

```

1 #include <stdio.h>
2 #include <math.h>
3 #define pi 3.14159265
4
5 /*AIM : TO CALCULATE CENTER OF GRAVITY OG BUGGY
6 AND
7 THE FORCES ON THE KNUCKLE BY TAKING INPUTS FROM THE USER */
8
9 int main()
10 {
11     int requirement;
12
13     float fl, fr, rr, rl, front_total, rear_total, left_total, right_total,
14     total_mass, fmp, lmp,
15     left_right_pos, mean_trackwidth, cg_right, front_rear_pos, wheelbase,
16     cg_rear,
17     height_lifted, rear_gc, frr, flr, frm, front_axlemass_change, cg_height,
18     wheel_radius,
19     sprung_mass, unsprung_mass, front_trackwidth, rear_trackwidth, braking_dist,
20     a, b,
21     max_velocity, accl, FR, meu, vertical_load_each, vertical_centrifugal,
22     corner_radius,
23     vertical_gyroscope, net_verticalload, radius_bump, bump_height, theta_rad,
24     arc_length,
25     time_bump, vertical_velocity, vertical_KE, total_bump_force, absorbed_per,
26     scrub_radius,
27     steeringArm_len;
28
29     double theta, angle;
30
31     // _____CENTER OF GRAVITY CALCULATION_____
32
33     printf("PLEASE ENTER ALL THE VALUES AS PER UNITS MENTIONED\n\n");
34
35     printf("ENTER FRONT LEFT MASS IN KG: ");
36     scanf("%f", &fl);
37
38     printf("ENTER FRONT RIGHT MASS IN KG : ");
39     scanf("%f", &fr);
40
41     printf("ENTER REAR LEFT MASS IN KG: ");
42     scanf("%f", &rl);
43
44     printf("ENTER REAR RIGHT MASS IN KG : ");
45     scanf("%f", &rr);
46
47     printf("ENTER FRONT TRACK WIDTH IN INCH: ");
48     scanf("%f", &front_trackwidth);
49
50     printf("ENTER REAR TRACKWIDTH IN INCH : ");
51     scanf("%f", &rear_trackwidth);
52
53     printf("ENTER WHEELBASE IN INCH : ");
54     scanf("%f", &wheelbase);
55
56     printf("ENTER LIFTED HEIGHT AT REAR END IN INCH: ");
57     scanf("%f", &height_lifted);
58

```

```

53 printf("REAR GROUND CLEARANCE IN INCH: ");
54 scanf("%f", &rear_gc);
55
56 printf("ENTER RIGHT FRONT RAISED MASS IN KG: ");
57 scanf("%f", &frr);
58
59 printf("ENTER LEFT FRONT RAISED MASS IN KG: ");
60 scanf("%f", &flr);
61
62 printf("ENTER WHEEL RADIUS IN INCH: ");
63 scanf("%f", &wheel_radius);
64
65 front_total = fl + fr;
66 rear_total = rr + rl;
67 left_total = fl + rl;
68 right_total = fr + rr;
69 total_mass = rear_total + front_total;
70 mean_trackwidth=(front_trackwidth+rear_trackwidth)/2;
71
72 printf("FRONT TOTAL MASS :%f KG\n", front_total);
73 printf("REAR TOTAL MASS :%f KG\n", rear_total);
74 printf("LEFT TOTAL MASS :%f KG\n", left_total);
75 printf("right TOTAL MASS :%f KG\n\n", right_total);
76 printf("TOTAL MASS OF THE BUGGY IS :%f KG\n\n", total_mass);
77
78 fmp = front_total / total_mass * 100;
79 lmp = left_total / total_mass * 100;
80
81 printf("FRONT MASS PERCNTAGE IS : %f PERCENT\n ", fmp);
82 printf("REAR MASS PERCNTAGE IS : %f PERCENT \n\n", lmp);
83
84 left_right_pos = mean_trackwidth * (lmp / 100);
85 printf("LEFT TO RIGHT POSITION IS : %f INCH\n", left_right_pos);
86
87 cg_right = mean_trackwidth / 2 - left_right_pos;
88 printf("CG FROM CENTER LINE IS : %f INCH\n", cg_right);
89
90 front_rear_pos = wheelbase * fmp / 100;
91 cg_rear = wheelbase / 2 - front_rear_pos;
92 printf("CG FROM FROM CENTER LINE IS : %f INCH \n\n", cg_rear);
93
94 theta = asin((height_lifted - rear_gc) / wheelbase) * (180 / pi);
95 printf("theta is : %lf DEGREE \n", theta);
96
97 frm = frr + flr;
98 printf("TOTAL FRONT RAISED MASS IS : %f KG\n\n", frm);
99
100 front_axlemass_change = frm - front_total;
101 printf("THE FRONT AXLE MASS CHANGE IS : %f KG\n\n", front_axlemass_change);
102
103 // FINAL CG HEIGHT CALCULATION
104 angle = tan(theta * pi / 180);
105
106 cg_height = ((wheelbase * front_axlemass_change) / (total_mass * angle)) +
wheel_radius;
107 printf("FINAL CG HEIGHT IS :%lf INCH\n", cg_height);
108 printf("*****\n\n");
109
110 // _____KNUCKLE FORCE CALCULATION_____
111

