```c
#include <stdio.h>
#include <math.h>
#define pi 3.14159265

/*AIM : TO CALCULATE CENTER OF GRAVITY OG BUGGY
AND
THE FORCES ON THE KNUCKLE BY TAKING INPUTS FROMT THE USER */

int main()
{
    int requirement;

    float fl, fr, rr, rl, front_total, rear_total, left_total, right_total, total_mass, fmp, lmp,
        left_right_pos, mean_trackwidth, cg_right, front_rear_pos, wheelbase, cg_rear,
        height_lifted, rear_gc, frr, flr, frm, front_axlemass_change, cg_height, wheel_radius,
        sprung_mass, unsprung_mass, front_trackwidth, rear_trackwidth, braking_dist, a, b,
        max_velocity, accl, FR, meu, vertical_load_each, vertical_centrifugal, corner_radius,
        vertical_gyroscope, net_verticalLoad, radius_bump, bump_height, theta_rad, arc_length,
        time_bump, vertical_velocity, vertical_KE, total_bump_force, absorbed_per, scrub_radius,
        steeringArm_len;

    double theta, angle;

    // ___________CENTER OF GRAVITY CALCULATION_____________________

    printf("PLEASE ENTER ALL THE VALUES AS PER UNITS MENTIONED\n\n");

    printf("ENTER FRONT LEFT MASS IN KG: ");
    scanf("%f", &fl);

    printf("ENTER FRONT RIGHT MASS IN KG : ");
    scanf("%f", &fr);

    printf("ENTER REAR LEFT MASS IN KG: ");
    scanf("%f", &rl);

    printf("ENTER REAR RIGHT MASS IN KG : ");
    scanf("%f", &rr);

    printf("ENTER FRONT TRACK WIDTH IN INCH: ");
    scanf("%f", &front_trackwidth);

    printf("ENTER REAR TRACKWIDTH IN INCH : ");
    scanf("%f", &rear_trackwidth);


    printf("ENTER WHEELBASE IN INCH : ");
    scanf("%f", &wheelbase);

    printf("ENTER LIFTED HEIGHT AT REAR END IN INCH: ");
    scanf("%f", &height_lifted);

```

```c
    printf("REAR GROUND CLEARANCE IN INCH: ");
    scanf("%f", &rear_gc);

    printf("ENTER RIGHT FRONT RAISED MASS IN KG: ");
    scanf("%f", &frr);

    printf("ENTER LEFT FRONT RAISED MASS IN KG: ");
    scanf("%f", &flr);

    printf("ENTER WHEEL RADIUS IN INCH: ");
    scanf("%f", &wheel_radius);

    front_total = fl + fr;
    rear_total = rr + rl;
    left_total = fl + rl;
    right_total = fr + rr;
    total_mass = rear_total + front_total;
    mean_trackwidth=(front_trackwidth+rear_trackwidth)/2;

    printf("FRONT TOTAL MASS :%f KG\n", front_total);
    printf("REAR TOTAL MASS :%f KG\n", rear_total);
    printf("LEFT TOTAL MASS :%f KG\n", left_total);
    printf("right TOTAL MASS :%f KG\n\n", right_total);
    printf("TOTAL MASS OF THE BUGGY IS :%f KG\n\n", total_mass);

    fmp = front_total / total_mass * 100;
    lmp = left_total / total_mass * 100;

    printf("FRONT MASS PERCNTAGE IS : %f PERCENT\n ", fmp);
    printf("REAR MASS PERCNTAGE IS : %f PERCENT \n\n", lmp);

    left_right_pos = mean_trackwidth * (lmp / 100);
    printf("LEFT TO RIGHT POSITION IS : %f INCH\n", left_right_pos);

    cg_right = mean_trackwidth / 2 - left_right_pos;
    printf("CG FROM CENTER LINE IS : %f INCH\n", cg_right);

    front_rear_pos = wheelbase * fmp / 100;
    cg_rear = wheelbase / 2 - front_rear_pos;
    printf("CG FROM FROM CENTER LINE IS : %f INCH \n\n", cg_rear);

    theta = asin((height_lifted - rear_gc) / wheelbase) * (180 / pi);
    printf("theta is : %lf DEGREE \n", theta);

    frm = frr + flr;
    printf("TOTAL FRONT RAISED MASS IS : %f KG\n\n", frm);

    front_axlemass_change = frm - front_total;
    printf("THE FRONT AXLE MASS CHANGE IS : %f KG\n\n", front_axlemass_change);

    // FINAL CG HEIGHT CALCULATION
    angle = tan(theta * pi / 180);

    cg_height = ((wheelbase * front_axlemass_change) / (total_mass * angle)) +
wheel_radius;
    printf("FINAL CG HEIGHT IS :%lf INCH\n", cg_height);
    printf("************************************************************\n\n");

    // ____________________KNUCKLE FORCE CALCULATION_____________________
```

```c

    /*converting the units from inch to meters*/
    // {
    wheel_radius = wheel_radius * 25.4 / 1000;
