M1	M2	M3	M4	Final Mark /15	Comments
Sabouri	Kalkhoran	Koolani		14	- shared variables not managed properly (cb_pop_front)
Bentefrit	Montanaro	Bohler	Boutermeur	14*	- circular buffer not dimensioned according to the baud rate
Benkredda	Triki			14*	- cb not correctly dimensioned
Petruzzello	Miceli	Cattin		13,5	- data_r.counter overflows and is not really needed - it is not needed to disable all interrupts - awkward use of STATE_S5/S6 and state++ - wrong deboucing due to INT0IE=1 and INT0IF=0 swap
Demaria	Galvagni			13,5	- wrong debouncing because INT0IF is not cleared before re-enabling INT0IE - shared variables not managed properly all the time (cb.bufferLength in pull())
Delucchi	Cappellini			13	- only 1 byte read every period (10 ms) -> CB overflow and loss of data
Berettieri	Corrao	Tabita		13	- no management of shared variables (chars2read) - no management of cb overflow
Bucciero	Piacenti			13	- calling main_function inside the interrupt generates problems with shared variables and peripherals - shared variable cb.bufferLength is also not handled properly in pull() - long ISR (main_function called inside the U2RX)
Nicchiarelli	Gavagna	Bolla		11,5	- long ISR routines (SPI, busy waiting on the UART) - shared variables not handled properly (checkAvailableBytes, SPI)
Tomaiuolo	Lima			11	- only 1 byte read every period (10 ms) -> CB overflow and loss of data - while(U2STAbits.UTXBF); in the main is wrong - no management of shared variables (isEmpty, dequeue)

Taromsari	Fatemi	Hosseini Rad	**10**	- while(SPI1STATbits.SPITBF == 1) misplaced (after writing to the SPI, sometimes not used at all) - uartWrite() does not check if there is available space in the FIFO - long ISR due to the uartRead() function implementing all the main logic - it is not necessary to disable and reenable the interrupt inside the interrupt itself, just clearing the flag is ok. - no shared variables handling, the buffer is managed by the SPI write without separation from the uartRead()
Aliabadian			**9**	- cb not correctly dimensioned - only 1 byte read every period (10 ms) -> CB overflow and loss of data - no management of full cb - not sending on uart when the button is pressed
Kosum	Toslak	Takele	**8**	- long ISR routines (waiting on the timer, using SPI) - debouncing not correct (waiting inside the ISR and not clearing the IF before exiting) - duplicated tmr_wait_ms - only 1 byte read every period (10 ms) -> overflow - shared variables not managed properly (SPI used both in main and ISR)
Ozgoli	Peyvandi Pour	Dalvand	**8**	- Timer 3 is activated but never stopped, leading to unnecessary interrupts - debouncing attempted but not correctly implemented - while (SPI1STATbits.SPITBF == 1); misplaced - all the logic of the data handling has been implemented in the ISR of the uart, no use of circular buffer
			*	please contact the teacher at the end of the class