Nemendur hefja þetta verkefni í 4-6 manna hópum.

Tímamælingar
Hver meðlimur hópsins velur sér eitt af eftirfarandi verkefnum. Einhver í hópnum skal velja hvert af þeim (svo engu sé sleppt) og ekki skulu fleiri en tveir velja sama verkefni (svo þeim sé sem jafnast dreift).

1. Finna leiðir til að framkvæma tímamælingar í python
    - Hægt skal vera að hefja tímamælingu
    - Hægt skal vera að keyra annan forritshluta eftir að tímamæling hefst
    - Hægt skal vera að stöðva og/eða fá núverandi tíma á tímamælingu

2. Hanna klasa sem heldur utan um tímamælingar
    - Hér þarf að ákveða hvaða aðgerðir hægt er að gera
    - Ákveða hvað gerist þegar hver aðgerð er keyrð
    - Hvernig styðja þessar aðgerðir það að hefja, enda og fá upplýsingar um tímamælingu?

3. Hanna og skilgreina útreikninga til að setja tíma fram á skýran hátt
    - Ef tími kemur inn sem ein tala, hvað táknar sú tala
    - Hvernig er hægt að setja hana fram á skýrari og læsilegri hátt?
        - Klst, mín, sek, hundraðshlutar, millisekúndur, míkrósek
        - Er þörf á þessu öllu, eða misjafnt eftir tölum/notkun?
    - Setjið fram (handskrifað) þá útreikninga sem þarf að framkvæma til að þetta sé mögulegt.

Nú fá nemendur nokkrar mínútur til að leita á netinu, skrifa niður hugmyndir og útreikninga eða skipuleggja sig á annan hátt til þess að útskýra niðurstöðu sína. Nemendur framkvæma þessa vinnu hver fyrir sig. Ef tvö hafa sama verkefni gera þau þetta engu að síður hvort í sínu lagi.

Þegar nemendur hafa fengið tíma til að skipuleggja sig hver fyrir sig kemur hópurinn aftur saman. Hann fær núna nokkrar mínútur til að ræða hvert af þessum þremur verkefnum. Í hverju tilfelli hafa þeir nemendur sem “áttu” þetta verkefni forsögu, en aðrir ættu endilega að leggja orð í belg, spyrja spurninga og almennt ganga úr skugga um að niðurstaða hópsins sé nægjanlega nákvæm til að hægt sé að byrja að forrita lausn án þess að taka frekari ákvarðanir. Í þessum hluta á hópurinn sem sagt að taka endanlegar ákvarðanir um það hvernig hvert af þessum verkefnum á að vera framkvæmt. Hér gætu einnig bæst inn þarfir sem ekki voru í neinu verkefnanna en koma upp þegar horft er á þau sameinuð, t.d. má velta fyrir sér hvort þörf sé á fleiri úttaksföllum á klasanum (eða nýjum færibreytum) ef gert er ráð fyrir mismunandi framsetningu á úttakinu.

Nú velur hver hópmeðlimur aftur eitt af þessum verkefnum ***en ekki það sama og hann hafði í fyrsta hlutanum***. Nemendurnir forrita næst þessar lausnir. Þeir nemendur sem hafa sama

verkefni mega hjálpast að eða vinna hvert fyrir sig en að lokum verða lausnirnar allar sameinaðar í eina.
Nemendur skulu byrja á því að forrita verkefnin einangruð og vinna bara með þær prófanir sem henta. Klasann þarf t.d. að útbúa með öllum föllum og prófa að hægt sé að kalla á þau, en þau hafa lítið sem ekkert innihald til að byrja með. Eins þarf fyrst bara að prófa tímamælinguna og athuga að allar aðgerðir virki og skili réttri niðurstöðu, en á einhverjum tímapunkti sjá nemendur að þeir verða að byrja að sameina þetta og koma virkninni inn í föllin á klasanum. Einnig má prófa skýru framsetninguna sjálfa til að byrja með og nota slembitölur, en að lokum þarf að koma því inn í klasann og passa að aðlaga allt inntak og úttak að sameinuðu útgáfunni.

Frekari hugmyndir og viðbætur
Fyrsti hluti verkefnisins er mjög opinn og nemendur ættu að reyna að fá sem flestar hugmyndir um útfærslur upp á yfirborðið, hvort sem þær eru notaðar eða ekki.

Hér koma nokkrar spurningar og hugmyndir að viðbætum sem hópurinn getur í sameiningu hannað og bætt við þegar grunnvirknin er komin.

- Er þörf á því bæði að geta stöðvað tímamælingu og beðið um núverandi tíma á henni, eða er hægt að fá allar upplýsingar sem þörf er á með öðru hvoru?
- Er eðlilegt að það sé sitt hvor virknin að “pása” og að “stöðva”.
    - Getur þetta nýst við ákveðnar tímamælingar?
        - Er hægt að setja klasann þannig upp að hann taki við einkenni (tölu eða streng) fyrir hverja tímamælingu og geti þannig haldið utan um fleiri en eina tímamælingu í einu?
    - Hvernig væri þetta útfært (og endilega útfærið)
    - Þyrftu start/stop/pause/get_time/etc föllin öll að taka við einkenni?
    - Hvaða innbyggða python gagnagrind gæti auðveldað þessa virkni?

