Afleveringsopgave 1

Glenn, Emily, Erlendur, Emil

Hvad er programmering?

At programmere er en proces, du bruger til at udvikle for eksempel software, der skal bruges til vores ESP32. I de fleste tilfælde bruger du et programmeringssprog, i vores tilfælde er det enten Python eller MicroPython.

Da du har et programmeringssprog, skal du også have en IDE til at skrive, ændre eller debugge din kode. Koden er grundlæggende input, der udfører en opgave, såsom beregninger, datahentning fra en database eller kommunikation med en server, osv."

Hvad er en ESP32?

Grundlæggende er en ESP32 også kaldet Espressif System Platform 32 en mikrocontroller (chip), der hovedsageligt benyttes i alt fra infotainment-systemer, fjernbetjeninger, droner, energistyringssystemer, IoT, blodtryksmålere osv. Enheden har en 32-bit processor. ESP32 kan ligeledes kommunikere trådløst igennem dens indbyggede bluetooth/wifi.

Hvad er MicroPython, og fordele/ulemper?

Simpelt beskrevet er MicroPython, som også fremgår af navnet, en forenkling af programmeringssproget Python (grundlæggende version). Sproget bruges hovedsageligt til udvikling af mikrokontrollere for at forenkle selve processen med Python.

2 Fordele

Let syntax, som der er baseret på python, Open Source

2 Ulemper

Mangel på API´s, Sproget har et stort resource brug

Hvad er en kodeeditor / IDE?

Helt basalt kaldes en IDE en Integrated Development Environment, eller på dansk kaldes den en kodeeditor. En IDE bruges eksempelvis til det helt basale, såsom at skrive din kode i for eksempel MicroPython, redigere din egen kode eller kode, der er blevet sendt til dig. Du har også muligheden for at hente kode fra eksempelvis åbne open-source GitHub-projekter. Da der er mange forskellige IDE'er at vælge imellem, har de alle tilfælles, at de kan redigere, tilføje eller fejlfinde din og andres kode. Din IDE bruges til at holde styr på dit projekt.

Hvad er variabler, og hvad kan de bruges til?

Simpelt forklaret er en variabel data som er gemt i symboler (tal, tekst, lister, ord, strings etc.)

Vores eksempel:

// Opret variabler og tildel værdier
x = 12
navn = 1A Gruppe 6

// Beregning, der tilføjer 6 til værdien af x (12) og den gemmer så resultatet i variablen kaldet resultat
resultat = x + 6

// Udskriv variabler
print("Værdien af x er:", x)
print("Navnet er:", navn)
print("Resultatet af beregningen er:", resultat)

Her er printet

Værdien af x er: 12
Navnet er: 1A Gruppe 6
Resultatet af beregningen er: 18

Hvilke datatyper er der i Python, og hvad bruges de til?

Datatyper i Python skrives i deres engelske form for lettere forståelse, når der skal kodes med dem. Vi har ligeledes tilføjet eksempler på de forskellige datatyper:

Integers

// -6, 6, 66
Hele tal

Floats

// -6.6, 6.6, 66.6
Bruges til decimaltal

Strings

// "Gruppe 6", 'Gruppe'
Bruges til tekst, både til dobbelt- eller enkelte anførselstegn

Sets

// {18, 6, 12}, {'b', 'a', 'c'}
Data/elementer ubestemt rækkefølge

Dictonaries

// dyreoplysninger = {'hunderace': 'golden retriever', 'farve': gylden}
// uddannelsesoplysninger = {'uddannelseinstution': 'kea', 'studerende': 5471}.
Associationspar, print(dyreoplysninger) = {'hunderace': 'golden retriever', 'farve': 'gylden'}

Lists [ ]

Kan ændres
// [6, 12, 18], ['windows', 'macos', 'linux']. - Der kan tilføjes eller fjernes elementer
Samlinger af data

Tuples ( )

Kan ikke ændres
// (6, 12, 18), ('fredag', 'lørdag', 'søndag') - Der kan ikke tilføjes eller fjernes elementer
Samling af data, der ikke skal ændres

Hvad er Pythons print funktion, og hvad kan den bruges til?

