     if (argc != 4) {
60         printf("Error - required arguments missing.\n");
61         printf("Please start with -t trips-filename stations-filename\n");
62         return 0;
63     }
64     if (strcmp(argv[1], "-t") != 0) {
65         printf("Sorry - only text mode is available,");
66         printf("Please start with -t\n");

```

```

67         return 0;
68     }
69
70     strcpy(trips_file, argv[2]);
71     strcpy(stations_file, argv[3]);
72
73     Trip * allTrips = readTripsData(trips_file);
74     Station * allStations = readStationData(stations_file);
75
76     mainMenu(allTrips, allStations);
77
78     return 0;
79 }
80
81 /* Command Line Interface */
82
83 /* mainMenu: prints the main menu and handles the main menu options
84  * \param tripList      the header of the trip list
85  *                      (can be the filtered list)
86  * \param stationsList  the header of the stations list
87  */
88 void mainMenu(Trip * tripsList, Station * stationsList){
89     int command;
90     printf("\n * M A I N   M E N U * \n\n");
91     printf(" [ 1 ]   Select the data\n\n");
92     printf(" [ 2 ]   Print List of trips\n\n");
93     printf(" [ 3 ]   Print List of stations\n\n");
94     printf(" [ 4 ]   Print List of routes\n\n");
95     printf(" [ 5 ]   Print List of statistics\n\n");
96     printf(" [ 6 ]   Quit\n\n");
97     scanf("%d", &command);
98     clearInputBuffer();
99     switch (command) {
100         case 1:
101             selectDataMenu(tripsList, stationsList);
102             break;
103         case 2:
104             command = -1;
105             while ((command < 0) || (command > 32000)) {
106                 printf("How many trips do you want to print?");
107                 printf("0..32000, enter 0 for all)\n");
108                 scanf("%d", &command);
109                 clearInputBuffer();
110             }
111             printTripsList(tripsList, command);
112             mainMenu(tripsList, stationsList);
113             break;
114         case 3:
115             listOfStationsMenu(tripsList, stationsList);
116             break;
117         case 4:
118             selectStationMenu(stationsList, tripsList, 0);
119             break;
120         case 5:
121             statsMenu();
122             mainMenu(tripsList, stationsList);
123             break;
124         case 6:
125             exit(EXIT_SUCCESS);
126             break;
127         default:
128             printf("Error: invalid command...\n");
129             mainMenu(tripsList, stationsList);
130             break;
131     }
132 }

```

```

133
134
135
136 /* List of Stations Menu:   creates and prints the list of
137 *                           stations with max/min/avg
138 * \param tripList           the header of the trip list
139 *                           (usually the filtered list)
140 * \param stationsList       the header of the stations list
141 */
142 void listOfStationsMenu(Trip * tripsList, Station * stationsList){
143     Station * filteredStations = countBikes(tripsList,
144     stationsList, filter_hour_start, filter_hour_end);
145
146     // option to ask the user: limit (0 for all),
147     //should print stations with no trips (YES, NO)
148     printStationsList(filteredStations, 0, YES);
149     mainMenu(tripsList, stationsList);
150 }
151
152 /* SelectDataMenu: prints and manages the menu to set
153 *                 search criteria
154 * \param filteredTrips     the header of the trip list
155 *                           (can be the filtered list)
156 * \param allStations       the header of all stations list
157 */
158 void selectDataMenu(Trip * filteredTrips, Station * allStations){
159
160     int duration = -1, command;
161
162     printf("\n * Select the mode of your search * \n\n");
163     printf(" [ 1 ]   Period of time (hour start, hour end)\n\n");
164     printf(" [ 2 ]   Day of week\n\n");
165     printf(" [ 3 ]   Max duration of trip (in seconds)\n\n");
166     printf(" [ 4 ]   New Search (reset list)\n\n");
167     printf(" [ 5 ]   Return to Main Menu\n\n");
168     scanf("%d", &command);
169     clearInputBuffer();
170     switch (command) {
171     case 1:
172         command = -1;
173
174         while ((command < 0) || (command > 23)) {
175             printf("Insert start time of trip (hour 0..23):\n");
176             scanf("%d", &command);
177             clearInputBuffer();
178         }
179         // temporary place to store start time cannot
180         //overwrite the global now
181         // because we use it to understand if user has
182         //used the filter before, later in the code
183         int start = command;
184
185         command = -1;
186         while ((command < 0) || (command > 23)) {
187             printf("Insert end time of trip (hour 0..23):\n");
188             scanf("%d", &command);
189             clearInputBuffer();
190         }
191         filter_hour_end = command;
192
193         // if user had filtered the list before,
194         // reset the list to avoid problems
195         if (filter_hour_start != -1) {
196             // now we can save the start time
197             filter_hour_start = start;
198             Trip * allTrips = readTripsData(trips_file);

```

