Python

Är et interpreterat programmeringstad. Högnivåprogrammeringpråk stöder både funktinoel och objektorienterad programering. Har dynamisk typning. Kan köras i två lägen. Som skript eller interaktivt. Finns många olika bilbiotek för vetenskapliga tillämpingar mm.

Lågnivå programeringspråk ligger nära maskin, nollor och ettor. Högnivå är lätt att att förstå för oss och kräver mer av datorn att förstå.

Interpreterat programmeringsspråk. Är en uppdelning och skiljs från Kompilerade där man använder textredigerare eller IDE – man skriver källkod sen används kompilator som översätter till maskinkod (binär). Därefter länkare som skapar körbar exe fil från det.

Graphical user interface

Description automatically generated

Interpretearde språk som python och mathlab kan man också använda texteditor eller IDE för skriva Sedan används interpretator som utför programmet rad efter rad. Det skapar således inte binär kod. De kallas därför skriptspråk istället för programmeringsspråk.

A screenshot of a computer

Description automatically generated with medium confidence

Finns fördelar och nackdelar med dessa. Olika snabba att köra och översäatta, Läsa och förstå. Olika specifika till processorerna.

Table

Description automatically generated with low confidence

När man väl har kompilerat ett program måste man inte kompiler det igen, mwn swr äe bweownsw OS. Måste också ta hand om fel efter kompliering. Och även om det är effektivare (snabbt), ligger hela i datorminnet så tar mer RAM.

De rekomenderar att man använder Visal studio code för man kan köra som skript och som interaktivt.

Dynamisk typning innebär att man inte måste deklerar vad variablerna har för datatyp utan den fattar vad som ät int och vad string

**Print** har lite oliak formateraringar. =end” ger till exempel att det skrivs på samma rad och (f” ...”) fsträng som placeras med objekt i klamerparanteser. Ex Print(f”Multiplikation av {a} och {b} är {c}”). Där är bokstäverna objekt.

I python kan man också skriva saker på samma rad ex a, b = 3, 5

Graphical user interface, application

Description automatically generated

**= input( ” text ”)** Programmet stoppar in det användaren ger. Blir sträng så måste omvandla till numeriskt tal

**Int(arg)** Omvandlar arg till int.

Kan göra det direkt i sig också

A picture containing graphical user interface

Description automatically generated

f- strängar anges i utskrivningsformat i klammerparantesre efter obejktsnamn följt av kolon. Ex om man skriver {a:5) kommer fältet a vara fem tecken långt om man skriver c:05 kommer tomma tecken ersättas med nollor.

När man arbetar med långa tal kan man använda \_ ex 1\_000\_000 de ignoreras då. Vill man att de ska spereras med komma skriver man {a:, }

**Float(obj:X.Yf)** Gör om till float

X är länd på fält och Y är antal sifforer efter decimalpunkten och f flytt tal. Skriver man e skrivs det i exponentiell form. Skriver man ex {a:5.2f} kommer man få två siffror efter decimalpunkt och 5 tecken totalt

Operatorer!!

Table

Description automatically generated

Om man vill använda utökade tilldelningsoperatorer

A screenshot of a computer

Description automatically generated with medium confidence

Om man vill göra matimatik behöver man Biblioteket math. (trigonometrin är x vinkel i radianer)

**Import bibliteknamn** .

När man vill anävnda det för man sedan skriva biblioteknamn.funktionen exempelvis x = math.pi/6 är i radianer.

**#** Kommentar!!

Komplexa tal

Använder biblioteket **cmath.** De är två dimentionella. Har en realdel och en imaginär del

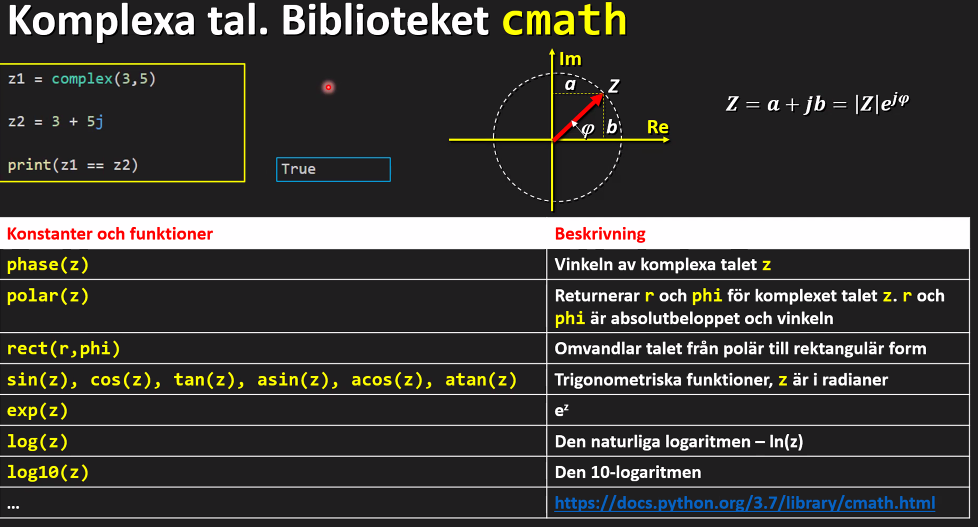
Text

Description automatically generated J är roten ur minus ett vilket ger den imaginära komponenten. Det i sidan är i rektangulär form först med a + jb och dianoäl form är absolutbeloppet mm.

