Projekt støttet af:

LKM0: Forløb/Opgaver

Der vil være:

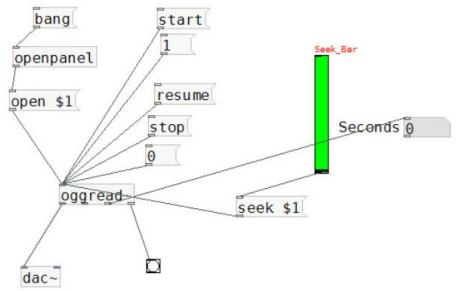
- En introduktion til Micro:Bit og til Python/MicroPython, samt udviklingsmiljøet – Thonny.
- Gennemgang af sensorer, radio og styring af højtaler med hands-on kodning på klassen.
  1) Løbende opgaver til at undersøge funktion selvstændigt.
- Opgave i grupper, hvor der skal bygges 2 produkter – En bunden opgave og én fri der begge afslutningsvis skal fremlægges og forklares som nævnt i indledning.
  1) Lav en aktiv udgave af Corona badgen, som giver en advarsel med lyd og lys når der kommer nogen for tæt på.
  2) Lav et andet produkt efter eget valg. Det kan f.eks. være en leg, et måleinstrument eller et forsøg med nudging. For eksempel om man kan få flere til at gå til højre end til venstre når man går ind ad en dør eller tage trappen - Se: https://www.youtube.com/watch?v=2lXh2n0aPyw&feature=youtu.be

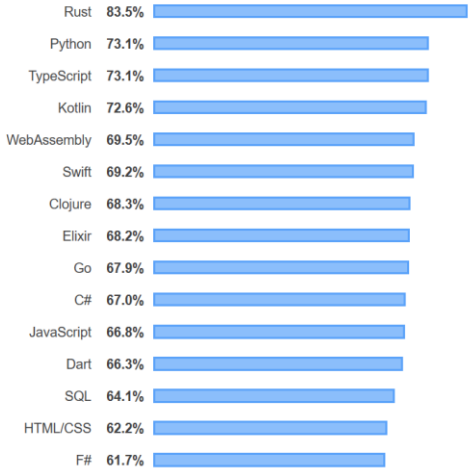
Begge produkter skal testes, og test dokumenteres og bruges i argumentation ved fremlæggelse.

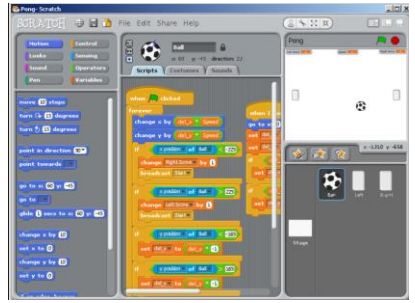
Projekt støttet af:

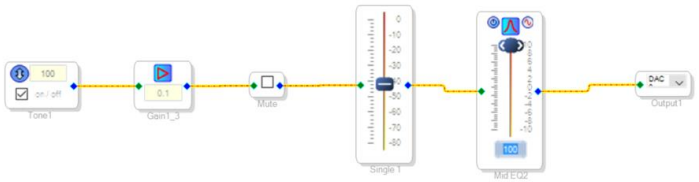
midt regionmidtjylland

Programmeringssprog





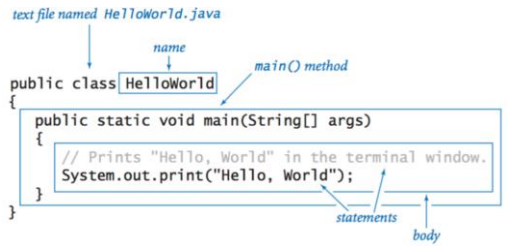




For the fourth year in a row, Rust is the most loved programming language among our respondents, followed close behind by Python, the fastest-growing major language today. This means that proportionally, more developers want to continue working with these than other languages.

VBA and Objective-C rank as the most dreaded languages this year. Most dreaded means that a high percentage of developers who are currently using these technologies express no interest in continuing to do so.

Python is the most wanted language for the third year in a row, meaning that developers who do not yet use it say they want to learn it.

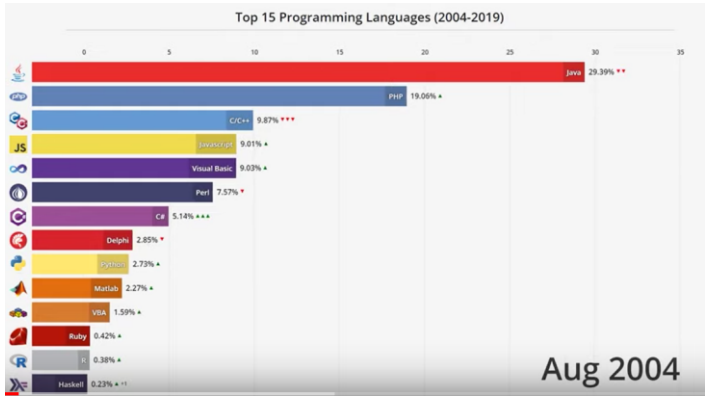


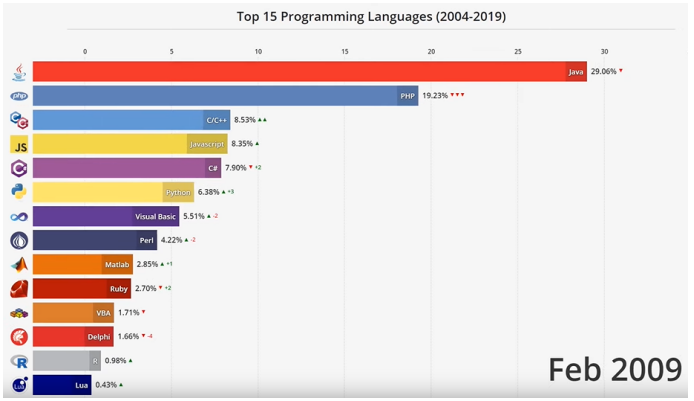
https://insights.stackoverflow.com/survey/2019#most-loved-dreaded-and-wanted

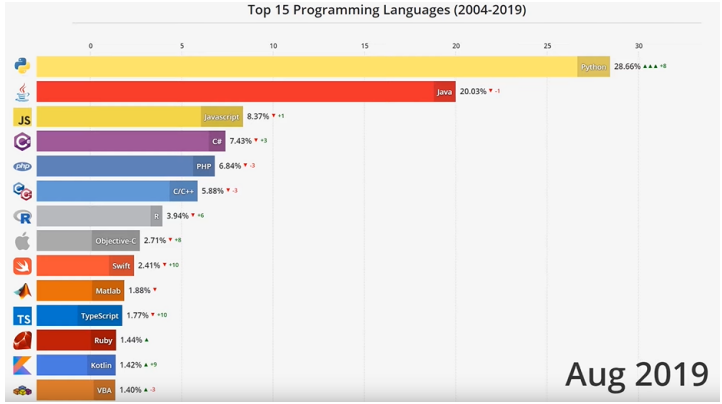
Projekt støttet af:

Programmeringssprog

Statistikker og lignende optællinger har altid nogle forudsætninger så man kan finde forskellige bud på ex. hvad der er mest udbredt så vær varsom. Men de 2 der er i denne præsentation indikerer at Python er udbredt og er blevet mere og mere anvendt over tid.







https://www.youtube.com/watch?v=yL704C1PI4o

Basis strukturer

Projekt støttet af:

midt regionmidtjylland

```
while <betingelse>:
    <gør dit>
    <gør dat>
```

The while Loop

With the `while` loop we can execute a set of statements as long as a condition is true.

https://www.w3schools.com/python/python_while_loops.asp

```
if <betingelse 1>:
    <gør dit når betingelse 1 er opfyldt>
    <gør dat når betingelse 1 er opfyldt>
elif <betingelse 2>:
    <gør noget andet dit når betingelse 2 er opfyldt>
    < gør noget tredje når betingelse 2 er opfyldt>
else:
    <gør noget femte når hverken betingelse 1 eller 2 er opfyldt>
```

Python Conditions and If statements

Python supports the usual logical conditions from mathematics:

- Equals: `a == b`
- Not Equals: `a != b`
- Less than: `a < b`
- Less than or equal to: `a <= b`
- Greater than: `a > b`
- Greater than or equal to: `a >= b`

https://www.w3schools.com/python/python_conditions.asp

HUSK ":" og indrykning er afgørende for at fortælle fortolkeren hvilken kode stump der hører til i løkker og udføres på baggrund af en betingelse!!

