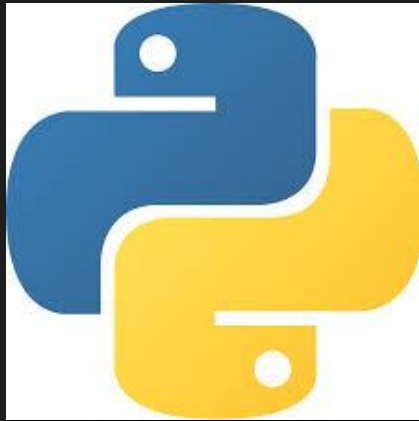


Programming med Python, FYD095

VT26, lp3

Python



Visual Studio Code



Vitali Zhaunerchyk

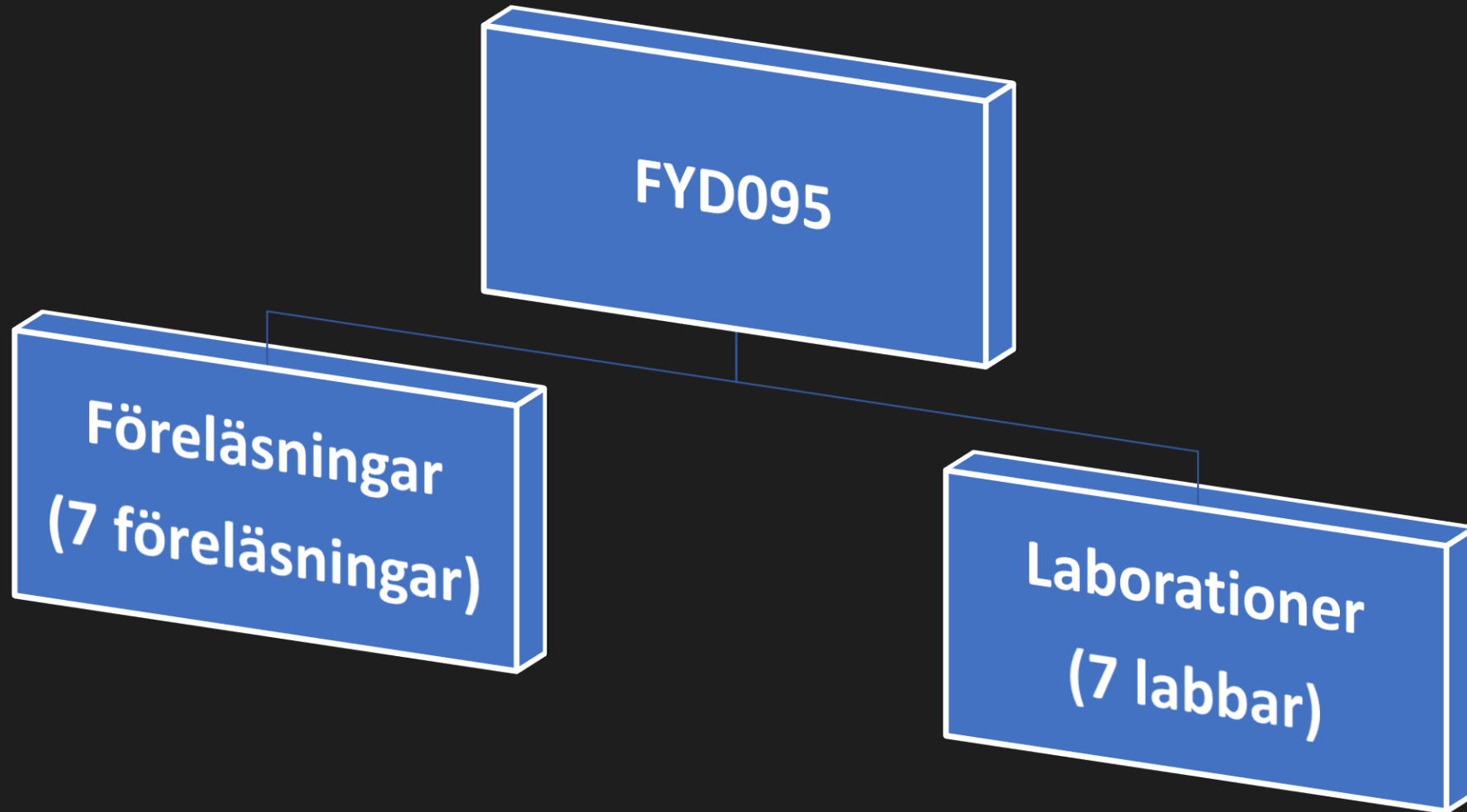
vitali.zhaunerchyk@physics.gu.se

F8115

Denna kurs handlar om

- Datatyper / databehållare
- Flödeskontroll
- Operatorer
- Funktioner
- Arbete med filer
- Numeriska fält från NumPy modulen
- Grafik
- Objektorienterad programmering

Kursstruktur



kursen hålls på Zoom (både föreläsningar och laborationer)

Betyg

Löpande tenta

G : 30 poäng för laborationer

VG : 45 poäng för laborationer

Kurslitteratur



Python från början av Jan Skansholm, Studentlitteratur

Föreläsningar

Online föreläsningar med Zoom
Föreläsare: Vitali Zhaunerchyk

Föreläsning 1: Måndag 19/1 (18:00 - 20:00)