```

```

112
113 /*converting the units from inch to meters*/
114 // {
115 wheel_radius = wheel_radius * 25.4 / 1000;
116 wheelbase = wheelbase * 25.4 / 1000;
117 cg_height = cg_height * 25.4 / 1000;
118 front_trackwidth = front_trackwidth*25.4/1000;
119 rear_trackwidth = rear_trackwidth*25.4/1000; //}
120
121 printf("PRESS 1 to get BRAKING TORQUE\nPRESS 2 to get LATERAL FORCE\nPRESS 3 to
get BUMP FORCE\nPRESS 4 TO GET FORCE ON STEERING ARM \nPRESS 5 TO GET ALL VALUES \n
");
122 printf("PLEASE ENTER A VALID NUMBER BY SEEING THE ABOVE CHART : ");
123 scanf("%d", &requirement);
124
125 if (requirement <= 0 || requirement > 5)
126 {
127     printf("ENTER A VALID NUMBER BY REFERING THE ABOVE CHART");
128     return 0;
129 }
130
131 printf("ENTER SPRUNG MASS in KG: ");
132 scanf("%f", &sprung_mass);
133
134 printf("ENTER UNSPRUNG MASS in KG: ");
135 scanf("%f", &unsprung_mass);
136
137
138 printf("ENTER BRAKING DISTANCE IN METER : ");
139 scanf("%f", &braking_dist);
140
141 printf("ENTER CG FROM FRONT OF THE VEHICLE IN METER: ");
142 scanf("%f", &a);
143
144 printf("ENTER MAX VELOCITY IN METER PER SQUARE SECOND: ");
145 scanf("%f", &max_velocity);
146
147 printf("ENTER CO-EFFICIENT OF FRICTION : ");
148 scanf("%f", &meu);
149
150 printf("*****\n\n");
151
152 // _____BRAKING TORQUE_____
153
154 if (requirement == 1)
155 {
156     printf("BRAKING TORQUE\n\n");
157
158     b = (wheelbase - a);
159     printf("THE CG FROM REAR IS in METER: %f M\n\n", b);
160
161     //-----acceleration calculation-----
162
163     printf("ACCELERATION CALCULATION\n\n");
164
165     accl = pow(max_velocity, 2) / (2 * braking_dist);
166     printf("THE ACCELERATION IS :%f M/S^2\n", accl);
167     FR = ((total_mass * 9.81 * b) + (total_mass * accl * cg_height)) /
wheelbase;
168     printf("FRONT REACTION FORCE IS :%f N\n", FR);