Et simpelt program

Projekt støttet af:
midt regionmidtjylland

```
1  # En variabel med navnet "a" som initielt får tildelt værdi "0" nul
2  a = 0
3  # En while løkke der gentages sålænge betingelsen (a<=5) "a mindre eller lig med 5" er opfyldt
4  while a<=5:
5      # udfører en handling/en funktion der udskriver værdien af variablen a i shell
6      print(a)
7      # en betinget sætning der tester om a er større eller mindre end 3 og udskriver det i shell
8      if a<3:
9          print("a mindre end 3")
10     elif a>3:
11         print("a større end 3")
12     else:
13         print("a er lig med 3")
14     # der lægges 1 til værdien i variablen a
15     a = a + 1
16 print("while er nu overstået - Kan både køre på PC/MAC og på Micro:Bit")
```

HUSK – tekstbaseret programmering kræver at man skriver det rigtige. Ellers forstår fortolkeren det ikke og man får en syntaks fejl. Der er også forskel på store og små bogstaver. Men prøv bare der er ikke noget der går i stykker og man får en fejlmelding i shell som man kan bruge til at få løst fejlen.

Brug microbit bibliotek

Projekt støttet af:
midt regionmidtjylland

```
1  from microbit import *
2
3  while True:
4      if button_a.is_pressed():
5          display.show(Image.HAPPY)
6      elif button_b.is_pressed():
7          break
8      else:
9          display.show(Image.SAD)
10
11 display.clear()
```

Microbit biblioteket indeholder bl.a. funktioner (også kaldet metoder) der kan fortælle om knap a og/eller knap b er blevet trykket ned/aktiveret, og til at udskrive ting på displayet.

Bemærk at ”button_a.is_pressed()” er en funktion der returnerer en værdi som er sand eller falsk (True eller False (None)) som så kan bruges som betingelse i en if … sætning.

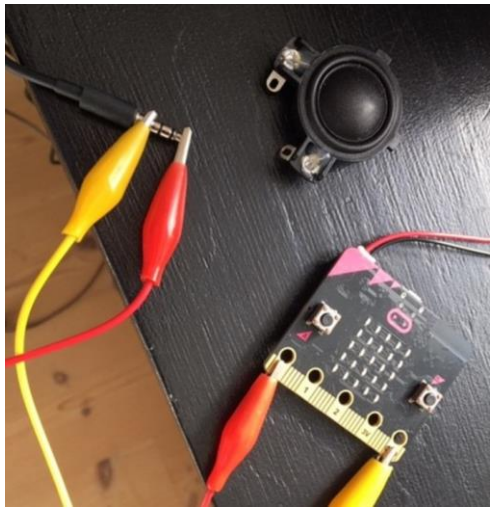
Der er en ”break” kommando i python generelt, som kan bruges til at komme ud af en igangværende løkke – Bemærk at betingelsen i while løkken er en konstant True, som jo aldrig vil blive ændret – så hvis ikke break blev brugt ville den aldrig stoppe og komme til den kode som ligger efter while løkken

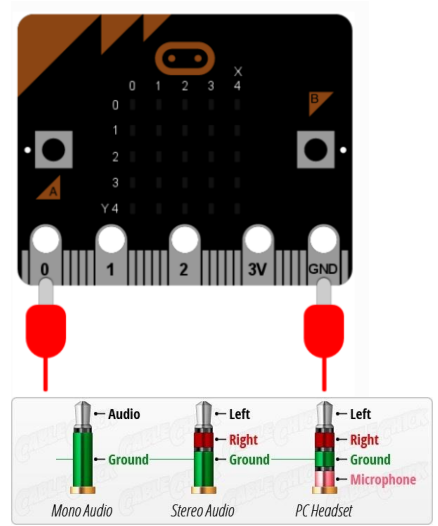
Prøv med andre billeder/evt. en tekst.
Prøv med display.scroll(”En tekst”) og se hvad der sker.
Se/Find flere eksempler i MicroBit/MicroPython dokumentationen.

Projekt støttet af:

Brug speech og music bibliotek

```
1 import speech
2
3 speech.say("Hi - Is there anyone out there")
```






```
1 import music
2
3 music.pitch(440,1000)
```

Tilslut hovedtelefon eller lille højtaler til MB og prøv speech og music funktionerne til venstre.

Eksperimenter med andre tekster i speech og andre kald til music. Find inspiration i dokumentationen. Der er for eksempel forud definerede melodier der kan spilles med music.play(…).

Findes i afsnittet ”Music” i ”Tutorial” sektionen i dokumentationen for MicroPython på MicroBit.

Kombiner

Projekt støttet af:
midt regionmidtjylland

```
1  from microbit import *
2  import music
3
4  while True:
5      if button_a.is_pressed():
6          display.show(Image.HAPPY)
7          music.pitch(440)
8      elif button_b.is_pressed():
9          break
10     else:
11         display.show(Image.SAD)
12         music.stop()
13
14 display.clear()
15 music.stop()
```

Tilføjet at der spilles en tone med frekvens 440 (Kammertonen) når knap a aktiveres – og at tonen stoppes når knap slippes, samt når knap b trykkes ned.

Når der ikke angives flere parametre/argumenter til funktionen ”music.pitch(440)” end de 440 så bliver den ved med at spille tonen til den aktivt bliver stoppet med funktionen ”music.stop()”

Man kan også kalde music.pitch med flere parametre. Ex. ”music.pitch(440,50)” – Det vil resultere i at tonen kun spiller i 50 millisekunder – og så vil ”music.stop()” ikke være så vigtig i koden ved siden af – og pitch bliver genstartet så længe knap a er aktiveret – Men det vil lyde anderledes – Prøv det..

PS – det vil fortsat være fornuftigt at bruge ”music.stop()” da det sikrer at PIN’s på MB er i en veldefineret tilstand når koden afsluttes.

Find grænsen for hvad pitch kan.

Projekt støttet af:
midt regionmidtjylland

```
1 from microbit import *
2 import music
3
4 for x in .....
```

Det hørbare område for lyd er 20-20000 Hz

music.pitch() funktionen kan kun håndtere frekvenser op til en værdi der ligger mellem 20 og 20000 Hz.

Lav eksperimenter med kode for at finde den værdi – Det står nemlig ikke i dokumentationen (mig bekendt) – men står den der er det også en måde at finde værdien på! ☺

Så hvad er største værdi man kan angive som frekvens i pitch funktionen uden at programmet crasher (går ned)?
Og den mindste værdi som får det til at gå ned?

Brug evt. en for løkke til at eksperimentere med..
Og måske over flere gange for at indsnævre..

Find grænsen for hvad pitch kan.

Projekt støttet af:
midt regionmidtjylland

```
from microbit import *
import music

for x in range(<start>,<slut>,<step>):
    music.pitch(x)
    sleep(100)
    print(x)
```

Det hørbare område for lyd er 20-20000 Hz

music.pitch() funktionen kan kun håndtere frekvenser op til en værdi der ligger mellem 20 og 20000 Hz.

Lav eksperimenter med kode for at finde den værdi – Det står nemlig ikke i dokumentationen (mig bekendt) – men står den der er det også en måde at finde værdien på! ☺

Så hvad er største værdi man kan angive som frekvens i pitch funktionen uden at programmet crasher (går ned)?
Og den mindste værdi som får det til at gå ned?

Brug evt. en for løkke til at eksperimentere med..
Og måske over flere gange for at indsnævre..

Find grænsen for hvad pitch kan.

