Hauptprogramm mit Scheduler (Anwenderprogramm läuft alle 100ms)

```
#include <avr/io.h>
#include <avr/interrupt.h>

volatile unsigned char trigger_10hz, trigger_2000hz;    //Variablen auch für ISR gültig

scheduler tasking;                    //scheduler objekt für programm-timing erzeugen

int main(void)
{
   sei();                             //interrupts ein
   while(1)                           //endlosschleife
   {
      if (trigger_10hz == 1)          //timergesteuerte ausführung alle 100ms
      {
         trigger_10hz = 0;            //..und hier gleich wieder null


      //*************************************************************

         //HIER KOMMT IHR PROGRAMM REIN

      //*************************************************************

      }//ende der 10hz-task in der "while 1" -main-schleife
   };//ende "while 1" - schleife innerhalb main

   return 0;

}//ende von main


//---------------------------------------------------------------------------
//Interruptroputinen

//Generator für scheduler
ISR(TIMER1_COMPA_vect)
{
   trigger_2000hz ++;                 //trigger für scheduler im 0.5ms-raster
   if (trigger_2000hz > 200)          //durch 200 - > 100 ms für den main-secheduler
      {
         trigger_10hz = 1;            //scheduling-flag fürs hauptprogramm (wird dort null)
         trigger_2000hz = 1;          //wieder von vorne mit dem zählen anfangen
      }
}
```

Konstruktor (und Methoden) der Klasse “scheduler“ : scheduler.cpp

```
#include "scheduler.h"
#include <avr/io.h>

scheduler::scheduler()                          //hier außer dem Konstruktor keine Methoden
{
   TCCR1A = 0;
   TCCR1C = 0;
   OCR1AH = 0;
   OCR1AL = 125;
   TCCR1B |= ((1<<WGM12) | ( 1<<CS11) | (1<<CS10));
   TIMSK1 |= (1<<OCIE1A);
}
```

Definitionsdatei der Klasse “scheduler“ :

```
class scheduler
{

   public:
      scheduler();
};
```