```
/*
 *  Specifies the driver library to be used,
 *  important to signifiy the HW model of sensor.
 *  (Bosch, LSM, NXP)
 */

typedef enum DriverLibrary {
        BSH = 0,
        LSM = 1,
        NXP = 2
} DriverLibrary;

/*
 *  Specifies the type of sensor to be:
 *  Gyroscoepe, Magnetometer or Accelerometer
 */

typedef enum SensorType {
        Gyr = 0,
        Acc = 1,
        Mag = 2
} SensorType;

/*
 *  Object: SensorConfig
 *
 *  Params:
 *    - id: I2C address of sensor (unique)
 *    - sensor_type: specifies acc, gyr or mag type
 *    - driver_library: sensor manufacturer, for driver support
 *    - x_offset: degree tilt on x axis of hw sensor (+90, -90, 0, +180)
 *    - y_offset: degree tilt on y axis of hw sensor (+90, -90, 0, +180)
 *    - z_offset: degree tilt on z axis of hw sensor (+90, -90, 0, +180)
 *
 *  Description:
 *    Specifies HW config of sensor in SW.
 */

typedef struct SensorConfig {
        unsigned int id;
        unsigned int addr;
        SensorType sensor_type;
        DriverLibrary driver_library;
        int x_offset;
        int y_offset;
        int z_offset;
} SensorConfig;

/*
 *  Object: SensorRead
 *
 *  Params:
 *    - x: x axis value of hw sensor read
 *    - y: y axis value of hw sensor read
 *    - z: z axis value of hw sensor read
 *    - time: epoch time (ms since epoch)
 *    - sensor: reference to sensor config that
 *              has been read
 *
 *  Description:
 *    This object contains information regarding the reading of a specific
 *    hw sensor.
 */

typedef struct SensorRead {
        double x;
        double y;
        double z;
        unsigned int time;
```

```
        SensorConfig *sensor;
} SensorRead;

/*
 *  Function: read_sensors
 *
 *  Params:
 *    - sensor_count: int pointer that will be updated to the number of
 *                    sensors currently configured in the HW setup
 *    - sensor_readings: SensorRead array pointer populated to size
 *                    sensor_count of most recent SensorRead for each
 *                    HW sensor
 *
 *  Description:
 *    Takes in two pointer references and assigns them accordingly. Read
 *    parameter descriptions for each to see how the references are
 *    populated.
 *
 */
void read_sensors(int *sensor_count,SensorRead *sensor_readings);

void clear_sensors();
void load_sensors(char* configFile);
```