Projekt støttet af:
midt regionmidtjylland

```
1 from microbit import *
2 import music
3
4 for x in range(20,20000,100):
5     music.pitch(x)
6     sleep(100)
7     print(x)
```

```
1 from microbit import *
2 import music
3
4 for x in range(<mindste der duede>,<den første som den gik ned med>,1):
5     music.pitch(x)
6     sleep(100)
7     print(x)
```

HUSK – Eksperimenter kan man lære af. Brug det til at få en bedre fornemmelse/forståelse af hvad MB'en kan/ikke kan. Eksperimenter er et godt supplement til at læse om tingen. Men begge dele er ikke så tossede. I dokumentationen eller på nettet ved at søge kan i finde inspiration til hvad man kan og så kan jeres egne eksperimenter gøre jer endnu klogere på den specifikke ting.

Opg. 3c – Lys til lyd.

Projekt støttet af:
midt regionmidtjylland

```
1  from microbit import *
2
3  while True:
4      x = display.read_light_level()
5      print(x)
6      sleep(100)
```

Først eksperimenteres lidt med sensoren, der læser lyd niveau.

Lav et simpelt program der læser lys niveau og putter det over i én variabel, der udskrives. Her bruger vi navnet 'x' til variablen. Det vil være en god ide at bruge et mere sigende navn i længere programmer. Det vil hjælpe med læsevenligheden mht. at forstå koden.

I dokumentationen står at man kan få værdier mellem 0 og 255. Stemmer det overens med det I oplever. Det I får udskrevet i shell med print() funktionen?

Vi fandt ud af at music.pitch() funktionen kan håndtere frekvenser fra nul op til og med 3906. Desuden ved vi at frekvensen skal over 50 for at vi får noget ud af det (Kan høre det ordentligt).

Så hvad skal der til for at få lys sensor input til at passe med en fornuftig pitch værdi?

Projekt støttet af:

regionmidtjylland

Opg. 3c – Lys til lyd.

```python
# 3c - lys til lyd.py
# Konvertering af lys intensitet til lyd (pitch) Version 1.1 10-Nov-2020
# Importer de nødvendige biblioteker/moduler.
# Den generelle microbit for at nå display.read_light_level() fundktionen
# og music for at nå music.pitch() funktionen
from microbit import *
import music

# Lys niveau kan have værdier mellem 0 og 255
# pitch kan håndtere værdier mellem 0 og 3906Hz. Vi kan høre fra ca. 50Hz.
# - derfor lys niveau*14+50 som 'fornuftig' frekvens til pitch funktionen
while True:
    x = display.read_light_level()
    music.pitch(x*14+50)
    print(x)
    sleep(100)
```

Gange 15 er også ok!

Et svar. Ganger med 14 og adderer 50 til lys niveau for at konvertere til en pitch. HUSK – Der er ikke kun eet rigtigt svar mht. kode!
Det er en god ide at tilføje kommentarer til sin kode. Det kan hjælpe med til at man forstår den bedre når man på et senere tidspunkt kommer tilbage til den. Også værdifuldt hvis andre skal se og forstå koden. Bemærk også reference til det navn som koden er gemt under og en for information om hvilken version det er. Vil også være værdifuldt når man skal finde 'tilbage'.

Projekt støttet af:
midt regionmidtjylland

Opg. 4b – Magnet felt til lyd.

```
1  from microbit import *
2
3  while True:
4      x = compass.get_field_strength()
5      print(x)
6      sleep(100)
```

Først eksperimenteres lidt med sensoren, der læser magnet feltet.

Lav et simpelt program der læser styrken af magnet feltet og putter det over i én variabel, der udskrives. Her bruger vi navnet 'x' til variablen. Her kan I se at det måske ville være godt med et andet navn – for det er jo IKKE længere det samme som i opgaven med lysintensiteten.

I dokumentationen står der IKKE noget om hvilke værdier der kan komme tilbage fra funktionskaldet. Kun at det er i nano tesla. Så der er kun en vej til at finde ud af det. Prøv/Eksperimenter med en magnet omkring Micro:Bitten!

Hvad skal der til for at få magnet felt værdi omsat til en pitch der kan håndteres af funktionen music.pitch()?

Man kan i stedet for compass.get_field_strength() også prøve med funktionerne:
compass.get_x(), compass.get_y() og compass.get_z()
Og dermed få en fornemmelse for hvad forskellen er.

Projekt støttet af:

regionmidtjylland

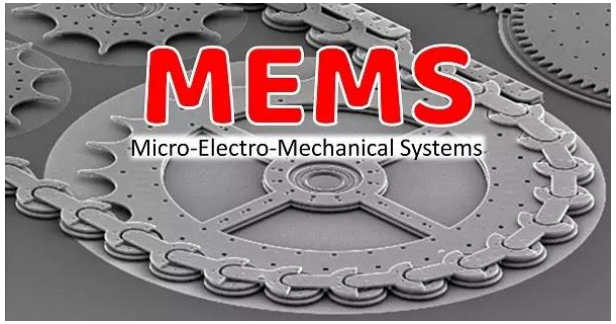
Opg. 4b – Magnet felt til lyd.

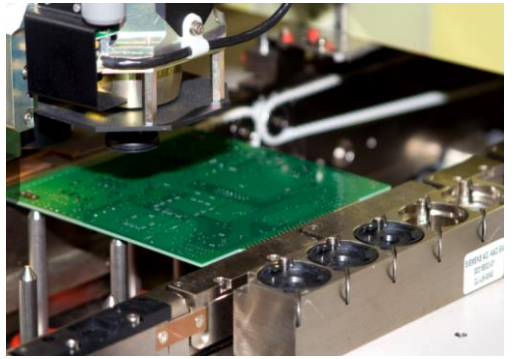
```
# 4b - mag felt til lyd.py
# Omsæt magnet felt styrke til lyd (pitch) Version 1.1  0-nov-2020
# Først importeres det generelle microbit bibliotek (for at nå compass() funktionen)
# og music (for at nå music.pitch() funktionen)
from microbit import *
import music

# Da felt styrken ser ud til at kunne nå op til ca. 2.5 mill og altid er positiv
# divideres med 1000 for at ramme indenfor det som pitch kan håndtere.
# MEN skal desuden få det til at være et heltal og ved normal division (/) får vi decimaler.
# - derfor bruges funktionen int() til at konvertere til heltal.
# Adderer 50 til for at vi kan høre det.
while True:
    x = compass.get_field_strength()
    print(x)
    music.pitch(int(x/1000)+50)      # Kan også bruge floor division // - Se python aritmetiske operatorer
    sleep(100)
```

Et svar. Deler med 1000 og adderer 50 til magnet felt værdi for at konvertere til en pitch MEN skal også lave det til heltal (ikke decimaltal).
Fordi music.pitch() funktionen forstår kun positive heltal.
HUSK – Der er ikke kun eet rigtigt svar mht. kode!

Projekt støttet af:

MEMS.



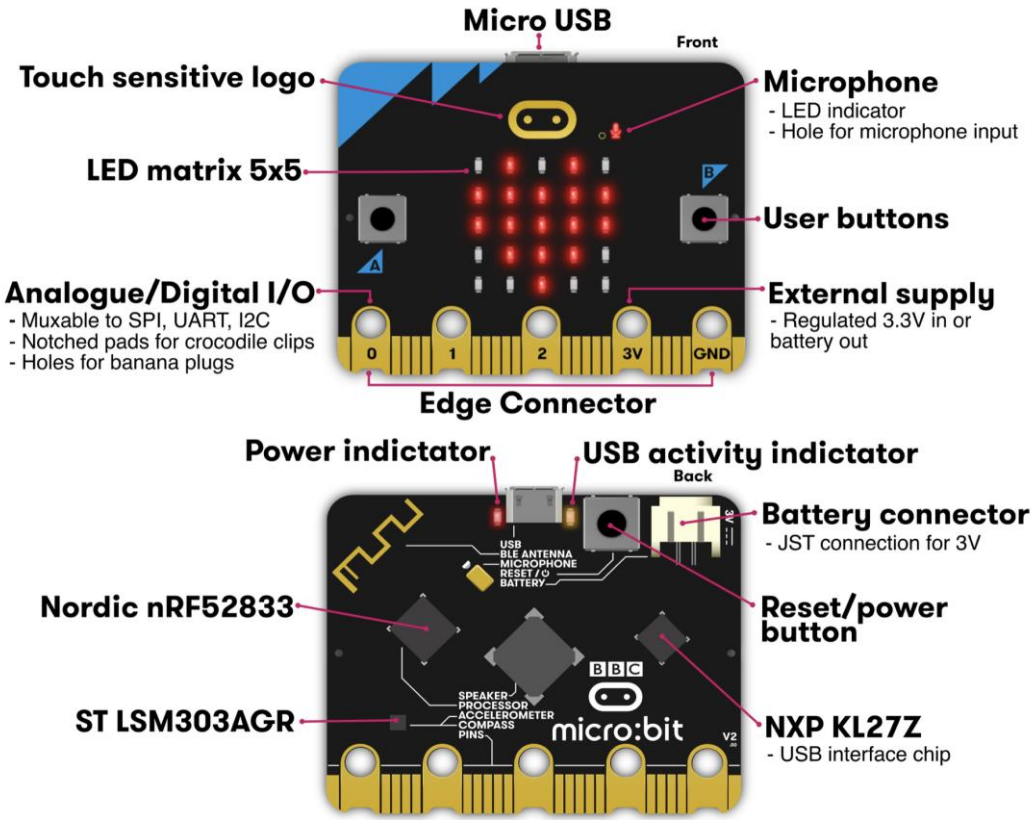


Mindre – men nok så vigtigt - en helt anden fremstillings proces og kan indgå i produktion af PCB (printed circuit board) på linje med elektroniske komponenter