Kan presentera det som punkt på graph enligt

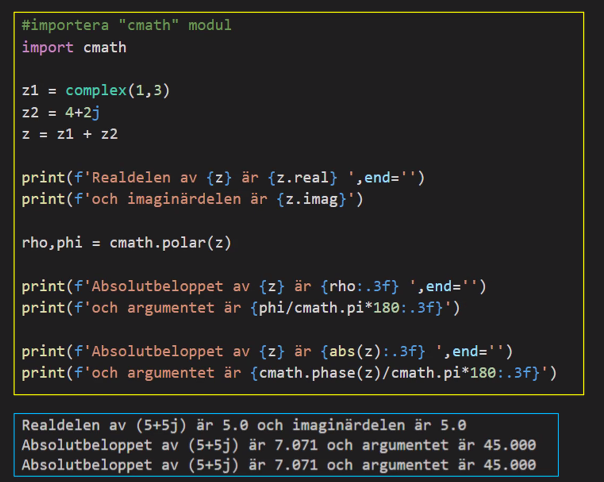
Diagram, schematic

Description automatically generatedFör att skapande i python kan man anävnda **complex(reel, imaginär). Eller z = 3 + 5j.**

****

Z här är det inkommande komplexatalet. Polar omvanldar rektangulär presentation till polär-foprmen. Polär kan ta beloppet och vinkeln. Rect gär motsatsen.

Exempel po arbte med komplexa tal



Boolean.

Variabler fårn denna data typ kan antingen vara True eller False. Viktigt att komma gemener spelar roll. Måste börja med versal för att få det som bool. Kan använda andra andra data typer för att skapa Boolsika typer. Alla värden har ett Bool värde. Endast 0 blir False. -1, 1, 10, 2, 1 alla andra blir true

Table

Description automatically generatedOperatorer för jämförelser som ger en bool variable

Logiska operatorer.

Graphical user interface, text, application, email

Description automatically generated

Logiska operatorer tra in booliska värden och retunerar även booliska värden.

Prioritert hos operatorer

A picture containing graphical user interface

Description automatically generated

Bitvis här ingår inte men får vara med för att ha fullständig tablell.



Detta blir således FALSE eftersom 0<0 är false.

**Uniär operator** är minuteckent i minu fem men när man har a – b så har det lägre prioritet

Flödeskontroll

Diagram

Description automatically generated

If- satser

**If – logiskt utryck**: Beroende på om True eller false kommer en eller flera satser utföras. Satser som ingår skrivs indragna. Kan fortsätta dem med **else**: Kan också korta ner det. Korsuytrck

Text

Description automatically generatedText

Description automatically generated

Om man har flera utryck

Text

Description automatically generated

Exempel på att selektera om ett tal är ett heltal eller ej. .floor avrundar nedåt

Text

Description automatically generated

Nestade if-satser är när man har flera ifsatser i en ej om talet är reelt och om det är reelt om det är naturligt

Ex -2 är ett heltal, men inte naturligt tal. 3 är Naturligt tal och 1.02 är flytande tal.

Viktigt att dra in rader så man ser vad som händer. Verkar också som det har funktion.

Text

Description automatically generated

Repetitionssatser – While och for. loops

**While logiskt\_utryck.** Om det är sant utför man en elelr flera satser och sen går man upp och titta igen. Upprepar tills logiskt utryck blir falskt. Ger möjligthet till oändlig repititoin

For anänder operationer är interationen

Text

Description automatically generated**For** variable (iterator i) **range (**först, sist, steg): först och sist och steg måste vara heltal, men kan vara negativ. Sist inkluderars inte.

Om vi inte anger måsta operator kommer steg bli 1 och start bli 0.

Exempel på for-satser

A screenshot of a computer

Description automatically generated with medium confidence

Iteratorn kan inte endras i for-satsen utan bara för resterande del av loopen.

Nyckelord

* **Continue** gå till nästa iteration
* **Break** avbryter irterationsatsen

A screenshot of a computer

Description automatically generated with medium confidence

Repetionssatser while-else, for-else

Finns inte på samma sätt i många andra programeringsspråk..om repitionsatsen slutförs utan break kommer else att utföras.

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated with low confidenceExempel

Men avbryter man med break innan kommer else inte att skrivas ut.

Sekvenser

Finns flera och i denna kurs kommer vi tala om strängar, tupler och listor. Skevenser är samlingar av ordnade element. Finns således en ordning, alla har sin plats

* Str’ngar är som sekvens av tekcen. De är immutable vilket betyder att man itne kan ändra dem. Sätt att formatera sina skapade sätrngar nedan:

Diagram

Description automatically generated

Indexering av strängar

Kan göra det med både positiva och negativa. **Enumerate()** ger ger ett par med två värden. Första blir index och andra blir värde