```

```

169     printf("THE REACTION FORCE FOR ONE WHEEL IS %f N\n", FR / 2);
170
171     //----- VERTICAL LOAD DUE TO UNSPRUNG MASS-----
172
173     printf("  VERTICAL LOAD DUE TO UNSPRUNG MASS      ");
174
175     printf("TOTAL VERTICAL LOAD ON ONE WHEEL IS : %f N\n", FR / 2 +
(unsprung_mass / 4 * 9.81));
176
177     printf("THE FRICTIONAL FORCE IS %f N\n", meu * (FR / 2 + (unsprung_mass / 4
* 9.81)));
178
179     printf("THE BRAKING TORQUE is %f N-M\n", (meu * (FR / 2 + (unsprung_mass / 4
* 9.81))) * wheel_radius);
180     printf("*****\n\n");
181 }
182
183 // _____LATERAL FORCE_____
184
185 else if (requirement == 2)
186 {
187     printf("LATERAL FORCE\n\n");
188
189     printf("enter corner radius in meter: ");
190     scanf("%f", &corner_radius);
191
192     vertical_load_each = front_total * 9.81 / 2;
193     printf("VERTICAL LOAD ON EACH WHEEL is %f N\n", vertical_load_each);
194
195     vertical_centrifugal = (total_mass * pow(max_velocity, 2) * cg_height) / (2
* corner_radius * front_trackwidth);
196     printf("VERTICAL FORCE DUE TO CENTRIFUGAL FORCE IS :%f N \n",
vertical_centrifugal);
197
198     vertical_gyroscope = 4 * (((front_total / 2) * (pow(wheel_radius, 2) / 2)) *
((pow(max_velocity, 2)) / (wheel_radius * corner_radius)));
199     printf("VERTICAL FORCE DUE TO GYROSCOPIC EFFECT IS :%f N\n",
vertical_gyroscope);
200
201     net_verticalLoad = vertical_load_each + vertical_centrifugal +
vertical_gyroscope;
202     printf("NET VERTICAL LOAD ON EACH WHEEL IS :%f N\n\n", net_verticalLoad);
203
204     printf("LATERAL FORCE ON EACH WHEEL IS : %f N \n ", meu * net_verticalLoad);
205     printf("*****\n\n");
206 }
207
208 // _____BUMP FORCE_____
209
210 else if (requirement == 3)
211 {
212     printf("BUMP FORCE\n\n");
213
214     printf("ENTER BUMP RADIUS IN METER: ");
215     scanf("%f", &radius_bump);
216
217     printf("ENTER BUMP height IN METER : ");
218     scanf("%f", &bump_height);
219
220     printf("ENTER BUMP theta in RADIANS : ");

```

```

221     scanf("%f", &theta_rad);
222
223     printf("ENTER THE TIME TAKEN TO CROSS THE BUMP: ");
224     scanf("%f", &time_bump);
225
226     printf("ENTER vertical velocity IN METER PER SQUARE SECOND: ");
227     scanf("%f", &vertical_velocity);
228
229     printf("ENTER THE ASSUMED ABSORBED FORCE BY SUSPENSION SYSTEM IN PERCENTAGE
");
230     scanf("%f", &absorbed_per);
231
232     arc_length = radius_bump * theta_rad;
233     printf("THE ARC LENGTH OF BUMP IS :%f M\n ", arc_length);
234     absorbed_per = 100 - absorbed_per;
235
236     vertical_KE = 0.5 * (front_total / 2) * pow(vertical_velocity, 2);
237     printf("VERTICAL KINETIC ENERGY IS : %f JOULES\n", vertical_KE);
238     total_bump_force = vertical_KE / bump_height;
239
240     //-----ASSUMING 70% OF THE FORCE AS DAMPED BY ARMS , KNUCKLE AND
DAMPER,MOUNT;
241
242     printf("RESULTANT FORCE DUE TO BUMP : %f N\n", absorbed_per / 100 *
total_bump_force);
243     printf("*****\n\n");
244 }
245
246 //-----FORCE ON STEERING ARM-----
247
248 else if (requirement == 4)
249 {
250     printf("FORCE ON STEERING ARM\n\n");
251
252     printf("ENTER SCRUB RADIUS IN MM ");
253     scanf("%f", &scrub_radius);
254
255     printf("ENTER STEERING ARM LENGTH IN MM : ");
256     scanf("%f", &steeringArm_len);
257
258     printf("STATIC LOAD ON ONE WHEEL IS :%f N\n", vertical_load_each);
259     printf("FRICITONAL FORCE IS :%f N\n", meu * vertical_load_each);
260     printf("TORQUE ACTING ABOUT STEERING AXIS is :%f N/mm\n", meu *
vertical_load_each * scrub_radius);
261     printf("FORCE ACTING ON TIE ROD IS : %f N", (meu * vertical_load_each *
scrub_radius) / steeringArm_len);
262 }
263
264 /* IF THE USER INPUTS 5 AS REQUIREMENT ALL THE PARAMETER WILL BE CALCULATED */
265 else
266 {
267     printf("BRAKING TORQUE\n\n");
268
269     b = (wheelbase - a);
270     printf("THE CG FROM REAR IS in METER: %f M\n", b);
271
272     //-----acceleration calculation-----
273
274     printf("ACCELERATION CALCULATION\n");
275