Projekt støttet af:

midt regionmidtjylland

Ex. Micro:Bit v2



Der kommer en MEMS mikrofon og højtaler på næste version af MicroBitten.
Et eksempel på at sætte nye funktioner på print (PCB) når man får nye teknologiske muligheder.
Det bliver noget lettere at lave de øvelser vi har været igennem med den når man ikke skal bøvle med ekstern højtaler og krokodillekabler! ☺
MEN kvaliteten af lyden bliver IKKE så god som det i har i jeres headset! Og ex. speech funktionen vil være sværere at forstå på den PCB monterede højtaler

Projekt støttet af:

Opg. 5b – Accelerometer til lyd.

```
1  from microbit import *
2
3  while True:
4      acc_x,acc_y,acc_z = accelerometer.get_values()
5  #    acc_all = accelerometer.get_values()          # alternativ - tuple/sæt af værdier
6      print(acc_x)
7  #    print(acc_all[0])                             # alternativ - udpeg i tuple
8      sleep(100)
```

Først eksperimenteres lidt med sensoren, der læser g–værdier fra accelerometeret .

Hvad skal der til for at få magnet felt værdi omsat til en pitch der kan håndteres af funktionen music.pitch()?

Læs evt. om tuples og lister i w3schools python..

Opg. 5b – Accelerometer til lyd.

Projekt støttet af:
midt regionmidtjylland

```
# 5b - acc til lyd.py
# Omsæt accelerometerværdi i x-aksen til lyd (pitch) Version 1.0 12-Nov-2020
# Først importeres det generelle microbit bibliotek (for at nå accelerometer funktionen)
# og music (for at nå music.pitch() funktionen)
from microbit import *

# Bruger accelerometer.get_values() til at hente værdier i alle 3-akser, selvom
# programmet her kun bruger værdi i x-retningen. Dvs vip højre/venstre med knap a til venstre.
# Prøv med de andre akser for at blive fortrolig med dem.
#
while True:
    acc_x,acc_y,acc_z = accelerometer.get_values()
    print(acc_x)
#    music.pitch(abs(acc_x))                    # den hurtige løsning
                                                # - abs() laver neg til pos

    freq = acc_x+1500           # adderer tilstrækkelig stor værdi
    if freq>50 and freq<3907:   # tester at det er i området for at være sikker
        music.pitch(freq)
    sleep(50)
```

Opg. 6a – Radio.

Projekt støttet af:
midt regionmidtjylland

```python
# 6a modtager
from microbit import *
import radio

radio.on()

while True:
    besked = radio.receive()
    print(besked)
    sleep(100)
```

Med sender 2 bliver beskederne løbende nummereret fra 1-9.
Så kan man eksperimentere med forskellig værdi i sleep() kaldet i hhv. sender og modtager og se hvad det betyder for det man modtager.

Tømmer man postkassen oftere end postbuddet har været der modtager man ingenting. (None)
Tømmer man den for sjældent kan det ende med at postbuddet ikke kan aflevere posten fordi postkassen er overfyldt, og modtageren får ikke alle brevene.

```python
# 6a - sender 1.py - enkelt besked til alle på default kanal med default sendestyrke
from microbit import *
import radio

radio.on()

while True:
    radio.send("Hej alle i 1x - Alt OK ? ")
    sleep(100)
```

```python
# 6a - sender 2.py - Besked til alle med nr tilføjet så man kan se om alle kommer igennem
from microbit import *
import radio

radio.on()
besked_nr = 1

while True:
    radio.send("Hej alle i 1x - Alt OK ? "+ str(besked_nr))
    print(besked_nr)
    besked_nr = besked_nr+1
    if besked_nr>9:
        besked_nr = 1
    sleep(100)
```

Projekt støttet af:

regionmidtjylland

Opg. 6b – 5b på 2 MB's via Radio.

```python
# 6b - Sender.py
# Accelerometer data fra x-akse sendt på radio som tekst - Version 1.0 18-Nov-2020
# Først importeres det generelle microbit bibliotek (for at læse accelerometer værdier)
# samt radio for at  kunne sende værdierne på radio.
from microbit import *
import radio

display.show("6b-S")              # Udskriver en tekst på display for at identificere MB når der kommer strøm på

radio.on()
radio.config(channel=60)          # hvis kanal angives skal det være samme kanal i både sender og modtager
                                  # Hvis kanal ikke angives er default kanal nr 7 i begge
while True:
    acc_x,acc_y,acc_z = accelerometer.get_values()
    f_num = acc_x+1000            # adderer 1000 for at få de fleste værdier positive - se modtager..
    print(f_num,"  ",type(f_num)) # test udskrift for at vise hvad vi har fået fra funktionskaldet til accelerometeret
    radio.send(str(f_num))        # radio.send() funktionen sender tekst strenge så tal f_num skal konverteres
    sleep(50)                     # Bør bruge samme værdi i både sender og modtager. Samme takt.
```

