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

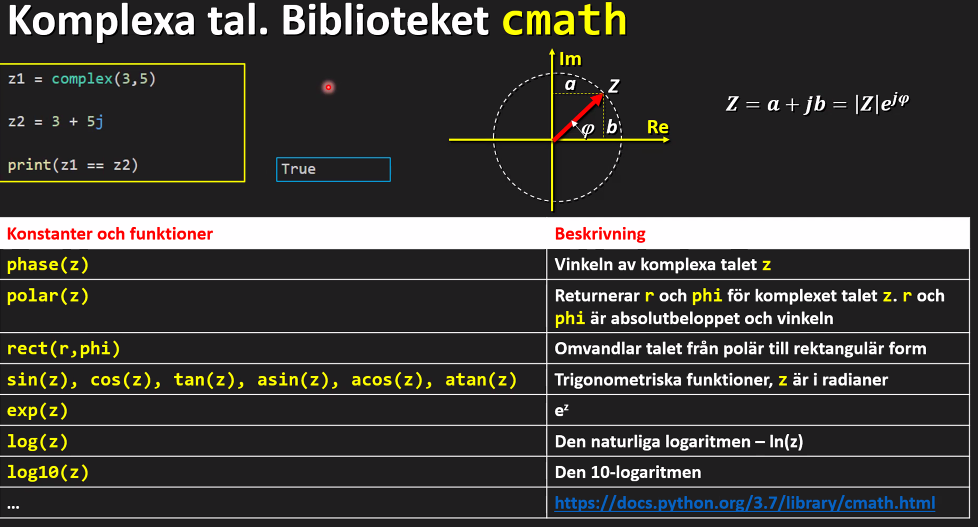
Text

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

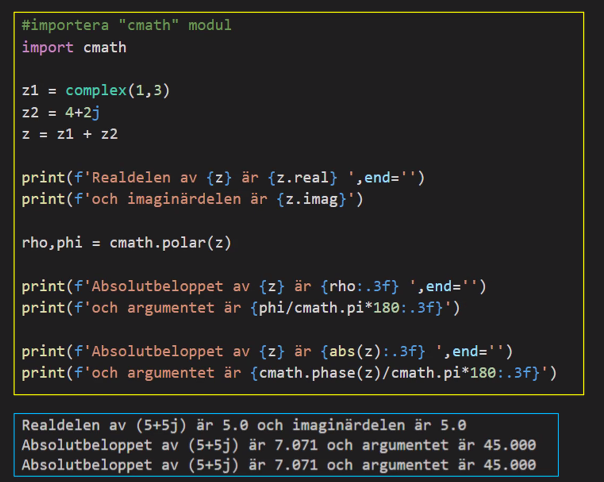
Diagram, schematic

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

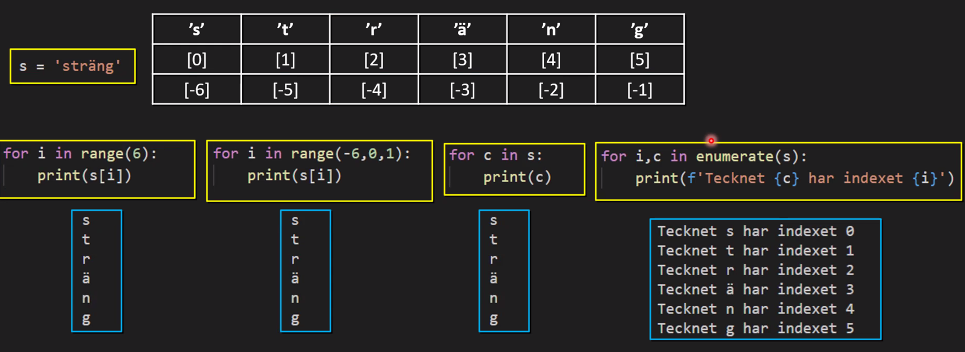
* Str’ngar är som sekvens av tekcen. De är immutable vilket betyder att man itne kan ändra dem. Sätt att formatera sina skapade sätrngar nedan:

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



Föreläsning 3

* Gekensa funktion och operatorer för sekvenser
* Tupler och listor
* Multiplel tilldelning
* List comprehension
* Flerdimensionella listor (varje element är en annan lista)

Referenser

Bättre att använda ordet ”objekt” än variabel. Varje objekt har tre egenskaper, id, typ och värde

* Id är objektets adress och man för det genom funktionens id
* Varabls namn är referens till objektet. Ex x = 5 är 5 objektet och x refrensen. De har sedan ett gömt **id** också

Text

Description automatically generated

Om man går vidare ock söger att x = y och sen skapar ett objekt till som är z = 3 kommer vi se att x och z har samma id eftersom de refererar till 3 båda två. 3 har således ett id-fast. Objektet är 3 med tre referenser (har samma id)

**Is** är en logisk operator som kollar på om två referesner pekar på samma objekt. Ex x is y kommer ge True.

Diagram

Description automatically generatedOm man ersätter ett element i en lista kommer det dock ha samma id.

Varje element i listan har alltså eget id

Om det är imutable data typ som tuple och skapa en lika tuple som har samma värden kommer de få samma id.