```

```

276     accl = pow(max_velocity, 2) / (2 * braking_dist);
277     printf("THE ACCELERATION IS :%f M/S^2\n", accl);
278     FR = ((total_mass * 9.81 * b) + (total_mass * accl * cg_height)) /
wheelbase;
279     printf("FRONT REACTION FORCE IS :%f N\n", FR);
280     printf("THE REACTION FORCE FOR ONE WHEEL IS %f N\n", FR / 2);
281
282     //----- VERTICAL LOAD DUE TO UNSPRUNG MASS-----
283
284     printf(" VERTICAL LOAD DUE TO UNSPRUNG MASS ");
285
286     printf("TOTAL VERTICAL LOAD ON ONE WHEEL IS : %f N\n", FR / 2 +
(unsprung_mass / 4 * 9.81));
287
288     printf("THE FRICTIONAL FORCE IS %f N\n", meu * (FR / 2 + (unsprung_mass / 4
* 9.81)));
289
290     printf("THE BRAKING TORQUE is %f N-M\n", (meu * (FR / 2 + (unsprung_mass / 4
* 9.81))) * wheel_radius);
291     printf("*****\n\n");
292
293     // _____ LATERAL FORCE _____
294
295     printf("LATERAL FORCE\n\n");
296
297     printf("enter corner radius in meter: ");
298     scanf("%f", &corner_radius);
299
300     vertical_load_each = front_total * 9.81 / 2;
301     printf("VERTICAL LOAD ON EACH WHEEL is %f N\n", vertical_load_each);
302
303     vertical_centrifugal = (total_mass * pow(max_velocity, 2) * cg_height) / (2
* corner_radius * front_trackwidth);
304     printf("VERTICAL FORCE DUE TO CENTRIFUGAL FORCE IS :%f N \n",
vertical_centrifugal);
305
306     vertical_gyroscope = 4 * (((front_total / 2) * (pow(wheel_radius, 2) / 2)) *
((pow(max_velocity, 2)) / (wheel_radius * corner_radius)));
307     printf("VERTICAL FORCE DUE TO GYROSCOPIC EFFECT IS :%f N\n",
vertical_gyroscope);
308
309     net_verticalLoad = vertical_load_each + vertical_centrifugal +
vertical_gyroscope;
310     printf("NET VERTICAL LOAD ON EACH WHEEL IS :%f N\n\n", net_verticalLoad);
311
312     printf("LATERAL FORCE ON EACH WHEEL IS : %f N \n ", meu * net_verticalLoad);
313     printf("*****\n\n");
314
315     // _____ BUMP FORCE _____
316
317     printf("BUMP FORCE\n\n");
318
319     printf("ENTER BUMP RADIUS IN METER: ");
320     scanf("%f", &radius_bump);
321
322     printf("ENTER BUMP height IN METER: ");
323     scanf("%f", &bump_height);
324
325     printf("ENTER BUMP theta in RADIANS : ");
326     scanf("%f", &theta_rad);

```

```

327
328     printf("ENTER THE TIME TAKEN TO CROSS THE BUMP: ");
329     scanf("%f", &time_bump);
330
331     printf("ENTER vertical velocity IN METER PER SQUARE SECOND: ");
332     scanf("%f", &vertical_velocity);
333
334     printf("ENTER THE ASSUMED ABSORBED FORCE BY SUSPENSION SYSTEM IN PERCENTAGE
");
335     scanf("%f", &absorbed_per);
336
337     arc_length = radius_bump * theta_rad;
338     printf("THE ARC LENGTH OF BUMP IS :%f M\n ", arc_length);
339     absorbed_per = 100 - absorbed_per;
340
341     vertical_KE = 0.5 * (front_total / 2) * pow(vertical_velocity, 2);
342     printf("VERTICAL KINETIC ENERGY IS : %f JOULES\n", vertical_KE);
343     total_bump_force = vertical_KE / bump_height;
344     //_____ -ASSUMING 70% OF THE FORCE AS DAMPED BY ARMS , KNUCKLE AND
DAMPER,MOUNT;
345
346     printf("RESULTANT FORCE DUE TO BUMP : %f N\n", absorbed_per / 100 *
total_bump_force);
347     printf("*****\n\n");
348
349     //_____FORCE ON STEERING ARM_____
350
351     printf("FORCE ON STEERING ARM\n\n");
352
353     printf("ENTER SCRUB RADIUS IN MM ");
354     scanf("%f", &scrub_radius);
355
356     printf("ENTER STEERING ARM LENGTH IN MM : ");
357     scanf("%f", &steeringArm_len);
358
359     printf("STATIC LOAD ON ONE WHEEL IS :%f N\n", vertical_load_each);
360     printf("FRICITONAL FORCE IS :%f N\n", meu * vertical_load_each);
361     printf("TORQUE ACTING ABOUT STEERING AXIS is :%f N/mm\n", meu *
vertical_load_each * scrub_radius);
362     printf("FORCE ACTING ON TIE ROD IS : %f N", (meu * vertical_load_each *
scrub_radius) / steeringArm_len);
363 }
364 return 0;
365 }

```