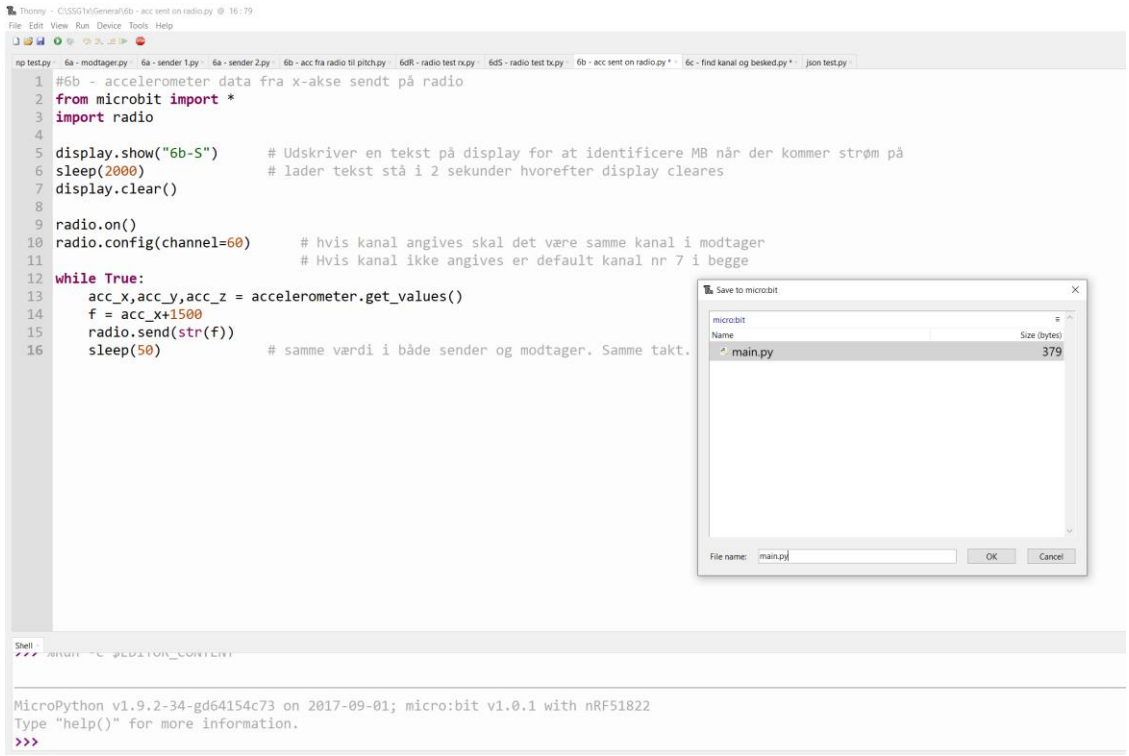
Find koden på filer i Teams – generel

- Læs og forstå (hjemmeopgave)
  - Indsæt kommentarer i sidste del af modtager. Hvad bruges nof_nones til?
- Få den op at køre på 2 MB's i gruppe
- Brug tildelt kanal i gruppen!

```python
# 6b - Modtager.py
# Forventer at modtage en tal værdi fra radio som kan bruges til at lave en lyd. - Version 1.0 18-Nov-2020
# Vil IKKE virke hvis der modtages andet end et tal.
# Sikrer selv at værdi er mellem 50 og 3907 før den kalder music.pitch() med det tal der modtages
from microbit import *
import music
import radio

display.show("6b-M")              # Udskriver på MB display for at identificere

radio.on()
radio.config(channel=60)          # Hvis kanal angives skal det være samme kanal i både sender og modtager
                                  # Hvis kanal ikke angives er default 7 i begge.
nof_nones = 0

while True:
    f_str = radio.receive()
    print(f_str,"  ",type(f_str))    # Test udskrift af hvad vi modtager i shell incl typen af det
    if f_str:                        # Det samme som "f_str != None:"  men IKKE som "f_str == True:"
        f_num = int(f_str)           # Men altså 'noget er modtaget' fra funktionen. Her en streng vi konverterer
        nof_nones = 0
        if (f_num>50) and (f_num<3907):
            music.pitch(f_num)
        else:
            music.stop()
    else: nof_nones = nof_nones+1
    if nof_nones > 20 : music.stop()
    sleep(50)
```

Projekt støttet af:

Gemme program lokalt på MB



Man kan gemme et program lokalt på Micro:Bitten og ved at tilslutte en batteripakke så kan det program køre lokalt uden at MB er tilsluttet PC/MAC.

Det gør man ved at gå til filer>save as og så vælge MicroBitten. Man skal så gemme program med navnet ”main.py” (alle bogstaver SKAL være små).

Det program startes så snart MB får strøm.

Når I så efterfølgende skal gennem noget/eller have anden kode til at køre på den MB skal I først stoppe programmet med STOP og sikre jer at I har prompt i shell som vist. Ellers kan I få div fejl meldinger/eller ikke kunne gemme etc..

Opg. 6c – Find kanal/besked.

Projekt støttet af:
midt regionmidtjylland

```python
# 6c - find kanal og besked
from microbit import *
import radio

radio.on()
# Trin 1: find kanal
for x in range(0,83,1):
    radio.config(channel=x)
    f=radio.receive()
    print("Channel: ",x," Modtaget besked: ",f)
    sleep(100)                                  # vigtigt med samme timing som i senderen!
                                                # Er det nul kan den nå igennem alle
                                                # inden der modtages noget. 100 ms er lang tid...
```

Find hemmelig besked.
2 trin:

- Find kanal med for løkke
- Check op på kanalen for at finde alle beskeder.

Kanal var ”76” og besked var at I skulle eksperimentere med sende styrken i radio.config(power=x), hvor x iflg. dokumentationen kan være mellem 0 og 7 med 0 som den svageste.

```python
# 6c - find kanal og besked
from microbit import *
import radio

radio.on()

# Trin 2: find besked - Check den kanal ud I har fundet ...
radio.config(channel=<kanal nummer fundet i trin 1>)    # Trin 3 : Måske har man kommet een for langt mht kanal nummer, så prøv igen...
while True:
    f = radio.receive()
    print(f)
    sleep(100)
```

Projekt støttet af:

midt regionmidtjylland

Opg. 6d – Eksperimenter med sendestyrke.

```python
# 6dM - sende styrke test.py
# Modtager til radio sende styrke eksperiment/test. Version 1.0 18-Nov-2020
# Giver et kort lydsignal samt udskriver "Heart beat" i shell og viser et hjerte i MB display, hvis den modtager noget
# Hvis der ikke modtages noget udskriver den "None" i shell.
# importer de funktonaliteter der skal bruges - her både music og radio udover det generelle MB bibliotek

from microbit import *
import radio
import music

# Sæt radio op til at kommunikere på bestemt kanal - her 10 og tænd for radio så MB kan sende/modtage
radio.config(channel=10)   # Brug kanal som I har til den enkelte gruppe!
radio.on()

# For at identificere MB som hørende til opgave 6d - modtager delen!
# Den er ikke på nogen måde afgørende for testen af modtagelse af 'signal' på radio.
display.show("6d-M")
sleep(1000)          # Venter 1 sekund

while True:
    heart_beat = radio.receive()
    if heart_beat:               # Det samme som "heart_beat != None" - Betingelse "noget modtaget" vs "ingenting modtaget"
        music.pitch(440,20)      # Lyd i 20 ms med frekvens 440 når signal modtaget
        print("heart beat")      # teksten "heart beat" i shell når radio signal er modtaget
        display.show(Image.HEART) # Hjerte vises i display på MB
    else:
        print("None")            # Tekst "None" i shell når radio signal er modtaget
        music.stop()             # Clear pin setting på music når signal ikke modtages. Har her ikke stor effekt
        display.clear()          # Clear lokal display
    sleep(100)                  # Pause på 100 ms - hvis den samme pause er i sender delen så kan de følges ad
                                # og er sender og modtager 'helt' tæt på hinanden kommer der ingen 'Nones' i modtager.
                                # Er de langt nok fra hinanden kommer der kun 'None' retur fra kaldet til radio.receive().
                                # Det er naturligvis også tilfældet hvis der ikke er nogen sender på kanalen (slukket).
```

Find koden på filer i Teams – generel

- 6dM – sendestyrke test.py
- 6dS – sendestyrke test.py
- Få den op at køre på 2 MB's i gruppe.
- Senderen med batteri pakke så I kan bevæge jer væk fra modtageren
- Brug tildelt kanal i gruppen!

```python
# 6dS - sendestyrke test.py
# Sender med forskellig sende styrke. Version 1.0 18-Nov-2020
# Sende styrken er initielt sat til 0, men kan ændres med tryk på knap a
# Den aktuelle sendestyrke vises i MicroBittens display.
from microbit import *
import radio

radio.on()                  # Tænd for radio
radio.config(channel=10)    # Sæt kanal (mellem 0 og 83 som ikke er brugt af andre).
                            # Brug det kanal nummer I er tildelt i gruppen i både sender og modtager!
p=0                         # Variabel til at holde sendestyrke
radio.config(power=p)       # Sæt sendestyrken. Værdi fra 0 til 7 med stigende sende styrke - initielt sat til 0

display.show("6d-S")        # Viser hvilken opgave koden hører til i MB dispaly S for at vise det er sender delen.
sleep(1000)                 # Venter et sekund, hvorefter den starter med at vise sendestyrken
display.show(str(p))

while True:
    radio.send(str(p))
    if button_a.is_pressed():   # Hvis knap er aktiveret tælles p een op og sende styrken ændres og udskrives på display.
        p=p+1
        if p>7: p=0
        radio.config(power=p)
        display.show(str(p))
    sleep(100)                  # Benyt samme værdi i både sender og modtager
                                # - Modtager har chancen for at modtage alle beskeder og hvis den ikke gør
                                #   er det sandsynligvis fordi modtager er for langt væk.
                                #   Dvs None i modtager indikerer at sender er for langt væk iforhold til sende styrke (eller slukket)
```

Opg. 6d – Eksperimenter med sendestyrke.

