```
#include <bits/stdc++.h>
using namespace std;

#define lat_1 12.9611159 // Latitude of customer
#define lon_1 77.6362214 // Longitude of customer

#define pi 3.14159265358979323846
#define earth_radius 6371.0

 ifstream customer_list ("customers.json");
ofstream out ("sort.json");

double degtorad(double deg) // Function to convert degree to radian.
{
   return ( deg * pi / 180);
}

double customerdistance(double lat_2, double lon_2)
{

   double lat1, lon1, lat2, lon2,  delta_lon, central_ang;
   lat1 = degtorad(lat_1);
   lon1 = degtorad(lon_1);
   lat2 = degtorad(lat_2);
   lon2 = degtorad(lon_2);

   delta_lon = lon2 - lon1;

   // great circle distance formula.

   central_ang = acos ( sin(lat1) *  sin(lat2) + cos(lat1) * cos(lat2) * cos(delta_lon) );
   return (earth_radius * central_ang);
}

struct json
{
   long long int length, i, j, x, y, m,  n, f, friends, id[100000];
   char latitude_as_string[1000],  longitude_as_string[1000],  id_as_string[1000],
name[1000];

    double lat_2, lon_2;
   string line;

   void distance_calculator()

   {

      if (distanceEarth(lat_2, lon_2) <= 50.0000)
```

```
    {

        id[i] = atoll(id_as_string); // Converting id to int format.
        i++;

        out << "{\"user_id\": " << id[i - 1] <<  ", \"name\": " << name << "}" << endl;

    }

}

void file_parser()
{
    if (customer_list.is_open())
    {
        while (getline(customer_list, line))
        {
            f = 0; x = 0; y = 0; friends = 0; m = 0, n = 0;
            length = line.size();

            for (j = 0; j < length; j++)
            {
                if (line[j] == '"')
                    f++;

                else if (line[j] == ':')
                    fi++;
            // To get latitude of the location.
                if (f == 3)
                {
                    j++;
                    while (line[j] != '"')
                    {
                        latitude_as_string[x] = line[j];
                        x++; j++;
                    }
                    j--; latitude_as_string[x] = '\0';
                }
                // To get longitude of the location.
                else if (f == 13)
                {
                    j++;
                    while (line[j] != '"')
                    {
                        longitude_as_string[y] = line[j];
                        y++; j++;
                    }
```

```
                j--; longitude_as_string[y] = '\0';
            }

            // To get id of the friend.
            if (fi == 2)
            {
                j += 2;
               while (line[j] != ',')
                {
                   id_as_string[m] = line[j];
                    m++; j++;
                }
                j--; id_as_string[m] = '\0';
                fi++;
            }

            // To get name of the friend.
            else if (fi == 4)
            {
                j += 2;
                while (line[j] != ',')
                {
                    name[n] = line[j];
                    n++; j++;
                }
                  j--; name[n] = '\0';
                fi++; f += 2;
            }
        }


        lat_2 = atof(latitude_as_string);

        lon_2 = atof(longitude_as_string);

        distance_calculator();

      }

    }

    customer_list.close();
    out.close();

  }
};
```

```
int main()
{

    // Creating object of the structure json
    json obj;

    // To read customers.json file.
    obj.file_parser();

    return 0;
}
```