Listor

Om man börjar med att skpa en lista med tre element. L1 = L då har den fått en till referens. Men om vi vill skapa en ny lista som har samma värden måste man skapa en kopia då använder man funktion .copy(). Ett annat sätt är att använda funktion **list** där objektet man kopierar är

Graphical user interface, text

Description automatically generated

Således **is** jämför id, pekar de på samma objekt. Medan operator == undersöker om de har samma värden.

Set

Det är en annan datatyp. Mängder på svenska. Ockås en datasamling men den är oordnad av unika elment. Kan alltså inte inedxeras eller skivas. De är unika så om man har {a,b,c} = {b,c,a} = {a,a,b,c}. Frozen set är den immutable varianten

Text

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated

Oliak sätt att lösa uppgiften

Text

Description automatically generated

De olika kommandon för mängder

Schematic

Description automatically generated

Diagram, schematic

Description automatically generated

Dictonary – Associativ tabell

**Dict** ordnad samling av par. **Key:value** key är immutable och value kan vara både mutable och immutable. Det är en mängd men varje element ärett par.

Exempelvis om man har en tabell med namn och ålder A picture containing text, scoreboard

Description automatically generated

Text

Description automatically generated

Om man sedan gör en dict av det där namn och ålder hör ihop.

A screenshot of a computer

Description automatically generated with medium confidence

Här gör man det isätllet av tupler där första värde blir key och nästa value. Ett annat sätt är att göra zip för att placera dem tillsammans

Kommandon för associativ tabell dict

Graphical user interface, text

Description automatically generated

Nycklar i hakparanteser så kan söka dem där.

Text

Description automatically generated

Key som iterator kommer gå igenom alla elmment i tabll men.

El blir en tuple där första element blir namn och nästa blir år.

Funktioner

Funktion är en del av koden som körs när den anropas. Varje funktion har en viss bestämd iuppgift. Python har en mänd olika inbygda, men kan också definiera egen med **def.**

Text

Description automatically generatedKommandon / parametrar kommer dras in

**Return** Kan ge ett eller flera objekt. För att kunna använda funktion måste man först defineras och sen anropas

Graphical user interface, text

Description automatically generatedEtt exempel kan vara om man vill skapa en funktion för att kvadrata en

Graphical user interface, text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

Den andra utnyttjar list-comprehenssion

Funktioner kan returenra ett eller flera objekt som placeras i tuple eller inte retunerar något. Ex denna som ber användare ange och sen printar med formatering, dock ingen return

Text

Description automatically generated

Om man vill ha flera kommer de svaras i tuple

A screenshot of a computer

Description automatically generated with medium confidence

Funktioner – lokal och global och lokal variabler.

När man går utanför försviner de lokala, makuleras.

Graphical user interface

Description automatically generated with medium confidence

Funktionerna kan alltså andå använda blobala varibler, notera hur z kan användas i funktionen även om det defineras efter funktionen definieras.

Varibler med samma innehåll som skiljer sig i global och funktionen kommer ha olika id. Vill man ha global i en funktion kan man använda **global**

**Text

Description automatically generated**

Funktioner anrop med namn

Ex funktioin som omvandlar bråk till flyt tall och skriver ute dem med tre decimaler

Graphical user interface

Description automatically generated with low confidence

Finns också möjlighet att ge sina funktioner default värden då måste man inte fylla i alla värden när man anropar funktionen skriver ”=” i definitoinen

Text

Description automatically generated

Exempel på när detta är relevant

Funktion har två inkommande elemt där ett av dem är lista. Funktionen kommer lägga till två listor till den angivna listan. I dett fall 4-7. Sen kommer man säga att a är en sifra och Listan är 1,2,3 som lista

Text

Description automatically generatedDetta returernar alltså en dubbel lista för M och 5 för x. Alltås att L kommer påverkas av append men L som skapas i funktionen kommer vara lokal endast där

I Python anropas funktioner med objektsreferens det vill säga att ufnktionens ingående parametrar är kopior av refreser på objekten som funktionen anropas med. Dessa kopior makuleras efter att funktionen är avslutad.

A screenshot of a computer

Description automatically generated with medium confidence

Med \*tecken så kan man göra en tuple med alla infkommande argument. Ex om man vill skaopa funktionen sum av alla inkommande argument

Text

Description automatically generatedEftersom arg är en tuple kan vi använda len för att se hur många som finns i

Rekursiva funktioner anropar sig själva exmepelvis faktulretet

Text

Description automatically generated

Alla data i python är objekt även funktioner och det gör att man kan stoppa in en funktion i en annan funktion.

A picture containing graphical user interface

Description automatically generated

f\_sum blir alltså en refrens till funktionen SUM

Lambda funktioner

Om en funktioner innehåller endast en returnstas kan ma nskapa en anonym funktion med ex lamdbda utryck. Det är ett förkortat sätt att göra det på. De heter anonyma funktioner eftersom de inte har namn utan SUM i nedan fall är bara peckere av lamda.

**Lambda** är nyckel ordet.

A picture containing text

Description automatically generated

La