Projekt støttet af:
midt regionmidtjylland

```python
# 6dM - sende styrke test rssi.py
# Modtager til radio sende styrke eksperiment/test. Version 1.0 18-Nov-2020
# Giver et kort lydsignal samt udskriver "Heart beat" i shell og viser et hjerte i MB display, hvis den modtager noget
# Hvis der ikke modtages noget udskriver den "None" i shell.
# importer de funktonaliteter der skal bruges - her både music og radio udover det generelle MB bibliotek

from microbit import *
import radio
import music

# Sæt radio op til at kommunikere på bestemt kanal - her 10 og tænd for radio så MB kan sende/modtage
radio.config(channel=80)   # Brug kanal som I har til den enkelte gruppe!
radio.on()

# For at identificere MB som hørende til opgave 6d - modtager delen!
# Den er ikke på nogen måde afgørende for testen af modtagelse af 'signal' på radio.
display.show("6d-M")
sleep(1000)          # Venter 1 sekund

while True:
    details = radio.receive_full()
    if details:                    # Det samme som "details != None" - Betingelse "noget modtaget" vs "ingenting modtaget"
        music.pitch(440,20)        # Lyd i 20 ms med frekvens 440 når signal modtaget
        display.show(Image.HEART) # Hjerte vises i display på MB
        msg, rssi, timestamp = details
        print("heart beat, rssi: ",str(rssi))       # teksten "heart beat" + rssi værdi i shell når radio signal er modtaget
    else:
        print("None")              # Tekst "None" i shell når radio signal IKKE er modtaget
        music.stop()               # Clear pin setting på music når signal ikke modtages. Har her ikke stor effekt
        display.clear()            # Clear lokal display
    sleep(100)                    # Pause på 100 ms - hvis den samme pause er i sender delen så kan de følges ad
                                  # og er sender og modtager 'helt' tæt på hinanden kommer der ingen 'Nones' i modtager.
                                  # Er de langt nok fra hinanden kommer der kun 'None' retur fra kaldet til radio.receive().
                                  # Det er naturligvis også tilfældet hvis der ikke er nogen sender på kanalen (slukket).
```

Nu også en modtager med sendestyrke måling (rssi), hvis I kan bruge det til noget.

- 6dM – sendestyrke test rssi.py

Til Badge:

- Tænk over trigger. Hvad skal til for at lave en 'sikker' detektion af "Tæt på" versus "Langt nok fra"
- Tænk over tilstand og skift mellem tilstande. I Alam tilstand versus i OK tilstand

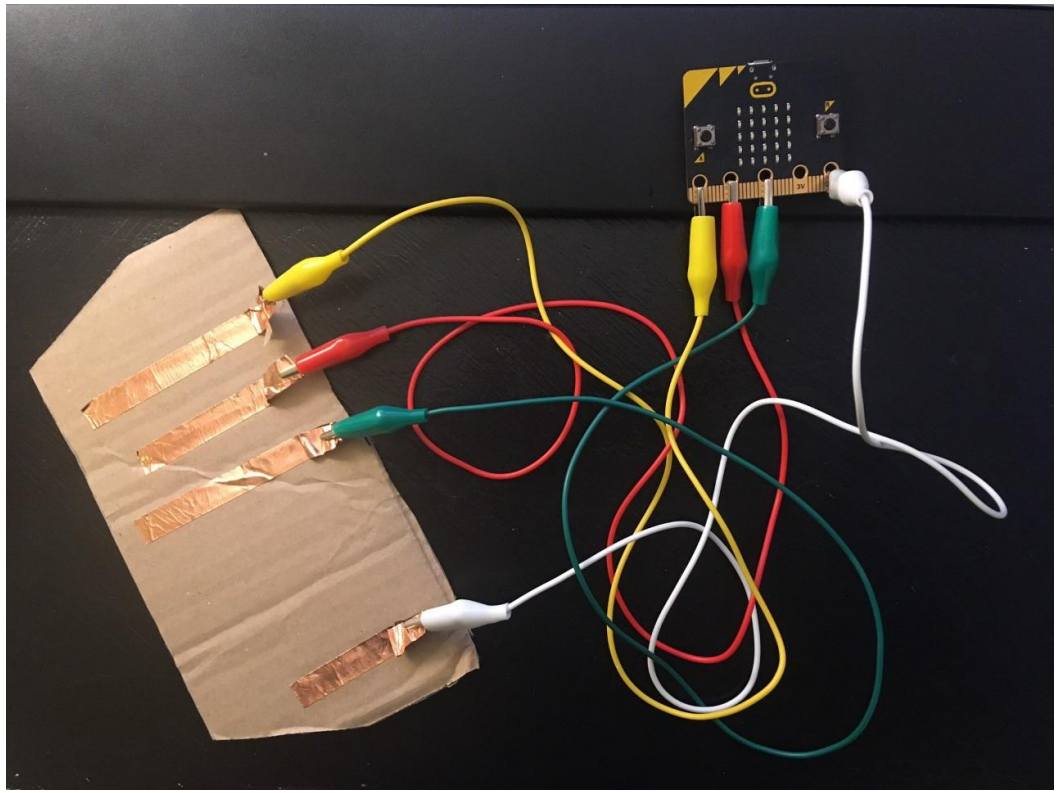
Trigger / Tilstand	Tæt på	Langt nok fra
ALARM	Gentag alarm eller?	Stop evt. alarm Skal der evt være et 'klap på skulderen'? Skift til tilstand OK
OK	Giv alarm og skift til tilstand ALAM	Ingenting

Projekt støttet af:
midt
regionmidtjylland

Opg. 7b – Pin 0,2,3.is_touched() på radio

I får noget kobber tape og et par krokodilleledninger mere.

- Sæt 4 ledninger på pin 0,1,2 samt GND på én MB
- Sæt 4 strimler kobbertape på bord eller et stykke pap og forbind til MB som vist



Projekt støttet af:

Opg. 7b – Pin 0,2,3.is_touched() på radio

Find koden på filer i Teams – generel

- 7b - Modtager.py
- 7b - Sender.py
- Put sender koden i den MB som har tilsluttet de 4 krokodilleledninger og kobberstrimler.
- Tilslut 2 krokodilleledninger til hovedtelefon/højtaler på MB der får modtager koden
- Brug tildelt kanal i gruppen!

```python
# 7b - Sender.py
# Sender 3 værdier for hhv pin 0, 1 og 2 på radio - Version 1.0 18-Nov-2020
# Først importeres det generelle microbit bibliotek (for at læse on pin 0-3 is_touched()) og vise på display
# samt radio for at  kunne sende værdierne på radio.
from microbit import *
import radio

display.show("7b-S")              # Udskriver en tekst på display for at identificere MB når der kommer strøm på
sleep(1000)                       # Venter 1 sekund for inden vi går videre for at besked kan ses færdig.

radio.on()
radio.config(channel=60)          # hvis kanal angives skal det være samme kanal i både sender og modtager
                                  # Hvis kanal ikke angives er default kanal nr 7 i begge
while True:
    p0 = pin0.is_touched()
    p1 = pin1.is_touched()
    p2 = pin2.is_touched()
    if p0:
        key = "1"
    elif p1:
        key = "2"
    elif p2:
        key = "4"
    else:
        key = "0"

    radio.send(key)               # radio.send() funktionen sender og det er key så ingen grund til konvertering
    display.show(key)             # viser key på MBs display
    sleep(50)                     # Bør bruge samme værdi i både sender og modtager. Samme takt.
```

```python
# 7b - Modtager.py
# Forventer at modtage en værdi for een tast 1-7 og 0 for ingen tast. - Version 1.0 18-Nov-2020
# For hver af de 7 taster spilles een bestemt tone. Modtages 0 stoppes tone
# Import af de biblioteker der er brug for
from microbit import *
import music
import radio

display.show("7b-M")              # Udskriver på MB display for at identificere
sleep(1000)

radio.on()
radio.config(channel=60)          # Hvis kanal angives skal det være samme kanal i både sender og modtager
                                  # Hvis kanal ikke angives er default 7 i begge.
nof_nones = 0

while True:
    key_str = radio.receive()
    if key_str:                   # noget modtaget
        nof_nones = 0
        print(key_str)
        if   key_str=="1":
            music.pitch(440)      # A4 - kammertonen.
        elif key_str=="2":
            music.pitch(493)      # B4
        elif key_str=="3":
            music.pitch(523)      # C5
        elif key_str=="4":
            music.pitch(587)      # D5
        elif key_str=="5":
            music.pitch(659)      # E5
        elif key_str=="6":
            music.pitch(698)      # F5
        elif key_str=="7":
            music.pitch(783)      # G5
        elif key_str=="0":
            music.stop()
    else: nof_nones = nof_nones+1
    if nof_nones > 20 :           # sender er slukket eller udenfor rækkevidde. Stop og reset.
        music.stop()
        music.reset()
        print("Ikke modtaget noget længe")
        nof_nones = 0
    sleep(50)
```

