Question 1

```
#define F_CPU 1000000U
#include <avr/io.h>
#include <util/delay.h>
#include <stdlib.h>
#include <avr/interrupt.h>

int interr = 0;
int i=0;

/* timer */
ISR(TCA0_OVF_vect)
{
	int intflags = TCA0.SPLIT.INTFLAGS;
	TCA0.SPLIT.INTFLAGS = intflags;
	PORTD.OUTTGL |= PIN0_bm | PIN1_bm;	//breakpoint
}

ISR(TCA0_CMP0_vect){
	int intflags = TCA0.SPLIT.INTFLAGS;
	TCA0.SPLIT.INTFLAGS = intflags;
}


void TCA0_init(void)
{
	TCA0.SPLIT.CTRLA = TCA_SPLIT_CLKSEL_DIV1024_gc;
	TCA0.SPLIT.HPER = 254;
	TCA0.SPLIT.LPER = 254;
```

```
	TCA0.SPLIT.HCMP0 = 127;
	TCA0.SPLIT.LCMP0 = 127;
	TCA0.SPLIT.CTRLB |= TCA_SINGLE_WGMODE_SINGLESLOPE_gc;
	TCA0.SPLIT.CTRLD |= TCA_SPLIT_SPLITM_bm;
	TCA0.SPLIT.INTCTRL = TCA_SPLIT_LUNF_bm;
	TCA0.SPLIT.INTCTRL |= TCA_SPLIT_HUNF_bm;
	TCA0.SPLIT.INTCTRL |= TCA_SPLIT_LCMP0_bm | TCA_SPLIT_LCMP0_bm;
	TCA0.SPLIT.CTRLA |= TCA_SPLIT_ENABLE_bm;
	sei();    //breakpoint
	while(1){
	}
}

int main(void)
{
	PORTD.DIR |= PIN0_bm;        //right
	PORTD.DIR |= PIN1_bm;        //left

	while(1){

		TCA0_init();
	}

	cli();
}
```

Question 2

```
#define F_CPU 1000000U
#include <avr/io.h>
```

```
#include <util/delay.h>
#include <stdlib.h>
#include <avr/interrupt.h>

int interr = 0;
int i=0;

/* timer */
ISR(TCA0_OVF_vect)
{
	int intflags = TCA0.SPLIT.INTFLAGS;
	TCA0.SPLIT.INTFLAGS = intflags;
	PORTD.OUTTGL |= PIN0_bm | PIN1_bm;	//breakpoint
	}
}

ISR(TCA0_CMP0_vect){
	int intflags = TCA0.SPLIT.INTFLAGS;
	TCA0.SPLIT.INTFLAGS = intflags;
}

ISR(ADC0_WCOMP_vect)
{
	int intflags = ADC0.INTFLAGS;
	ADC0.INTFLAGS = intflags;
	PORTD.OUTCLR |= PIN2_bm;  //we change res>10 so that it will get out of
ADC
}

void TCA0_init(void)
```

```
{
	TCA0.SPLIT.CTRLA = TCA_SPLIT_CLKSEL_DIV1024_gc;
	TCA0.SPLIT.HPER = 254;
	TCA0.SPLIT.LPER = 254;
	TCA0.SPLIT.HCMP0 = 127;
	TCA0.SPLIT.LCMP0 = 127;
	TCA0.SPLIT.CTRLB |= TCA_SINGLE_WGMODE_SINGLESLOPE_gc;
	TCA0.SPLIT.CTRLD |= TCA_SPLIT_SPLITM_bm;
	TCA0.SPLIT.INTCTRL = TCA_SPLIT_LUNF_bm;
	TCA0.SPLIT.INTCTRL |= TCA_SPLIT_HUNF_bm;
	TCA0.SPLIT.INTCTRL |= TCA_SPLIT_LCMP0_bm | TCA_SPLIT_LCMP0_bm;
	TCA0.SPLIT.CTRLA |= TCA_SPLIT_ENABLE_bm;
	sei();    //breakpoint
	while(1){
	}
}

void ADC_init(void)
{
	ADC0.CTRLA |= ADC_RESSEL_10BIT_gc;
	ADC0.CTRLA |=ADC_FREERUN_bm;
	ADC0.CTRLA |= ADC_ENABLE_bm;
	ADC0.MUXPOS |= ADC_MUXPOS_AIN7_gc;
	ADC0.DBGCTRL |= ADC_DBGRUN_bm;
	ADC0.WINLT |= 10;
	ADC0.INTCTRL |= ADC_WCMP_bm;
	ADC0.CTRLE |= ADC_WINCM0_bm;
	ADC0.COMMAND |= ADC_STCONV_bm;    //breakpoint
}
```

```
int main(void)
{
	PORTD.DIR |= PIN0_bm;		//right
	PORTD.DIR |= PIN2_bm;		//left
	PORTD.DIR |= PIN1_bm;		//LED2

	ADC_init();

	while(1){

		TCA0_init();
	}

	cli();
}
```

Questions 3-4

```
#define F_CPU 1000000U
#include <avr/io.h>
#include <util/delay.h>
#include <stdlib.h>
#include <avr/interrupt.h>

int interr = 0;
int i=0;

/* timer */