Nú ætti hópurinn að vera kominn virka lausn á tímamælingaklasa sem hægt er að nota til þess að mæla hve langan tíma einhver forritsaðgerð tekur.

Flottast væri að flytja tímamælingaklasann í sérskrá sem inniheldur ekkert annað. Í skránni þar sem nota á mælingarnar er þá hægt að gera **import** *setningu til að vísa í klasann þar og þá er engum kóða blandað saman, þ.e. tímamælingakóða og kóða aðgerðanna sjálfra.*

Aðgerðir til tímamælinga
Langbest er að búa til föll (eða jafnvel klasa) í kringum þessar aðgerðir, svo að þegar tímamælingin fer fram sé bara kallað á upphafsfall á tímamælingaklasa, síðan á fallið sem á að mæla í einni línu og að lokum á lokafallið í tímamælingunni. ***Venja sig á snyrtilega forritun!***

1. **Stak sótt í lista**
   Búið til lista í python (`some_lis = []`)
   Setjið ákveðinn fjölda staka í listann (10, 100, 1000, 10000, etc.)

Lúppa i frá 0 upp í 9 (eða 99 eða 999) og setja random stak í lis[i].
Sækið ákveðið stak í listann (some_var = lis[index]).
Það má búa til index með random. Það getur líka verið sniðugt að framkvæma aðgerðina oftar, t.d. 1000 sinnum og tímamæla það, til að fá jafnara meðaltal. Ein slík aðgerð tekur svo ótrúlega stuttan tíma að það er óvíst að það sé að marka.
Mælið hve langan tíma það tekur að sækja stak (eða 1000 stök).
Spurning: *Hefur áhrif á þennan tíma hversu mörg stök eru í listanum?*
Hversu vel er hægt að einangra aðgerðina svo að aðrar aðgerðir hafi ekki áhrif á tímamælinguna?

2. **Stak fundið í lista**
   Setjið aftur stök í lista og keyrið núna setninguna:
   ```
   if some_value in lis:
       #Gera eitthvað mjög lítið og einfalt
       pass
   ```
   Prófið þetta með misstórum listum og reynið að fá jafna og góða tímamælingu.
   Spurning: *Hefur áhrif á þennan tíma hversu mörg stök eru í listanum?*

3. **Lykill fundinn í dictionary**
   Búið núna til dictionary og setjið lykla inn í það á sama hátt og þið settuð áður stök inn í lista. Setjið bara einhver slembigildi inn sem gildin.
   Dæmi: `my_dictionary[some_value] = some_random_string`
   Prófið núna að keyra:
   ```
   if some_value in my_dictionary:
       #Gera eitthvað mjög lítið og einfalt
       pass
   ```
   Prófið þetta með misstórum listum og reynið að fá jafna og góða tímamælingu.
   Spurning: *Hefur áhrif á þennan tíma hversu mörg stök eru í listanum?*

Flóknari aðgerðir og innbyggðar aðgerðir

Þið ráðið algerlega í hvaða röð þið gerið þetta og gerið helst það sem þið eruð spenntust fyrir að sjá og prófa.

Prófið að finna ákveðið gildi (ekki ákveðna index staðsetningu) í venjulegum lista, með því að labba allan listann þangað til gildið finnst (eða listinn klárast).
Mælið þessa aðgerð. ***Hefur stærð listans núna áhrif?***

Ræðið ykkar á milli, og við kennara, um áhugaverðar aðgerðir til að útfæra í lista. Það þarf ekkert að vera skynsamlegt, bara prófa að gera ákveðna hluti í lista og tímamæla þessar aðgerðir. Athuga hvort stærð listans hafi áhrif á útkomuna eða hvort það komi á óvart hversu

langan eða stuttan tíma þær taka.

Prófið innbyggðar aðgerðir á python listum

- **append**
- **insert**
- **remove**
- **sort**
- **count**
- **reverse**
- **copy**
- **clear**

Prófið þessar aðgerðir á misstórum listum og reynið að finna reglu á því hvaða áhrif lengd listans hefur (ef einhver). Ef þið tvöfaldið lengd listans, tvöfaldast tími aðgerðar, stendur hann í stað, fjórfaldast hann eða eitthvað annað?

Prófið að leggja saman tvo lista (eða tvo strengi) og sjáið hvaða áhrif lengd listans hefur.

Athugið hjá öðrum hópum hvort þeir hafi komist að svipuðum niðurstöðum og þið varðandi tímamælingar og áhrif á tilteknar aðgerðir. Athugið hvort mælingarnar eru verulega mismunandi á milli tölva meðlima hópsins, jafnvel með nákvæmlega sama forritstexta.

Prófið, **prófið, *prófið, PRÓFIÐ!!!***