Projekt støttet af:

Opg. 7c – 7 key's ud fra de 3 pin's

Pin 0,1,2.is_touched() funktionerne returnere enten True eller False afhængig af om der er en kortslutning eller ej. Dvs. der er 8 forskellige kombinationer af de 3

- Opgaven er nu at lave en ny version af senderen, som kan sende 8 forskellige key's til modtageren.

p0	p1	p2	Key
False	False	False	"0"
True	False	False	"1"
False	True	False	"2"
True	True	False	"3"
False	False	True	"4"
True	False	True	"5"
False	True	True	"6"
True	True	True	"7"

```python
# 7b - Sender.py
# Sender 3 værdier for hhv pin 0, 1 og 2 på radio - Version 1.0 18-Nov-2020
# Først importeres det generelle microbit bibliotek (for at læse on pin 0-3 is_touched()) og vise på display
# samt radio for at  kunne sende værdierne på radio.
from microbit import *
import radio

display.show("7b-S")              # Udskriver en tekst på display for at identificere MB når der kommer strøm på
sleep(1000)                       # Venter 1 sekund for inden vi går videre for at besked kan ses færdig.

radio.on()
radio.config(channel=60)          # hvis kanal angives skal det være samme kanal i både sender og modtager
                                  # Hvis kanal ikke angives er default kanal nr 7 i begge
while True:
    p0 = pin0.is_touched()
    p1 = pin1.is_touched()
    p2 = pin2.is_touched()
    if p0:
        key = "1"
    elif p1:
        key = "2"
    elif p2:
        key = "4"
    else:
        key = "0"

    radio.send(key)               # radio.send() funktionen sender og det er key så ingen grund til konvertering
    display.show(key)             # viser key på MBs display
    sleep(50)                     # Bør bruge samme værdi i både sender og modtager. Samme takt.
```

Når I har løst opgaven og vist det frem så kan I enten:

- Arbejde med de 2 opgaver I har fået i starten af forløb. - Eller:
- Få 2 sendere og udvid til 15 forskellige toner i modtager. – Eller:
- Byg eller skitser en fysisk/mekanisk 'knap' eller tastatur der kan bruge det I lige har været igennem med kobbertape. Hvad skal der ellers bruges af materiale?

Opg. 7c – 7 key's ud fra de 3 pin's

Projekt støttet af:
midt regionmidtjylland

```
# 7c - Sender.py
# Sender 8 værdier for alle kombinationer af pin 0, 1 og 2 på radio - Version 1.1 25-Nov-2020
# Først importeres det generelle microbit bibliotek (for at læse om pin 0-3 is_touched()) og vise på display
# samt radio for at  kunne sende værdierne på radio.
from microbit import *
import radio

display.show("7c-S")              # Udskriver en tekst på display for at identificere MB når der kommer strøm på
sleep(1000)                       # Venter 1 sekund for inden vi går videre for at besked kan ses færdig.
radio.on()
radio.config(channel=60)          # hvis kanal angives skal det være samme kanal i både sender og modtager
                                  # Hvis kanal ikke angives er default kanal nr 7 i begge

while True:
    p0 = pin0.is_touched()                # Læser et øjebliksbillede af pin 0,1 og 2 ind i variabler
    p1 = pin1.is_touched()                #         p0      p1      p2         Key
    p2 = pin2.is_touched()                #-----------------------------------------------------
    if p0 and p1 and p2:                  #        True    True    True        "7"
        key = "7"                         #
    elif not p0 and p1 and p2:            #        False   True    True        "6"
        key = "6"                         #
    elif p0 and not p1 and p2:            #        True    False   True        "5"
        key = "5"                         #
    elif not p0 and not p1 and p2:        #        False   False   True        "4"
        key = "4"                         #
    elif p0 and p1 and not p2:            #        True    True    False       "3"
        key = "3"                         #
    elif not p0 and p1 and not p2:        #        False   True    False       "2"
        key = "2"                         #
    elif p0 and not p1 and not p2:        #        True    False   False       "1"
        key = "1"                         #
    else:                                 #        False   False   Fasle       "0"  sidste mulighed derfor bare else
        key = "0"
    radio.send(key)               # radio.send() funktionen sender og det er key så ingen grund til konvertering
    display.show(key)             # viser key på MBs display
    sleep(50)                     # Bør bruge samme værdi i både sender og modtager. Samme takt.
```

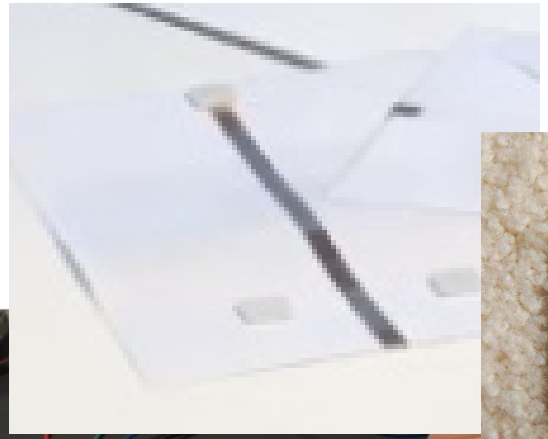
Et svar. Der er andre!

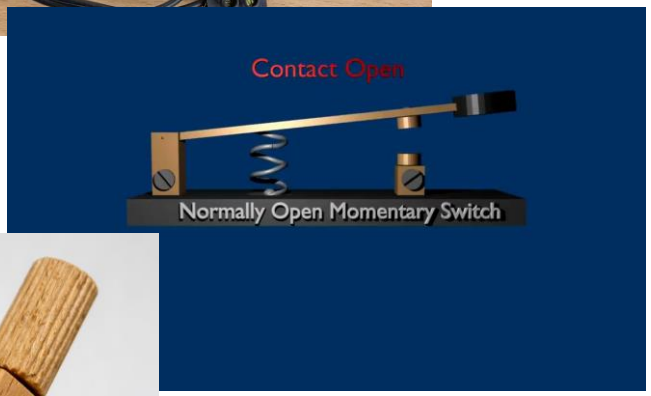
Kan forbedres ved at man checker key 2 gange med ex 20 ms mellemrum og ser om værdien er den samme. Ellers svært at undgå nogle af de primære (1,2,4) på vejen til eller fra de andre.

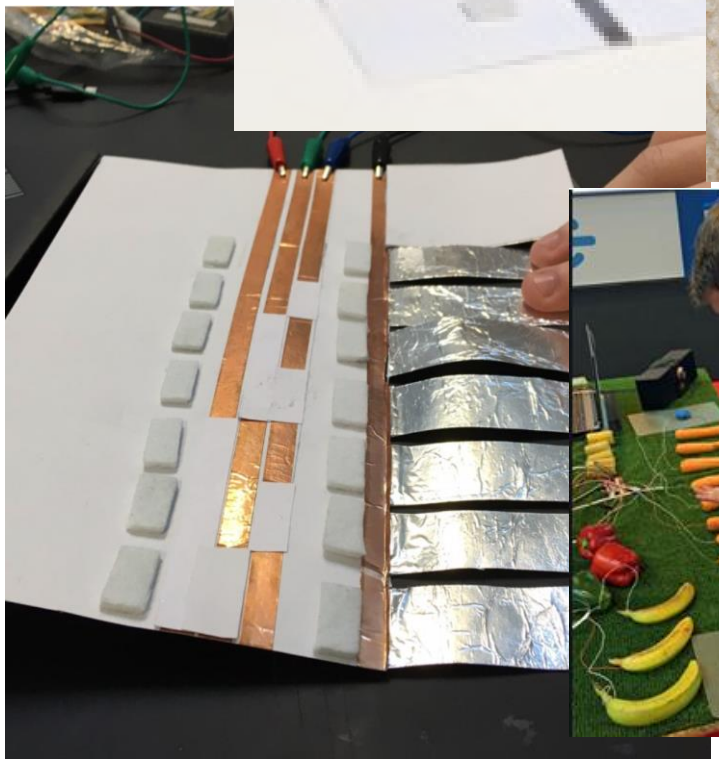
Projekt støttet af:

midt regionmidtjylland

Opg. 7c – Byg evt. noget fysisk (knap/switch)







https://www.youtube.com/watch?v=4-cq70KFxhc

Opg. Badge – Et bud baseret på 6d.

