Hvad er **programmering**?

*Programmering* er opbygningen af instruktioner og sekvenser, eller programmer, som computere kan læse, og forstå for at løse forskellige opgaver og problemer. Vi bruger oftest et *high-level programmeringssprog* (en type programmeringssprog der er så abstrakt fra rent binær som muligt, f.eks. Python) til at opbygge disse programmer.

Hvad er en **ESP32?**

*ESP32-serien* er et system på en *microchip controller* – ofte brugt til undervisning eller studieforbrug. De er også set i industrielle, og kommercielle produkter. ESP32-systemet er udviklet af Espressif Systems, i Shanghai.

Hvad er **MicroPython** og hvad er fordele og ulemper ved det?

*MicroPython* er en softwareimplementering af et programmeringssprog hovedsageligt kompatibelt med *Python*, som er skrevet i *C*. Det er optimeret med intentionen om at kunne køre på en micro controller. Man ville også kunne bruge Python, men hvis man vælger at bruge MicroPython, skal man ikke selv optimere programmet til brug på en micro controller – det er klaret for dig.

Hvad er en **code editor / IDE**?

En code editor/IDE (integrated development environment) er basalt set bare den tekst-editor som man bruger til at skrive, og i nogle tilfælde også eksekvere kode. En IDE har oftest mange features, der gør kodeskrivningen lidt nemmere, eller giver programmøren forskellige værktøjer til at optimere deres kode.

Hvad er **variabler** og hvad kan de bruges til?

I programmering er *variabler* essentielt kasser, fyldt med information, som man kan bruge til at forkorte sin kode længere nede i programmet. Altså en form for *shorthand* man selv definerer. I Python kan et variabel f.eks. defineres ved at skrive:

MyVariable = ”I’m a variable!”

Derefter kan man altid kalde informationen ”I’m a variable!” ved bare at skrive MyVariable.

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

Python har de følgende datatyper indbygget:

* Tekstdata: str (string)
* Numeriskdata: *int, float, complex*
* Sekvensdata: *list, tuple, range*
* Mapping-data: *dict*
* Set-data: *set, frozenset*
* Boolean-data: *bool*
* Binærdata: *bytes, bytearray, memoryview*
* None-data: *NoneType*

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

*print()-funktionen* bliver brugt til at ”printe” – at skrive – en specificeret besked eller information til skærmen, *Shell*-display, eller en anden standard enhed. Dette kan være en *string*, eller et hvert andet objekt, men vil blive konverteret til en string inden print.

Hvordan kan man lave **Kode kommentarer** i Python kode og hvordan fungerer det?

Ved at bruge et *Hash* (#) og et mellemrum vil alt på den kodelinje anses som en kodekommentar. Man kan også gøre brug af tre citationstegn (''' eller """) rundt om en ”blok” af tekst. Den vil så i mange IDE’er blive grå eller på anden måde visuelt markeret som en kommentar. Kommentarer er til for at forklare hvad kodesegmenters funktioner er, evt. til samarbejdspartnere eller til en selv i fremtiden. Altså dokumentation. De kan også bruges til at markere sektioner som der f.eks. skal optimeres, eller på andre måder fikses. Kodekommentarer fungerer på en noget så simpel måde, at det er tekst der blive ekskluderet i eksekveringen af koden. Altså det er tekst som ikke bliver ”læst” af computeren.

Hvilke **kontrolstrukturer** findes der i Python og hvordan fungerer de?

I Python kan man gøre brug af sekventiel kontrolstruktur, altså at programmet går igennem kommandoerne linje for linje – dette er den standard kontrolstruktur. Dertil er der også selektiv kontrolstruktur som der tjekker *conditions* og *branching* før den eksekverer kode, og repeterende kontrolstruktur der bliver brugt til at *loope* forskellige kodeblokke igen og igen.

Hvilke **operatører** kan bruges til **kontrol strukturer**?

F.eks /\*+-, <>%&, etc. Kan blive brugt mellem to værdier eller variabler, med forskellige resultater.

Hvad er MicroPythons **Pin klasse** og hvad kan man med den?

.