Αρχικά πρέπει να εισάγουμε τον κωδικό. Θα οριστούν interrupts στο portf και κάθε φορά που θα πατιέται το sw5 και sw6 θα γίνεται διακοπή. Στην ISR θα ελέγχεται ποιος διακόπτης πατήθηκε και μέσω μεταβλητών θα υλοποιξθεί μια μνήμη ώστε κάθε φορά που μπαίνουν 4 ψηφία να ελέγχει αν είναι τα σωστά. Εννοείται ότι κάθε φορά που βγαίνει από την ISR το πρόγραμμά μας θα ξαναμηδενίζει τα switches. Αν πατηθούν και οι δύο διακόπτες ταυτόχρονα και έχουν τιμή 1, τότε θα ελέγχεται και αποθηκεύεται η τιμή μόνο του πρώτου. Μόλις γίνει σωστά το πρώτο βήμα θα ενεργοποιηθεί ένας timer TCA για να προλάβουν να φύγουν οι πελάτες. Εκεί θα βάλουμε μια while συνάρτηση να περιμένει τον timer και μόλις γίνει διακοπή θα συνεχίσει η ροή του προγράμματος κανονικά. Μετά θα αρχικοποιηθεί και θα τεθεί σε λειτουργία ο ADC. Μόλις Εντοπίσει κάτι, ένας tca ενεργοποιείται.

Ερώτημα 1

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#define ped=1000;
#define TCB_CMP_EXAMPLE_VALUE (0x80ff)
int x=-1;
int d=0;
int array[3]=0;
void First_Function(void);
void Second_Function(void);

int main() {
	PORTD.DIR |= PIN0_bm;
	PORTD.OUT=PIN0_bm; // led is off
	void First_Function();




};
void TCA_init (void);  {
	TCA0.SINGLE.CNT=0; //clear counter
	TCA0.SINGLE.CTRLB=0; // normal mode
	TCA0.SINGLE.CMP0=ped;// when reaches that value interrupt
	TCA0.SINGLE.CTRLA = TCA_SINGLE_CLKSEL_CIV1024_gc;
	TCA0.SINGLE.CTRLA |=1; // enable
	TCA0.SINGLE.INTCTRL = TCA_SINGLE_CMP0bm; // interrupt enable
	sei(); // accept interrupts
	while (interr==0) {}
	cli();
}

ISR(TCA_CMP0_vect){ //isr for initial tca

	TCA.SINGLE.CTRLA=0; // disable clear fl
	int intflags=TCA0.SINGLE.INTFLAGS;
	TCA0.SINGLE.INTFLAGS=inflags;
interr=1;}


void First_Function(void){
	label;
	while(x!=3){ //4 digits

		PORTF.DIR|=PIN5_bm; //PIN 5 IS OUTPUT
		PORTF.OUTCLR|=PIN5_bm; //PIN5 CLEAR
		PORTF.PIN5CTRL|=PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc; //pullup enabled and interrupt enabled with sense on both edges
		sei(); //enable interrupts
		while (interr==0) {};  };
		cli();//disenable interrupts

		int arraycode[3]=(6,5,5,6); //right code
		for (i=0; i<4; i++)
		if (arraycode[i]!=array[i]) { //is right code ?
			array[i]=(0,0,0,0);
			x=-1; //restart
		goto label;   }

		//there the code is right

		void TCA_init (void);
	}


	ISR(PORTF_PORT_vect) //first function isr
	{
		int intflags=PORTF.INTFLAGS;
		PORTF.INTFLAGS=inflags;
		interr=1;
		x=x+1;
		if (PORTF.OUT == 01000000) //SW6
		{    array[x]=6;}

