```c
/* ************************************************************** */
/* File name:        pid.h                                        */
/* File description: Header file containing the functions/methods */
/*                   interfaces for handling the PID              */
/* Author name:      julioalvesMS, lagoAF, rBacurau               */
/* Creation date:    21jun2018                                    */
/* Revision date:    27mai2020                                    */
/* ************************************************************** */

#ifndef SOURCES_CONTROLLER_PID_H_
#define SOURCES_CONTROLLER_PID_H_


typedef struct pid_data_type {
 float fKp, fKi, fKd;        // PID gains
 float fError_previous;      // used in the derivative
 float fError_sum;           // integrator cumulative error
} pid_data_type;


/* ********************************************** */
/* Method name:        pid_init                   */
/* Method description: Initialize the PID controller*/
/* Input params:       n/a                        */
/* Output params:      n/a                        */
/* ********************************************** */
void pid_init(void);


/* ********************************************** */
/* Method name:        pid_setKp                  */
/* Method description: Set a new value for the PID    */
/*                     proportional constant      */
/* Input params:       fKp: New value             */
/* Output params:      n/a                        */
/* ********************************************** */
void pid_setKp(float fKp);


/* ********************************************** */
/* Method name:        pid_getKp                  */
/* Method description: Get the value from the PID     */
/*                     proportional constant      */
/* Input params:       n/a                        */
/* Output params:      float: Value               */
float pid_getKp(void);


/* ********************************************** */
/* Method name:        pid_setKi                  */
/* Method description: Set a new value for the PID    */
/*                     integrative constant       */
/* Input params:       fKi: New value             */
/* Output params:      n/a                        */
/* ********************************************** */
void pid_setKi(float fKi);


/* ********************************************** */
/* Method name:        pid_getKi                  */
/* Method description: Get the value from the PID     */
/*                     integrative constant       */
/* Input params:       n/a                        */
/* Output params:      float: Value               */
/* ********************************************** */
float pid_getKi(void);
```

```
/* ********************************************** */
/* Method name:        pid_setKd                  */
/* Method description: Set a new value for the PID    */
/*                     derivative constant            */
/* Input params:       fKd: New value                 */
/* Output params:      n/a                        */
/* ********************************************** */
void pid_setKd(float fKd);


/* ********************************************** */
/* Method name:        pid_getKd                  */
/* Method description: Get the value from the PID     */
/*                     derivative constant            */
/* Input params:       n/a                        */
/* Output params:      float: Value                   */
/* ********************************************** */
float pid_getKd(void);


/* ********************************************** */
/* Method name:        pid_updateData                 */
/* Method description: Update the control output      */
/*                     using the reference and sensor */
/*                     value                          */
/* Input params:       fSensorValue: Value read from  */
/*                     the sensor                     */
/*                     fReferenceValue: Value used as */
/*                     control reference              */
/*        fDutyCycleHeater: Value of the */
/*        heater duty cycle     */
/* Output params:      float: New Control effort      */
/* ********************************************** */
float pidUpdateData(unsigned char ucTempAtual, float fSetValue, float fDutyCycleHeater);


#endif /* SOURCES_CONTROLLER_PID_H_ */
```