Tid – Obligatorisk Opgave

**Deadline d. 24/6-20**

# One-2-Watch

## Opgave Beskrivelse

Denne opgave går ud på at lave et multifunktionelt ur, som kan hjælpe med flere forskellige ting i din hverdag. Tænk opgaven som, at hvis DU skulle lave et ur til dig selv, der kan en masse forskellige ting; Hvordan skal det så se ud for at give et godt overblik.? Giver interfacet mening i forhold til hvad det skal kunne.? Er det let at interagere med dit ur/program og er det nemt at bruge.? Hvordan skal det virke, for at DU selv vil bruge det.?

### Opg. 1a – Tid er alt!

Først skal I have lavet et ur og gør det synligt, så vi har et udgangspunkt for programmet.

* Lav et interface i WFP, som er overskueligt og kan indeholde et digitalt ur.
* Lav et digitalt ur, som kan vise tiden på dagen.
  + F.eks. kl. 08.15 - (uret skal selvfølgelig kunne følge reel tid).

### Opg. 1b – Timing er alt!

I skal lave en udvidelse af jeres ur, så det også kan fungere som et stopur og bruges til nedtælling.

* Lav en udvidelse til jeres ur, så I har mulighed for at tage tid med uret.
  + Tænk over hvad et ”standard” stopur kan og dets funktionalitet (Start, Stop, Pause/fortsæt, nulstil).
* Lav en udvidelse af uret, så det også får en nedtællings-funktion.
  + Uret skal kunne tælle ned fra timer, minutter og sekunder.
  + Gør det muligt at kunne redigere i nedtællingen løbende! Så hvis der er sat en nedtælling i gang, så kan den f.eks. lægge 2min og 30 sekunder til, uden at der skal laves en ny nedtælling.
    - Læg tid til
    - Træk tid fra

**OBS!** - Husk en repræsentation i jeres program af alle de tilgængelige muligheder.

## Krav for hele projektet

* Skal lave projektet i C#.
* Skal arbejde i WFP.
* Fokus på OOP
* Skal bruge Github, med daglige Commits/Pushes.
* Skal lave minimum ét interface og gøre brug af det.
* Skal lave minimum én abstrakt klasse og gøre brug af det.

## Krav - OBS!

Vær opmærksom på brug af.:

* **Navngivning** (Lignende navngivning er ikke godt, f.eks.; string var4, void button2\_click() eller form2.cs. Disse er fremover et no-go, I er bedre end dette!)
  + Giv jeres variabler, funktioner og klasser mm., navne som giver mening for jeres projekt.
  + Brug camelCasing til jeres variabler – somDetteHer
  + Brug PascalCasing til Projektnavne/Namespaces, funktioner og klasser – SomDetteHer
  + Brug stort ’I’ foran jeres Interfaces – IsomDetteHer
* **OOP**
  + <https://www.freecodecamp.org/news/object-oriented-programming-concepts-21bb035f7260/>