		else  //SW5
		{    array[x]=5;}

	}


	ISR(TCB0_INT_vect)
	{
		TCB0.INTFLAGS=TCB_CAPT_bm; /*CLEAR THE INTERRUPT FLAG*/
		PORTB.IN=PIN0_bm; /*TOGGLE PB5 GPIO*/
		PORTD.OUTCLR |= PIN0_bm; //open led0
		if(TCB1.INTFLAGS==1)
		{
			TCB1.INTFLAGS=TCB_CAPT_bm; /*CLEAR THE INTERRUPT FLAG*/
			PORTB.IN=PIN1_bm; /*TOGGLE PB5 GPIO*/
			PORTD.OUTCLR |= PIN1_bm; //open led1
		};
	}
```

Ερώτημα 2

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#define ped=1000;
#define TCB_CMP_EXAMPLE_VALUE (0x80ff)
int x=-1;
int d,f=0;
int array[3]=0;
void First_Function(void);
void Second_Function(void);
void ADC_init (void);


int main() {
	PORTD.DIR |= PIN0_bm;
	PORTD.OUT=PIN0_bm; // led is off
```

```
};
void TCA_init (int y);  {
	if(y==1) d=1;
	else d=0;
	TCA0.SINGLE.CNT=0; //clear counter
	TCA0.SINGLE.CTRLB=0; // normal mode
	TCA0.SINGLE.CMP0=ped;// when reaches that value interrupt
	TCA0.SINGLE.CTRLA = TCA_SINGLE_CLKSEL_CIV1024_gc;
	TCA0.SINGLE.CTRLA |=1; // enable
	TCA0.SINGLE.INTCTRL = TCA_SINGLE_CMP0bm; // interrupt enable
	sei(); // accept interrupts
	while (interr==0) {}
	cli();
}

ISR(TCA_CMP0_vect){ //isr for initial tca
	TCA.SINGLE.CTRLA=0; // disable clear fl
	int intflags=TCA0.SINGLE.INTFLAGS;
	TCA0.SINGLE.INTFLAGS=inflags;
	interr=1;
if (d==1) goto label1; }


void First_Function(int y){
	if (y==0) f=1;
	else f=0; //elenxos gia 1h h 2h synartisi
	label;
	while(x!=3){ //4 digits

		PORTF.DIR|=PIN5_bm; //PIN 5 IS OUTPUT
		PORTF.OUTCLR|=PIN5_bm; //PIN5 CLEAR
		PORTF.PIN5CTRL|=PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc; //pullup enabled and interrupt enabled with sense on both edges
		sei(); //enable interrupts
		while (interr==0) {};  };
		cli();//disenable interrupts

		int arraycode[3]=(6,5,5,6); //right code
		for (i=0; i<4; i++)
		if (arraycode[i]!=array[i]) { //is right code ?
			array[i]=(0,0,0,0);
			x=-1; //restart
		goto label;   }

		//there the code is right

		void TCA_init (int y);
}


ISR(PORTF_PORT_vect) //first function isr
{
	int intflags=PORTF.INTFLAGS;
	PORTF.INTFLAGS=inflags;
	interr=1;
	x=x+1;
	if (PORTF.OUT == 01000000) //SW6
	{    array[x]=6;}

	else  //SW5
	{    array[x]=5;}

}



void ADC_init (void){
	PORT.DIR|= PIN0_bm; //PIN IS OUTPUT
	//INITIALLIZE THA ADC FOR FREE RUNNING MODE
	ADC0.CTRLA |= ADC_RESSEL_10BIT_gc;//10BIT RESOLUTION
	ADC0.CTRLA |= ADC_FREERUN_bm; //FREE RUNNING MODE ENABLED
	ADC0.CTRLA |= ADC_ENABLE_bm; //ENABLE ADC
	ADC0.MUXPOS |= ADC_MUXPOS_AIN7_gc; // THE BIT ENABLE DEBUG MODE
	ADC0.DBGCTRL |= ADC_DBGRUN_bm; // WINDOW COMPARATOR MODE
	ADC0.WINLT |= 10; // SET THRESHOLD
	ADC0.INTCTRL |= ADC_WINCM0_bm; // ENABLE INTERRUPTS FOR WCM
	ADC0.CTRLE |= ADC_WINCM0_bm; // INTERRUPT WHEN RESULT<WINLT
	sei();
	ADC0.COMMAND |= ADC_STCONV_bm; // START CONVERSION
	while(1){}
}

ISR(ADC0_WCMP_vect) { // adc
	int intflags=ADC0.INTFLAGS;
	ADC0.INTFLAGS = intflags;
	PORTD.OUTCLR=PIN0_bm; // led is on
	void Second_Function()
}



void Second_Function(void){
	void TCA_init()
	while(d!=4){
		label;
		while(x!=3){ //4 digits

			PORTF.DIR|=PIN5_bm; //PIN 5 IS OUTPUT
			PORTF.OUTCLR|=PIN5_bm; //PIN5 CLEAR
			PORTF.PIN5CTRL|=PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc; //pullup enabled and interrupt enabled with sense on both edges
			sei(); //enable interrupts
			while (interr==0) {};  };
			cli();//disenable interrupts

			int arraycode[3]=(6,5,5,6); //right code
			for (i=0; i<4; i++)
			if (arraycode[i]!=array[i]) { //is right code ?
				array[i]=(0,0,0,0);
				x=-1; //restart
			goto label;   }
		}
	}
	label1;

	void open_sireen();
	d=0; goto label;




}
```

```
Ερώτημα 3
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#define ped=1000;
#define TCB_CMP_EXAMPLE_VALUE (0x80ff)
int x=-1;
int d,f=0;
int array[3]=0;