Projekt støttet af:

midt regionmidtjylland

```python
# 6d-Badge - med tællere og sender.py
# 6d modtager lavet med tællere (nof_heart_beats og nof nones). Version 1.1 02-Dec-2020
# Desuden indsat radio send hvergang der testes for modtaget signal. Sendestyrke sat til 0. (svagest)
# Dermed er det en første forbedring på vej mod en corana badge løsning.
#
# importer de funktonaliteter der skal bruges - her både music og radio udover det generelle MB bibliotek
from microbit import *
import radio
import music
#
# Sæt radio op til at kommunikere på bestemt kanal - her 80 og tænd for radio så MB kan sende/modtage
radio.config(channel=80,power=0)   # Brug kanal som I har til den enkelte gruppe!
radio.on()

# For at identificere MB som hørende til opgave 6d - men med første forbedring mod en corona badge
display.show("6d-Badge")
sleep(1000)          # Venter 1 sekund

nof_heart_beats = 0
nof_nones = 0

while True:
    radio.send("Heart beat")
    heart_beat = radio.receive()
    if heart_beat:                    # Det samme som "heart_beat != None" - Betingelse "noget modtaget" vs "ingenting modtaget"
        nof_heart_beats = nof_heart_beats + 1   # Antal gange heart beat er modtaget i træk tælles een op.
        nof_nones = 0                 # reset tæller for antal nones i træk - dvs antal gange i træk der IKKE er modtaget noget.
    else:                             # ingenting modtaget
        nof_heart_beats = 0           # reset tæller for antal heart beats i træk
        nof_nones = nof_nones + 1     # antal gange der ikke er modtaget noget i træk tælles een op.
    sleep(100)                        # Pause på 100 ms - hvis den samme pause er i sender delen så kan de følges ad

    if nof_heart_beats > 7:           # Mere end 7 heart beats modtaget i træk - det vil vi sige er tæt på
        music.pitch(440,20)           # Lyd i 20 ms med frekvens 440 når signal modtaget - Hvad er en passende alarm lyd for "for tæt på"
        print("8*heart beat i streg") # test udskrift "8*heart beat i streg" i shell når radio signal er modtaget 8 gange i træk
        display.show(Image.HEART)     # Hjerte vises i display på MB - Hvad vil være passende som advarsel/alrm signal?
        nof_heart_beats = 0           # reset tæller for antal heart beats i træk
    if nof_nones > 3:                 # Mere end 3 nones i træk - det vil vi sige betyder der er langt nok i mellem
        print("4*None i streg")       # Tekst "4*None i streg" i shell når radio signal IKKE er modtaget 4 gange i træk
        music.stop()                  # Clear pin setting på music når signal ikke modtages. Har her ikke stor effekt
        display.clear()               # Clear lokal display - skal der være en indikator på at alt er ok - IKKE for tæt på længere?
        nof_nones = 0                 # reset tæller for antal nones i træk - dvs antal gange i træk der IKKE er modtaget noget.
                                      # for at den ikke skal løbe løbsk...
```

Find koden på filer i Teams – generel

- 6d – badge – med tællere og sender.py
- Få den op at køre på 2 MB's i gruppe.
- 2 med batteri pakke så I kan bevæge jer rundt
- Brug tildelt kanal i gruppen!
- Test i rigtig brugssituation – på brystet og i forskellige situationer.
    - Duer det?
    - Passer det med anbefalinger/krav til afstandskrav?
    - Hvilken lyd/billede passer til at få bruger til at flytte sig væk?
    - Sammenlign med andre advarselssignaler i form af lys og lyd I møder i hverdagen!
- Hvordan kunne et rigtigt produkt se ud?
    - Hvad kræves der for at nå til et 'rigtigt' produkt.
    - Hvad skal det indeholde?
- Dokumenter jeres overvejelser og test resultater i logbog med henblik på at det skal bruges til fremlæggelse.

Opg. Badge – et andet bud på løsning – Main loop

Projekt støttet af:
midt regionmidtjylland

Trigger / Tilstand	Tæt på	Langt nok fra
ALARM	Gentag alarm eller?	Stop evt. alarm Skal der evt være et 'klap på skulderen'? Skift til tilstand OK
OK	Giv alarm og skift til tilstand ALAM	Ingenting

```
# Tilstande
ALARM = 0
OK    = 1
tilstand = OK # initielt er vi i OK tilstand.
"
```

```
# Hoved program løkke

while True:
    nærved = trigger()
    if nærved and tilstand == OK:
        lav_alarm()
        tilstand = ALARM
    elif nærved and tilstand == ALARM:
        gentag_alarm()
    elif not nærved and tilstand == ALARM:
        stop_alarm()
        tilstand = OK
    else:
        alt_ok()
```

Kode findes på Teams under filer med navnet "badge ex.py"

Opg. Badge – et andet bud på løsning – trigger funktion

Projekt støttet af:
midt regionmidtjylland

```
#
# Antal omgange i een trigger undersøgelses periode, samt tærskel værdi for at sige nogen er tæt på.
#
ANTAL_OMGANGE = 20
TÆRSKEL       = 16  # Det antal ud af ANTAL_OMGANGE som bruges til at sige der er nu kommet post nok til at sige der er nogen nærved

def trigger():
    nof_heart_beats = 0
    for x in range(0,ANTAL_OMGANGE):
        sleep(100)
        radio.send("Heart_beat")                    # Hver gang vi checker for post sender vi post - for at have ens funktion i begge ender
        heart_beat = radio.receive()
        if heart_beat:                                 # Det samme som "details != None" - Betingelse "noget modtaget" vs "ingenting modtaget"
            nof_heart_beats = nof_heart_beats + 1
    print("Antal heart beats: ",nof_heart_beats," Antal omgange: ",ANTAL_OMGANGE)  # test udskrift som kan fjernes i endeligt produkt!!
    if nof_heart_beats>TÆRSKEL:
        t = True  # Tæt på
    else:
        t = False # Langt nok fra
    return t
```

Projekt støttet af:

regionmidtjylland

Opg. Badge – et bud på løsning – de 4 handlinger

```
def lav_alarm():
    # tilpas med lyd og billed/animation i MicroBit display som I synes det skal være
    # Test udskrift - som kan fjernes i endeligt produkt
    print("Nu er der alarm - Der er een som er kommet for tæt på")
    # Her vises sur smiley og en et lille beep med høj frekvens, som kører i baggrund
    display.show(Image.SAD)
    music.pitch(1000,500,wait=False)


def gentag_alarm():
    # tilpas med lyd og billed/animation i MicroBit display som I synes det skal være.
    # Test udskrift - som kan fjernes i endeligt produkt
    print("Der er fortsat een for tæt på - Hvad skal vi gøre for at få vedkommende til at tage afstand?")
    # Her laver vi bare et lille beep med høj frekvens, og til at køre i baggrund
    music.pitch(3000,500,wait=False)


def stop_alarm():
    # tilpas med lyd og billed/animation i MicroBit display som I synes det skal være.
    # Test udskrift - som kan fjernes i endeligt produkt
    print("Den der var for tæt på er nu kommet på afstand - Skal vi give positiv kvittering og hvad er det? Hvad duer?")
    # Her cleares display og evt lyd stoppes
    display.clear()
    music.stop()

def alt_ok():
    # tilpas med lyd og billed/animation i MicroBit display som I synes det skal være.
    # Test udskrift - som kan fjernes i endeligt produkt
    print("Alt OK ingen for tæt på - Skal der være noget som indikerer at badge er i live?")
    # Her sætter vi midterste pixel til lav intensitet.
    display.set_pixel(2,2,3)
```