    wheelbase = wheelbase * 25.4 / 1000;
    cg_height = cg_height * 25.4 / 1000;
    front_trackwidth = front_trackwidth*25.4/1000;
    rear_trackwidth = rear_trackwidth*25.4/1000; //}

    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");
    printf("PLEASE ENTER A VALID NUMBER BY SEEING THE ABOVE CHART : ");
    scanf("%d", &requirement);

    if (requirement <= 0 || requirement > 5)
    {
        printf("ENTER A VALID NUMBER BY REFERING THE ABOVE CHART");
        return 0;
    }

    printf("ENTER SPRUNG MASS in  KG: ");
    scanf("%f", &sprung_mass);

    printf("ENTER UNSPRUNG MASS in  KG: ");
    scanf("%f", &unsprung_mass);


    printf("ENTER BRAKING DISTANCE IN METER : ");
    scanf("%f", &braking_dist);

    printf("ENTER CG FROM FRONT OF THE VEHICLE IN METER: ");
    scanf("%f", &a);

    printf("ENTER MAX VELOCITY IN METER PER SQUARE SECOND: ");
    scanf("%f", &max_velocity);

    printf("ENTER CO-EFFICIENT OF FRICTION : ");
    scanf("%f", &meu);

    printf("************************************************************\n\n");

    // _________________BRAKING TORQUE_____________________________

    if (requirement == 1)
    {
        printf("BRAKING TORQUE\n\n");

        b = (wheelbase - a);
        printf("THE CG FROM REAR IS in METER: %f M\n\n", b);

        //-----------------acceleration calculation-----------------------

        printf("ACCELERATION CALCULATION\n\n");

        accl = pow(max_velocity, 2) / (2 * braking_dist);
        printf("THE ACCELERATION IS :%f M/S^2\n", accl);
        FR = ((total_mass * 9.81 * b) + (total_mass * accl * cg_height)) / wheelbase;
        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 : ");
```

```
        scanf("%f", &theta_rad);

        printf("ENTER THE TIME TAKEN TO CROSS THE BUMP: ");
        scanf("%f", &time_bump);

        printf("ENTER vertical velocity IN METER PER SQUARE SECOND: ");
        scanf("%f", &vertical_velocity);

        printf("ENTER THE ASSUMED ABSORBED FORCE BY SUSPENSION SYSTEM IN PERCENTAGE
");
        scanf("%f", &absorbed_per);

        arc_length = radius_bump * theta_rad;
        printf("THE ARC LENGTH OF BUMP IS :%f M\n ", arc_length);
        absorbed_per = 100 - absorbed_per;

        vertical_KE = 0.5 * (front_total / 2) * pow(vertical_velocity, 2);
        printf("VERTICAL KINETIC ENERGY IS : %f JOULES\n", vertical_KE);
        total_bump_force = vertical_KE / bump_height;

        //_______-ASSUMING 70% OF THE FORCE AS DAMPED BY ARMS , KNUCKLE AND
DAMPER,MOUNT;

        printf("RESULTANT FORCE DUE TO BUMP : %f N\n", absorbed_per / 100 *
total_bump_force);
        printf("************************************************************\n\n");
    }

    //____________________FORCE ON STEERING ARM_____________________________

    else if (requirement == 4)
    {
        printf("FORCE ON STEERING ARM\n\n");

        printf("ENTER SCRUB RADIUS IN MM ");
        scanf("%f", &scrub_radius);

        printf("ENTER STEERING ARM LENGTH IN MM : ");
        scanf("%f", &steeringArm_len);

        printf("STATIC LOAD ON ONE WHEEL IS :%f N\n", vertical_load_each);
        printf("FRICITONAL FORCE IS :%f N\n", meu * vertical_load_each);
        printf("TORQUE ACTING ABOUT STEERING AXIS is :%f N/mm\n", meu *
vertical_load_each * scrub_radius);
        printf("FORCE ACTING ON TIE ROD IS : %f N", (meu * vertical_load_each *
scrub_radius) / steeringArm_len);
    }

    /* IF THE USER INPUTS 5 AS REQUIREMENT ALL THE PARAMETER WILL BE CALCULATED */
    else
    {
        printf("BRAKING TORQUE\n\n");

        b = (wheelbase - a);
        printf("THE CG FROM REAR IS in METER: %f M\n", b);

        //-----------------acceleration calculation-----------------------

        printf("ACCELERATION CALCULATION\n");

```

```
        accl = pow(max_velocity, 2) / (2 * braking_dist);
        printf("THE ACCELERATION IS :%f M/S^2\n", accl);