Funktionen print() bruges til at vise data. Der vises et eksempel her:

gruppe = "Gruppe 6"
print("Du har bestået,", gruppe)

Dette bliver printet:
Du har bestået, Gruppe 6

Hvordan laver man kodekommentarer i Python, og hvordan fungerer det?

Kommentarer oprettes med dette # symbol og bruges til at assignerer en beskrivelse eller notater til koden i din IDE.

Hvilke kontrolstrukturer findes i Python, og hvordan fungerer de?

Vi har forklaret de forskellige strukturere med kort beskrivelse og eksempler.

If-else

Denne struktur udføres baseret på betingelse.

```
bestået = 12
if bestået >= 12:
   print("Bestået")
else:
   print("Afvist")
```

For-loop

Denne struktur udføres baseret på gentagelse over en sekvens af elementer

```
grupper = ["gruppe 1", "gruppe 2", "gruppe 4", "gruppe 6"]

for gruppe in grupper:
print(gruppe)
```

Og så for du printet grupperne: gruppe 1, gruppe 2, gruppe 4, gruppe 6

While-loop

Denne struktur udføres baseret på en grundlag at den forsætter indtil den er nået sin satte grænse.

```
Gruppenummer = 1
while tal <= 6:
print(tal)
Gruppenummer += 1
```

Og så for du printet gruppenummer 1 indtil den når nummer 6

Try-except

Denne struktur tillader end at håndtere undtagelser

```
try:
søkende = int(input(“Antal søskende: "))
print(“Denne studerendes søskende antal er:", søskende)
except ValueError:
print(“Benyt venlist integers.")
```

I har antal søskende har jeg skrevet nul I stedet for symbolet 0 og så får jeg denne besked = Benyt venlist integers., grundet jeg ikke har brugt et heltal. Jeg ville alternative få denne besked hvis jeg havde brugt et heltal: Denne studerendes søskende antal er 6.

Break

Denne struktur stoppes ved anvendelse af en betingelse

```
grupper = [“gruppe 1", “gruppe 2", “gruppe 4", “gruppe 6"]

for gruppe in grupper:
print(f"Denne gruppe har bestået{gruppe}")
if gruppe == “gruppe 6":
print(“Gruppe 6 har bestået.")
break
```

Continue

Denne struktur bruges til at springe en del af loopet over

```
grupper = ["gruppe 1", "gruppe 2", "gruppe 3", "gruppe 4", "gruppe 5", "gruppe 6"]

for gruppe in grupper:
if gruppe == "gruppe 6":
continue
print(f"Behandler {gruppe}...")
```

Pass

Denne struktur bruges til at vise at en handling ikke er implementeret

```
grupper = ["gruppe 1", "gruppe 2", "gruppe 3", "gruppe 4", "gruppe 5", "gruppe 6"]

for gruppe in grupper:
if gruppe == "gruppe 6":
print(f"Ignorerer gruppe 6, siden projektet ikke er færdigt.")
pass
else:
print(f"Behandler gruppe: {gruppe}”)
```

Hvilke operatører kan bruges til kontrolstrukturer?

if, elif, else, and, or, not, ==, !=, <, <=, >, >=, in, not in, is, is not, for, while, break, continue, pass, try, except, finally, raise

Hvad er Micropythons Pin klasse og hvad kan man med den?

Pin-klassen i MicroPython giver dig kontrol over metalbenene på en ESP32 mikrocontroller, så du kan styre forskellige komponenter på dit educaboard, såsom LED'er og sensorer, osv. Pin-klassen er afgørende, da den bestemmer, hvordan mikrocontrolleren skal styre de øvrige komponenter.