ISR(TCA0_OVF_vect)
```

```
{
	while(ADC0.RES>10){
		int intflags = TCA0.SPLIT.INTFLAGS;
		TCA0.SPLIT.INTFLAGS = intflags;
		PORTD.OUTTGL |= PIN0_bm | PIN1_bm;	//breakpoint, here I press INTFLAGS 5 BIT + SW5 / SW6 for right / left respectively
	}

}

ISR(TCA0_CMP0_vect){
	int intflags = TCA0.SPLIT.INTFLAGS;
	TCA0.SPLIT.INTFLAGS = intflags;
}

/* ADC */
ISR(ADC0_WCOMP_vect)
{
	int intflags = ADC0.INTFLAGS;
	ADC0.INTFLAGS = intflags;
	PORTD.OUTCLR |= PIN2_bm;
	while(ADC0.RES<10){ //breakpoint, here I change intflags, accordingly to where I want it to go
		if(PORTF.INTFLAGS == 32){		//right
			while(interr == 0){
				PORTD.OUTTGL = PIN0_bm;
				PORTD.OUTTGL = PIN0_bm;
				PORTD.OUTTGL = PIN0_bm;
				PORTD.OUTTGL = PIN1_bm;
```

```
                                if(PORTF.INTFLAGS == 33){     //+bit0 of intflags to stop
turning
                                        interr = 1;
                                }
                        }
                }
                if(PORTF.INTFLAGS == 64){     //left
                        while(interr == 0){
                                PORTD.OUTTGL = PIN1_bm;
                                PORTD.OUTTGL = PIN1_bm;
                                PORTD.OUTTGL = PIN1_bm;
                                PORTD.OUTTGL = PIN0_bm;
                                if(PORTF.INTFLAGS == 65){     //+bit0 of intflags to stop
turning
                                        interr = 1;
                                }
                        }
                }
        }
        interr = 0;
}

void TCA0_init(void)
{
        TCA0.SPLIT.CTRLA = TCA_SPLIT_CLKSEL_DIV1024_gc;
        TCA0.SPLIT.HPER = 254;
        TCA0.SPLIT.LPER = 254;
        TCA0.SPLIT.HCMP0 = 127;
        TCA0.SPLIT.LCMP0 = 127;
        TCA0.SPLIT.CTRLB |= TCA_SINGLE_WGMODE_SINGLESLOPE_gc;
```

```
	TCA0.SPLIT.CTRLD |= TCA_SPLIT_SPLITM_bm;
	TCA0.SPLIT.INTCTRL = TCA_SPLIT_LUNF_bm;
	TCA0.SPLIT.INTCTRL |= TCA_SPLIT_HUNF_bm;
	TCA0.SPLIT.INTCTRL |= TCA_SPLIT_LCMP0_bm | TCA_SPLIT_LCMP0_bm;
	TCA0.SPLIT.CTRLA |= TCA_SPLIT_ENABLE_bm;
	sei();    //breakpoint
	while(1){
	}
}

void ADC_init(void)
{
	ADC0.CTRLA |= ADC_RESSEL_10BIT_gc;
	ADC0.CTRLA |=ADC_FREERUN_bm;
	ADC0.CTRLA |= ADC_ENABLE_bm;
	ADC0.MUXPOS |= ADC_MUXPOS_AIN7_gc;
	ADC0.DBGCTRL |= ADC_DBGRUN_bm;
	ADC0.WINLT |= 10;
	ADC0.INTCTRL |= ADC_WCMP_bm;
	ADC0.CTRLE |= ADC_WINCM0_bm;
	sei();
	ADC0.COMMAND |= ADC_STCONV_bm;    //breakpoint



}

int main(void)
{
	PORTD.DIR |= PIN0_bm;        //right
```

```
        PORTD.DIR |= PIN1_bm;       //left
        PORTD.DIR |= PIN2_bm;       //LED2
        PORTF.DIR |= PIN5_bm;       //right SW5
        PORTF.DIR |= PIN6_bm;       //left SW6, breakpoint

        ADC_init();
        while(1){
                TCA0_init();  //breakpoint
        }

        cli();
}
```