```
1: //implemented in startup.c
2:     /*
3:     Author: Stefan Mathies
4:     No parameters.
5:     Returns a float corresponding to the distance the user entered on the screen.
6:     */
7:     float getUserDistance();
8:
9:     /*
10:    Author: Emily D'Silva
11:    No parameters.
12:    Initializes and resets sensors, then returns 1 if it succeeded and 0 if it failed.
13:    */
14:    void initializeSensors();
15:
16:
17: //implemented in logger.c
18:    #include "fileLib/PC_FileIO.c" //includes the file library provided by the teaching team
19:
20:    /*
21:    Author: Samuel Mailhot
22:    No parameters.
23:    Creates, then returns, a logfile with the proper name.
24:    */
25:    TFileHandle prepLog();
26:
27:    /*
28:    Author: Kiran Ghanekar
29:    Parameters: int time (the log's timestamp)
30:                            string msg (the message to be written)
31:                            (optional) float numarg or int numarg
32:                                (a numerical argument to log, may or may not be needed)
33:    No returns. Writes a log message to the file returned by prepLog().
34:    */
35:    void sendLog(TFileHandle &logfile, int &time, string &mesg);
36:    void sendLog(TFileHandle &logfile, int &time, string &mesg, float &numarg);
37:    void sendLog(TFileHandle &logfile, int &time, string &mesg, int &numarg);
38:
39:
40: //implemented in tasks.c
41:    /*
42:    Author: Emily D'Silva
43:    Parameters: float dist (the distance to drive)
44:                bool direction (1 for forward, 0 for backward)
45:                bool toStop (1 to stop at the end, 0 to keep moving)
46:                int speed (between 0 and 100)
47:                float &currentdist (reference to a variable containing
48:                                    how far the robot has advanced)
49:                int &time (reference to an integer corresponding to the current time value)
50:                TFileHandle &logfile (reference to a logfile to write to)
51:    Returns whether the robot was or was not moving after driving 'dist' in 'direction'.
52:    */
53:    bool drive(float dist, bool direction, bool toStop, int speed,
54:               float &currentdist, int &time, TFileHandle &logfile);
55:
56:    /*
57:    Author: Samuel Mailhot
58:    Parameters: int &pastRotations (reference to a variable holding the number of
59:                                    times the lead screw has already turned)
60:                bool spinDown (if true, spin the leadscrew all the way back to 0 rotations)
61:    No returns - rotates the lead screw a certain amount based on
```

```
62:                  pastRotations to tension the wheels further.
63:     */
64:     void tensionWheels(int &pastRotations, bool spinDown);
65:
66:     /*
67:     Author: Samuel Mailhot
68:     Parameters: float &currentdist (reference to a variable containing
69:                                     how far the robot has advanced)
70:                 int &time (reference to an integer corresponding to the current time value)
71:                 TFileHandle &logfile (reference to a logfile to write to)
72:     Returns 1 if the procedure succeeded, and 0 if it failed.
73:     */
74:     bool clean(float &currentdist, int &time, TFileHandle &logfile);
75:
76:     /*
77:     Author: Stefan Mathies
78:     Parameters: float &currentdist (reference to a variable containing
79:                                     how far the robot has advanced)
80:                 int &time (reference to an integer corresponding to the current time value)
81:                 TFileHandle &logfile (reference to a logfile to write to)
82:     No returns - runs the escape procedure in accordance with the flowchart.
83:     */
84:     void escape(float &currentdist, int &time, TFileHandle &logfile);
85:
86:     /*
87:     Author: Stefan Mathies
88:     Parameters: int &pastRotations (reference to a variable holding the number of
89:                                     times the lead screw has already turned)
90:                 int &time (reference to an integer corresponding to the current time value)
91:                 TFileHandle &logfile (reference to a logfile to write to)
92:     No returns - runs the shutdown procedure in accordance with the flowchart.
93:     */
94:     void shutdown(int &pastRotations, int &time, TFileHandle &logfile);
95:
96:
97: //implemented in checks.c
98:     /*
99:     Author: Kiran Ghanekar
100:    No parameters.
101:    Returns whether or not a blockage is within range or not.
102:    */
103:    bool ultrasonicDist();
104:
105:    /*
106:    Author: Kiran Ghanekar
107:    Parameters: float &currentdist (reference to a variable holding the distance
108:                                    moved by the robot since power-on)
109:                float &endpoint (reference to the user-input endpoint)
110:                bool didDrive (whether or not the robot moved, from the accelerometer)
111:                int &failures (reference to a variable containing the number of
112:                               health checks that have already failed)
113:                float drivedist (the distance to drive before each health check)
114:                int &time (reference to the current time counter)
115:                TFileHandle &logfile (reference to a logfile to write to)
116:    Returns: 0 if everything passes
117:             1 if the robot should start to leave the pipe
118:             5 if the robot should start cleaning operations
119:             10 if the robot should tension the wheels more
120:    */
121:    int healthCheck(float &currentdist, float &endpoint, bool didDrive, int &failures,
122:                    float drivedist, int &time, TFileHandle &logfile);
```