```
#Dette program kan beregne den levetid man har tilbage!
name = input("navn: ")  #først beder vi om brugerens navn & alder
age = input("alder: ")

#Vi skal være sikker på at det er et tal!
if age.isdigit():
        ageAsInt = int(age) #tallet skal konverteres fra en tekst til et tal!

    #vi vil være sikre på at det er en rigtig alder!
    if ageAsInt >= 0:
        expectedLife = 80
        #Vi har behov for at kende kønnet!
        gender = input("hvad er dit køn (k/m): ")

        if gender == "m" or gender == "k":
            #Alt efter om man er kvinde eller mand lever man forskelligt!
            if gender == "m":
                expectedLife = 78
            else:
                expectedLife = 82

            print("Hej "+name+" - og velkommen!")
            yearsLeft = expectedLife - ageAsInt

            #Vi beregner antallet af år, dage, timer, minutter & sekunder!
            print("Du har " + str(yearsLeft) + " år at leve i")
            daysLeft = yearsLeft*365
            print("Du har "+str(daysLeft)+" dage at leve i")
            hoursLeft = daysLeft*24
            print("Du har "+str(hoursLeft)+" timer at leve i")
            minLeft = hoursLeft*60
            print("Du har "+str(minLeft)+" minutter at leve i")
            secLeft = minLeft*60
            print("Du har "+str(secLeft)+" sekunder at leve i")
        else:
            print("det forstod jeg ikke!")
    else:
        print("det kan vidst ikke passe!")
else:
    print("du skal skrive en alder")
```

Løkker også kaldet Loops:

I Pyton og i hvilket som helst andet programmeringssprog er loops vigtige og findes i forskellige former! Det vi skal se på nu hedder et `while` loop!

I et while loop kører loopet indtil udsagnet efter while viser sig at blive FALSE! Dvs. så længe udsagnet er sandt looper den!

Det betyder at man har mulighed for at lave loops der kører uendeligt! Lad os se på et!

```
while True:
    print(”hej med dig”)
```

Vi kunne også have brugt False!

Opgave 1: Lav et uendeligt loop! Det kan stoppes ved at trykke CTRL-c. Lav herefter et loop som aldrig starter ved at bruge False!

While loop indtil 10!

Efter `while` kan man sætte en sammenligning ind! Det er oplagt at bruge tal her! Lad os derfor printe alle tal op til 10!

```
a = 0
n = 10

while a < n:
    #Vi udskriver tallet a!
    print(a)
    #vi lægger 1 til tallet a
    a = a+1
```

Opgave 2: Du skal lave et while loop som udskriver alle tal til og med 100!

Syntaksen er meget vigtig. Hvis man f.eks. ændre loopet til

```
while a < n:
    print(a)
a = a+1
```

Vil den skrive 0 i det uendelige fordi så er det kun print(a) der er med i loopet!

Loop & matematik:

Inde i loopet kan vi f.eks. regne - lægge tal sammen!

result = result +a

Dette udtryk vil lægge alle tal sammen fra 0 til 100!

Opgave 3: Lav et program der lægger alle tal sammen fra 0 til 100! NB: det skal give 5050 selvfølgelig!

Opgave 4: Lav et program som lægger alle tal sammen op til det tal brugeren giver som input!

Funktioner:

Indtil videre har vi lavet programmer som er simple og køre fra A til B! Dog oplever man ofte at man bruger den samme kode om og om igen! Da vi jo er dovne ville det være smart at kunne samle denne kode som man så kan kalde om og om igen - uden at skulle skrive den hele tiden! Dette kaldes for en funktion! En funktion skal selvfølgelig have et navn - få overført nogle variable (kaldt argumenter) og kunne returnere en værdi! Lad os se på et eksempel:

```
#Vi definerer en funktion som lægger 1 til et nummer a!
def incremention(a):
    a = a+1
    #Vi returner a tilbage!
    return a

a = 0
n = 101
result = 0

while a < n:
    print(result)

    #Vi kalder funktionen incremention med argumentet a!
    a = incremention(a)
    result = result+a
```

Opgave 5: Lav en function der lægger 1 til og brug den i programmet fra opgav 4!

Regneoperationen Modula %:

I den følgende opgave kommer vi til at bruge en regne operation som kan returnere resten af et divisions stykke!

12 % 2

Dette regnestykke vil returnere resten af divisionsstykket 12 / 2 hvor resten jo er 0!

13 % 2

Dette vil resulterer i en rest på 1!

Opgave 6: Lav et program som kan afgøre om det tal brugen skriver er et lige eller ulige tal!
NB: tal % 2 vil altid give 0 ved et lige tal!

Et primtal!

Et primtal er et tal hvor der kun er 2 divisorer 1 og tallet selv!

Opgave 7: Lav et program der udskriver alle primtal op til 1000. Du kan med fordel bruge følgende to funktioner!

```
def unEqualNumber(a):
    if a % 2 == 1:
        return True
    else:
        return False

def gotDivisors(n):
    i = 2
    while i < n:
        if n % i == 0:
            return True
        else:
            pass
        i = i+1
    return False
```