        FR = ((total_mass * 9.81 * b) + (total_mass * accl * cg_height)) /
wheelbase;
        printf("FRONT REACTION FORCE IS :%f N\n", FR);
        printf("THE REACTION FORCE FOR ONE WHEEL IS %f N\n", FR / 2);

        //------------------ VERTICAL LOAD DUE TO UNSPRUNG MASS-------------------

        printf("  VERTICAL LOAD DUE TO UNSPRUNG MASS       ");

        printf("TOTAL VERTICAL LOAD ON ONE WHEEL IS : %f N\n", FR / 2 +
(unsprung_mass / 4 * 9.81));

        printf("THE FRICTIONAL FORCE IS %f N\n", meu * (FR / 2 + (unsprung_mass / 4
* 9.81)));

        printf("THE BRAKING TORQUE is %f N-M\n", (meu * (FR / 2 + (unsprung_mass / 4
* 9.81))) * wheel_radius);
        printf("*************************************************************\n\n");

        // _____________________LATERAL FORCE_____________________________

        printf("LATERAL FORCE\n\n");

        printf("enter corner radius in meter: ");
        scanf("%f", &corner_radius);

        vertical_load_each = front_total * 9.81 / 2;
        printf("VERTICAL LOAD ON EACH WHEEL is %f N\n", vertical_load_each);

        vertical_centrifugal = (total_mass * pow(max_velocity, 2) * cg_height) / (2
* corner_radius * front_trackwidth);
        printf("VERTICAL FORCE DUE TO CENTRIFUGAL FORCE IS :%f N \n",
vertical_centrifugal);

        vertical_gyroscope = 4 * (((front_total / 2) * (pow(wheel_radius, 2) / 2)) *
((pow(max_velocity, 2)) / (wheel_radius * corner_radius)));
        printf("VERTICAL FORCE DUE TO GYROSCOPIC EFFECT IS :%f N\n",
vertical_gyroscope);

        net_verticalLoad = vertical_load_each + vertical_centrifugal +
vertical_gyroscope;
        printf("NET VERTICAL LOAD ON EACH WHEEL IS :%f N\n\n", net_verticalLoad);

        printf("LATERAL FORCE ON EACH WHEEL IS : %f N \n ", meu * net_verticalLoad);
        printf("*************************************************************\n\n");

        // _____________________BUMP FORCE_____________________________

        printf("BUMP FORCE\n\n");

        printf("ENTER BUMP RADIUS IN METER: ");
        scanf("%f", &radius_bump);

        printf("ENTER BUMP height IN METER: ");
        scanf("%f", &bump_height);

        printf("ENTER  BUMP theta in RADIANS : ");
        scanf("%f", &theta_rad);
```

```c

        printf("ENTER THE TIME TAKEN TO CROSS THE BUMP: ");
        scanf("%f", &time_bump);

        printf("ENTER vertical velocity IN METER PER SQUARE SECOND: ");
        scanf("%f", &vertical_velocity);

        printf("ENTER THE ASSUMED ABSORBED FORCE BY SUSPENSION SYSTEM IN PERCENTAGE
");
        scanf("%f", &absorbed_per);

        arc_length = radius_bump * theta_rad;
        printf("THE ARC LENGTH OF BUMP IS :%f M\n ", arc_length);
        absorbed_per = 100 - absorbed_per;

        vertical_KE = 0.5 * (front_total / 2) * pow(vertical_velocity, 2);
        printf("VERTICAL KINETIC ENERGY IS : %f JOULES\n", vertical_KE);
        total_bump_force = vertical_KE / bump_height;
        //_______-ASSUMING 70% OF THE FORCE AS DAMPED BY ARMS , KNUCKLE AND
DAMPER,MOUNT;

        printf("RESULTANT FORCE DUE TO BUMP : %f N\n", absorbed_per / 100 *
total_bump_force);
        printf("************************************************************\n\n");

        //____________________FORCE ON STEERING ARM_____________________________

        printf("FORCE ON STEERING ARM\n\n");

        printf("ENTER SCRUB RADIUS IN MM ");
        scanf("%f", &scrub_radius);

        printf("ENTER STEERING ARM LENGTH IN MM : ");
        scanf("%f", &steeringArm_len);

        printf("STATIC LOAD ON ONE WHEEL IS :%f N\n", vertical_load_each);
        printf("FRICITONAL FORCE IS :%f N\n", meu * vertical_load_each);
        printf("TORQUE ACTING ABOUT STEERING AXIS is :%f N/mm\n", meu *
vertical_load_each * scrub_radius);
        printf("FORCE ACTING ON TIE ROD IS : %f N", (meu * vertical_load_each *
scrub_radius) / steeringArm_len);
    }
    return 0;
}
```