VDM Specification of Health Monitoring System

```
class HealthMonitoringSystem
-- Defining custom types for signals and health statuses
types
 public Signal = <INCREASE> | <DECREASE> | <DO_NOTHING>;
 public HealthStatus = <HEALTHY> | <WARNING> | <CRITICAL>;
-- Constant values representing thresholds for heart rate and blood pressure
values
 public MAX_HEART_RATE: nat = 120;
 public MIN_HEART_RATE: nat = 60;
 public MAX_BLOOD_PRESSURE: nat = 140;
 public MIN_BLOOD_PRESSURE: nat = 90;
-- Private instance variables to store heart rate, blood pressure, and health status
instance variables
 private heartRate: [nat] := nil;
 private bloodPressure: [nat] := nil;
 private healthStatus: HealthStatus := <HEALTHY>;
-- Operations for recording heart rate and blood pressure, and for updating health
status
operations
 -- Records the heart rate and returns a signal based on the value
 public recordHeartRate: nat ==> Signal
 recordHeartRate(hr) == (
 heartRate := hr;
 return if hr > MAX_HEART_RATE then <DECREASE>
 elseif hr < MIN_HEART_RATE then <INCREASE>
 else <DO_NOTHING>;
 )
 pre inRange(hr); -- Precondition to check if heart rate is in the valid range
```

```
 -- Records the blood pressure and returns a signal based on the value
 public recordBloodPressure: nat ==> Signal
 recordBloodPressure(bp) == (
 bloodPressure := bp;
 return if bp > MAX_BLOOD_PRESSURE then <DECREASE>
 elseif bp < MIN_BLOOD_PRESSURE then <INCREASE>
 else <DO_NOTHING>;
 )
 pre inRange(bp); -- Precondition to check if blood pressure is in the valid range
 -- Updates the health status based on current heart rate and blood pressure
public updateHealthStatus: () ==> ()
 updateHealthStatus() == (
 healthStatus := evaluateHealthStatus(heartRate, bloodPressure);
 );
 -- Getter methods for heart rate, blood pressure, and health status
 public getHeartRate: () ==> [nat]
 getHeartRate() == return heartRate;
 public getBloodPressure: () ==> [nat]
 getBloodPressure() == return bloodPressure;
 public getHealthStatus: () ==> HealthStatus
 getHealthStatus() == return healthStatus;
-- Functions for evaluating range and health status
functions
 -- Checks if a given value is within the defined heart rate range
 inRange: nat -> bool
 inRange(val) == return val >= MIN_HEART_RATE and val <= MAX_HEART_RATE;
 -- Evaluates the health status based on heart rate and blood pressure readings
 evaluateHealthStatus: [nat] * [nat] -> HealthStatus
 evaluateHealthStatus(hr, bp) == (
```

```
 if exists h in set hr & h > MAX_HEART_RATE or h < MIN_HEART_RATE then return
<WARNING>;
 elseif exists b in set bp & b > MAX_BLOOD_PRESSURE or b <
MIN_BLOOD_PRESSURE then return <CRITICAL>;
 else return <HEALTHY>;
 );
end HealthMonitoringSystem
```