You are creating a portable vital sign monitoring device that measures a user’s heart rate and temperature and then responds accordingly. When the user’s vital signs are normal the device is quiet. If the vital signs are off a bit the device will alert the user. If the vital signs become serious the device will alert a doctor via [WIFI / GPS / 4G / etc].

The heart rate and temperature are monitored and their condition assessed according to their severity. The tables below show the assessment conditions for heart rate and temperature.

|  |  |
| --- | --- |
| **Heart Rate (bpm)** | **Assessment** |
| < 40 | Serious |
| 40 to 59 | Fair |
| 60 to 90 | Good |
| 91 to 120 | Fair |
| > 120 | Serious |

|  |  |
| --- | --- |
| **Temperature (°C)** | **Assessment** |
| < 34 | Serious |
| 34 to 35 | Fair |
| 36 to 38 | Good |
| 39 to 40 | Fair |
| > 40 | Serious |

Both the heart rate and temperature assessments are used to determine what alert state to recommend. The table below shows the criteria for the alert recommendation. The alert recommendation is to contact the doctor whenever the heart rate or temperature becomes serious; the alert recommendation is to alert the user whenever both the heart rate and temperature are fair; and a no alert recommendation is given for the rest of the cases.

|  |  |  |
| --- | --- | --- |
| **Heart Rate** | **Temperature** | **Alert Recommendation** |
| Good | Good | No alert |
| Good | Fair | No alert |
| Good | Serious | Alert doctor |
| Fair | Good | No Alert |
| Fair | Fair | Alert user |
| Fair | Serious | Alert doctor |
| Serious | Good | Alert doctor |
| Serious | Fair | Alert doctor |
| Serious | Serious | Alert doctor |

These are only recommendations though. A certain number of consecutive recommendations must occur before the output state of the device transitions from one state to the other, as shown in the table below.

|  |  |
| --- | --- |
| **Alert Recommendation** | **Number of Consecutive Alert Recommendations Needed before Output State Transitions** |
| No Alert | 3 |
| Alert user | 2 |
| Alert doctor | 1 (transition immediately) |

You contracted out the firmware design and have recently received it. Since this is a medical device you will be testing the firmware before shipping it.

The firmware consists of four functions: *returnHeartRateConcern*, *returnTemperatureConcern*, *updateAlertStatus*, and *initAlertStatus*. *returnHeartRateConcern* takes the heart rate as input and gives the corresponding assessment concern. *returnTemperatureConcern* takes the temperature as input and gives the corresponding assessment concern. *initAlertStatus* initializes the output of the device by setting it to no alert. *updateAlertStatus* is the main body of the program. Calling the function represents a new measurement cycle which may or may not change the alert state of the device.

Testfoo.c contains uncompleted unit tests for *returnHeartRateConcern, returnTemperatureConcern,* and *updateAlertStatus.* All three functions contain exactly one bug. Since *updateAlertStatus* relies on *returnTemperatureConcern* and *returnHeartRateConcern* you should test and fix those functions first before testing *updateAlertStatus*. Be sure to test corner cases for *returnTemperatureConcern* and *returnHeartRateConcern,* and include every combination of state transition when testing *updateAlertStatus.*

Objectives:

1. Test *returnHeartRateConcern* and identify the bug, then fix the bug.
2. Test *returnTemperatureConcern* and identify the bug, then fix the bug.
3. Test *updateAlertStatus* and identify the bug, then fix the bug.

Submit Testfoo.c and foo.c via e-mail to [steven.bellock@oit.edu](mailto:steven.bellock@oit.edu)

Assignment is due by Saturday, March 14th at 6 PM.