void First_Function(void);
void Second_Function(void);
void ADC_init (void);
void TCB0_init (void);


int main() {
	Label3
	PORTD.DIR |= PIN0_bm;
	PORTD.OUT=PIN0_bm; // led is off
	void First_Function(void);
	void Second_Function(void);




};
void TCA_init (int y);  {
	if(y==1) d=1;
	else d=0;
	TCA0.SINGLE.CNT=0; //clear counter
	TCA0.SINGLE.CTRLB=0; // normal mode
	TCA0.SINGLE.CMP0=ped;// when reaches that value interrupt
	TCA0.SINGLE.CTRLA = TCA_SINGLE_CLKSEL_CIV1024_gc;
	TCA0.SINGLE.CTRLA |=1; // enable
	TCA0.SINGLE.INTCTRL = TCA_SINGLE_CMP0bm; // interrupt enable
	sei(); // accept interrupts
	while (interr==0) {}
	cli();
}

ISR(TCA_CMP0_vect){ //isr for initial tca
	TCA.SINGLE.CTRLA=0; // disable clear fl
	int intflags=TCA0.SINGLE.INTFLAGS;
	TCA0.SINGLE.INTFLAGS=inflags;
	interr=1;
if (d==1) goto label1; }


void First_Function(int y){
	if (y==0) f=1;
	else f=0; //elenxos gia 1h h 2h synartisi
	label;
	while(x!=3){ //4 digits

		PORTF.DIR|=PIN5_bm; //PIN 5 IS OUTPUT
		PORTF.OUTCLR|=PIN5_bm; //PIN5 CLEAR
		PORTF.PIN5CTRL|=PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc; //pullup enabled and interrupt enabled with sense on both edges
		sei(); //enable interrupts
		while (interr==0) {};  };
		cli();//disenable interrupts

		int arraycode[3]=(6,5,5,6); //right code
		for (i=0; i<4; i++)
		if (arraycode[i]!=array[i]) { //is right code ?
			array[i]=(0,0,0,0);
			x=-1; //restart
		goto label;   }

		//there the code is right

		void TCA_init (int y);
	}


	ISR(PORTF_PORT_vect) //first function isr
	{
		int intflags=PORTF.INTFLAGS;
		PORTF.INTFLAGS=inflags;
		interr=1;
		x=x+1;
		if (PORTF.OUT == 01000000) //SW6
		{    array[x]=6;}

		else  //SW5
		{    array[x]=5;}

}




void ADC_init (void){
	PORT.DIR|= PIN0_bm; //PIN IS OUTPUT
	//INITIALLIZE THA ADC FOR FREE RUNNING MODE
	ADC0.CTRLA |= ADC_RESSEL_10BIT_gc; //10BIT RESOLUTION
	ADC0.CTRLA |= ADC_FREERUN_bm; //FREE RUNNING MODE ENABLED
	ADC0.CTRLA |= ADC_ENABLE_bm; //ENABLE ADC
	ADC0.MUXPOS |= ADC_MUXPOS_AIN7_gc; // THE BIT ENABLE DEBUG MODE
	ADC0.DBGCTRL |= ADC_DBGRUN_bm; // WINDOW COMPARATOR MODE
	ADC0.WINLT |= 10; // SET THRESHOLD
	ADC0.INTCTRL |= ADC_WINCM0_bm; // ENABLE INTERRUPTS FOR WCM
	ADC0.CTRLE |= ADC_WINCM0_bm; // INTERRUPT WHEN RESULT<WINLT
	sei();
	ADC0.COMMAND |= ADC_STCONV_bm; // START CONVERSION
	while(1){}


}

ISR(ADC0_WCMP_vect) { // adc
	int intflags=ADC0.INTFLAGS;
	ADC0.INTFLAGS = intflags;
	PORTD.OUTCLR=PIN0_bm; // led is on
	void Second_Function()
}



void Second_Function(void){
	void TCA_init()
```

```
while(d!=4){
	label;
	while(x!=3){ //4 digits

		PORTF.DIR|=PIN5_bm; //PIN 5 IS OUTPUT
		PORTF.OUTCLR|=PIN5_bm; //PIN5 CLEAR
		PORTF.PIN5CTRL|=PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc; //pullup enabled and interrupt enabled with sense on both edges
		sei(); //enable interrupts
		while (interr==0) {};  };
		cli();//disenable interrupts

		int arraycode[3]=(6,5,5,6); //right code
		for (i=0; i<4; i++)
		if (arraycode[i]!=array[i]) { //is right code ?
			array[i]=(0,0,0,0);
			x=-1; //restart
			goto label;
		else TCB0.CTRLA |= TCB_DISABLE_bm; goto label3   }  //go to main for func 1
	}
}
label1;

void TCB0_init (void);
d=0; goto label;
```

```
void TCB0_init (void){
	/*load ccmp register with the period and duty cycle of the PWM*/
	TCB0.CCMP=TCB_CMP_EXAMPLE_VALUE;

	/*enable tcb3 and divide clk_per by 2*/
	TCB0.CTRLA |= TCB_ENABLE_bm;
	TCB0.CTRLA |= TCB_CLKSEL_CLKDIV2_gc;

	/*ENABLE PIN OUTPUT and configure configure TCB in 8-bit PWM mode*/
	TCB0.CTRLB |=TCB_CCMPEN_bm;
	TCB0.CTRLB |= TCB_CNTMODE_PWM8_gc;}

}



ISR(TCB0_INT_vect)
{
	TCB0.INTFLAGS=TCB_CAPT_bm; /*CLEAR THE INTERRUPT FLAG*/
	PORTB.IN=PIN0_bm; /*TOGGLE PB5 GPIO*/
	PORTD.OUTCLR |= PIN0_bm; //open led0

}
```