```c
#include "print.h"

/* Print Functions */

/* printRoutesList: prints the list of routes
 * \param tripList              the header of the trips list
 *                              (can be filtered)
 * \param allStations           the header of all stations list
 * \param selected_station_id   the id for the station to print the routes
 * \param limit                 the number of routes to print
 */
void printRoutesList(Trip * tripList, Station * allStations,
                     int selected_station_id, int limit) {
    Route * route = createRoutesList(tripList, allStations,
                                     selected_station_id);
    printf(" Total | ID | Name    =>   ID | Name    \n");
    while (route != NULL) {
        printf("  %04d |", route->total);
        printf(" %02d | ", route->id_start_station);
        printf("%s  =>  ", route->name_start_station);
        printf(" %02d | ", route->id_final_station);
        printf("%s \n", route->name_final_station);
        route = route->next;
    }
}

/* printTripstList: prints trips list to screen
 * \param head      the header for the trip list (can be filtered)
 * \param limit     the number of trips to print
 */
void printTripsList(Trip *head, int limit) {

    struct Trip *aux = head;
    int lineCounter = 1;

    printf("ID      | Dur    | Start: Date      | St | ");
    printf("End: Date        | St | Bike ID | Type | Year | G\n");

    while (aux != NULL) {
        printf("%07ld | ", aux->id);
        printf("%06d | ", aux->duration);
        printf("%02d/%02d/%d %02d:%02d | ",
               aux->start.month, aux->start.day,
               aux->start.year, aux->start.hour, aux->start.minute);
        printf("%02d | ", aux->id_start_station);
        printf("%02d/%02d/%d %02d:%02d | ",
               aux->end.month, aux->end.day, aux->end.year,
               aux->end.hour, aux->end.minute);
        printf("%02d | ", aux->id_final_station);
        printf(" %s | ", aux->bike);
        if (aux->type == REGISTERED) {
            printf("Reg. | ");
        } else {
            printf("Cas. | ");
        }
        if (aux->year_birthday != 0) {
            printf("%04d |", aux->year_birthday);
        }
        if (aux->gender == MALE) {
            printf(" M");
        } else if (aux->gender == FEMALE) {
            printf(" F");
        }
        printf("\n");
        aux = aux->next;

        lineCounter++;

        if ((limit != 0) && (lineCounter >= limit)) {
            return;
        }
    }
    printf("\n%d trips found.\n", lineCounter);
}

/* printStationsList: prints stations list to screen
 * \param head              the header for the stations list
 * \param limit             the number of stations to print
 * \param printWithNoTrips  should print stations with no trips (YES, NO)
 */
void printStationsList(Station *head, int limit, int printWithNoTrips) {
    struct Station *aux = head;
    int lineCounter = 0;
    printf("ID | Name   | Latitude   | Longitude  | MaxIn | MinIn ");
    printf("|  Avg In | MaxOut | MinOut | Avg Out\n");

    while (aux != NULL) {

        // only print if the station has some trips, or if it should
        //print even with no trips
        if ((printWithNoTrips == 2 && aux->max_bikesIn != 0
             && aux->max_bikesOut != 0) || printWithNoTrips == 1) {

            //printf("%d\n", lineCounter);

            printf("%02d | ", aux->id);
            printf("%s | ", aux->name);
            printf(" %f | ", aux->latitude);
            printf("%f | ", aux->longitude);
            printf("  %03d | ", aux->max_bikesIn);
            printf("  %03d | ", aux->min_bikesIn);
            printf(" %06.2f | ", aux->avg_bikesIn);
            printf("   %03d | ", aux->max_bikesOut);
            printf("   %03d | ", aux->min_bikesOut);
            printf(" %06.2f \n", aux->avg_bikesOut);
        }

        aux = aux->next;
        lineCounter++;
        if ((limit != 0) && (lineCounter >= limit)) {
            return;
        }
    }
}
```