```

199         printf("Note: a time filter was already applied.\n");
200         printf("The trip list was reset to use the new param.\n");
201         filteredTrips = selectTripsByTime(allTrips,
202             filter_hour_start, filter_hour_end);
203     } else {
204         filter_hour_start = start;
205         filteredTrips = selectTripsByTime(filteredTrips,
206             filter_hour_start, filter_hour_end);
207     }
208     mainMenu(filteredTrips, allStations);
209     break;
210 case 2:
211     command = 0;
212     while ((command < 1) || (command > 7)) {
213         printf("\nPlease insert the day of trip:\n\n");
214         printf(" [ 1 ]   Monday\n");
215         printf(" [ 2 ]   Tuesday\n");
216         printf(" [ 3 ]   Wednesday\n");
217         printf(" [ 4 ]   Thursday\n");
218         printf(" [ 5 ]   Friday\n");
219         printf(" [ 6 ]   Saturday\n");
220         printf(" [ 7 ]   Sunday\n");
221         scanf("%d", &command);
222         clearInputBuffer();
223     }
224     filteredTrips = selectTripsByDay(filteredTrips, command);
225     mainMenu(filteredTrips, allStations);
226     break;
227 case 3:
228     while ((duration < 0) || (duration > 32767)) {
229         printf("Insert the max duration of trip (in seconds):\n");
230         scanf("%d", &duration);
231         clearInputBuffer();
232     }
233     filteredTrips = selectTripsByDuration(filteredTrips, duration);
234     mainMenu(filteredTrips, allStations);
235     break;
236 case 4:
237     filter_hour_end = -1;
238     filter_hour_start = -1;
239     Trip * allTrips = readTripsData(trips_file);
240     mainMenu(allTrips, allStations);
241     break;
242 case 5:
243     if (filteredTrips != NULL)
244         mainMenu(filteredTrips, allStations);
245     else {
246         Trip * allTrips = readTripsData(trips_file);
247         mainMenu(allTrips, allStations);
248     }
249     break;
250 default:
251     printf("Error: invalid command...\n");
252     if (filteredTrips != NULL)
253         selectDataMenu(filteredTrips, allStations);
254     else {
255         Trip * allTrips = readTripsData(trips_file);
256         selectDataMenu(allTrips, allStations);
257     }
258     break;
259 }
260 }
261
262 /* selectStationMenu:
263  * \param allStations    the header of all stations list
264  * \param filteredTrips  the header of the trips list (can be filtered)

```

```

265  * \param id           the ID of the station
266  */
267  void selectStationMenu(Station * allStations, Trip * filteredTrips, int id){
268      int command = -1;
269      char stationName[ID_SIZE] = "";
270
271      // validate id of station
272      while (strcmp(stationName, "") == 0) {
273          printf("Insert the id of the station:\n");
274          scanf("%d", &id);
275          clearInputBuffer();
276          strcpy(stationName, getStationNameById(id, allStations));
277      }
278      filteredTrips = selectTripsByIdStation(filteredTrips, id);
279
280      command = -1;
281      while ((command < 0) || (command > 32767)) {
282          printf("How many routes do yo want to print? (0 for all)\n");
283          scanf("%d", &command);
284          clearInputBuffer();
285      }
286      printRoutesList(filteredTrips, allStations, id, command);
287      Trip * allTrips = readTripsData(trips_file);
288      mainMenu(allTrips, allStations);
289  }
290
291  /* statsMenu: placeholder for the stats section
292  */
293  void statsMenu() {
294      printf("Sorry, the statistics are not yet implemented\n");
295  }
296
297  /* clearInputBuffer: handles input of multiple characters
298  * resets the input buffer to avoid double-commands
299  * Source: stackoverflow.com/questions/3969871/
300  */
301  void clearInputBuffer() // works only if the input buffer is not empty
302  {
303      char c;
304      do {
305          c = getchar();
306      } while (c != '\n' && c != EOF);
307      return;
308  }

```