Föreläsning 2: Torsdag 22/1 (18:00 - 20:00)

Föreläsning 3: Torsdag 29/1 (18:00 - 20:00)

Föreläsning 4: Måndag 2/2 (18:00 - 20:00)

Föreläsning 5: Torsdag 5/2 (18:00 - 20:00)

Föreläsning 6: Måndag 16/2 (18:00 - 20:00)

Föreläsning 7: Torsdag 19/2 (18:00 - 20:00)

Laborationer och labbkompendium

- Labbkompendiet består av fyra kapitel: 1) "*Grundläggande Python, del 1*" 2) "*Grundläggande Python, del 2*", 3) "*NumPy, SciPy, och matplotlib modulerna*", och 4) "*Objektorienterad programmering*".
- Varje kapitel består av delkapitel och varje delkapitel innehåller tre uppgifter
- Det är obligatoriskt att redovisa åtminstone en uppgift från varje delkapitel
- För varje redovisning får en student ett särskilt antal poäng som beror på uppgiftens svårighetsgrad
- Studenter arbetar och redovisar uppgifter individuellt
- För att bli godkänd på en uppgift, ska studenten 1) ha ett felfritt skript, 2) kunna förklara varje rad i skriptet, och 3) svara på frågor från labbassistenten
- Vi har flera labbtillfällen. Tanken är att labbtillfällena används för redovisning, och man arbetar på uppgifter individuellt hemma
- Under labbtillfället har en student redovisningspass på 15 minuter, vilket innebär ett individuellt Zoom-möte med en labbassistent

Laborationer och labbkompendium

- Om en student är ordentligt förberedd, brukar studenten hinna redovisa 4-7 uppgifter under ett 15-minuters redovisningspass
- Utöver 15-minuters redovisningspass, finns det möjlighet att få hjälp från kursansvarig
- Under ett redovisningspass kan studenten redovisa uppgifter från hela labbkompendiet
- Labbtillfällen är planerade på kvällar (kl. 18-21) och dagar (kl. 13-16)
- Studenter kommer att delas upp i 24 grupper så att grupp 1-12 redovisar under kvällar och grupp 13-24 på dagar
- Studenter ska registreras för varje labbtillfälle på Canvas under "Personer" i fickan "Lab" genom att välja en grupp. Redovisningstider för olika grupper kan hittas på Canvas. OBS: tilldelad grupp kan ändras från den valde gruppen
- Kursansvarig kommer att skicka ett meddelande genom Canvas i god tid när registrering öppnas och stängs. OBS: det är inte möjligt att registrera sig efter att registreringen har stängts
- Man får inte använda AI-verktyg (t. ex. ChatGPT) för att lösa uppgifter. Om man gör det kommer det att betraktas som fusk

Laborationer och labbkompendium

Online laborationer med Zoom

Labbandledare: Vitali Zhaunerchyk, Roman Khymyn,...

Tillfälle 1 (v6): Tisdag 3/2 (18:00 - 21:00) eller Onsdag 4/2 (13:00 - 16:00)

Tillfälle 2 (v8): Tisdag 17/2 (18:00 - 21:00) eller Onsdag 18/2 (13:00 - 16:00)

Tillfälle 3 (v9): Tisdag 24/2 (18:00 - 21:00) eller Onsdag 25/2 (13:00 - 16:00)

Tillfälle 4 (v10): Tisdag 3/3 (18:00 - 21:00) eller Onsdag 4/3 (13:00 - 16:00)

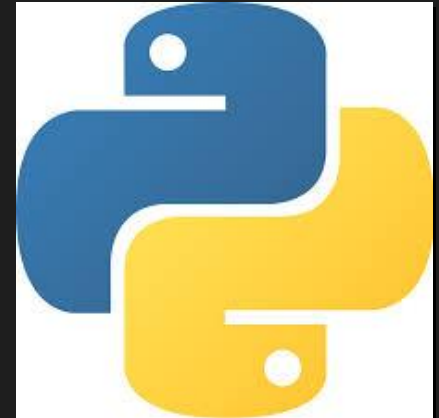
Tillfälle 5 (v11): Måndag 9/3 (13:00 - 16:00) eller Tisdag 10/3 (18:00 - 21:00)

Tillfälle 6 (v12): Måndag 16/3 (13:00 - 16:00) eller Tisdag 17/3 (18:00 - 21:00)

Tillfälle 7 (v13): Måndag 23/3 (13:00 - 16:00) eller Tisdag 24/3 (18:00 - 21:00)

Pythons egenskaper

- Python är ett interpreterat programmeringsspråk
- Python är ett högnivåprogrammeringsspråk
- Det stöder både procedurell och objektorienterad programmering
- Det är ett programmeringsspråk med dynamisk typning
- Det kan köras i två lägen: skript och interaktivt
- Det har en stor mängd bibliotek som man kan använda för vetenskapliga tillämpningar
- Nu är det ett av de mest populära programmeringsspråken i världen
- Det är gratis



Kompilerade programmeringsspråk (C, C++, ...)

```

content: "";
content: none;

table {
  border-collapse: collapse;
  border-spacing: 0;

  button, input, select, textarea { margin: 0;
    font: outline 0;
  }
  a {<del>link</del>-tap-highlight-color: #FF5E99 }
  img, video, object, embed {
    max-width: 100%;
    height: auto!important;
  }

  iframe {max-width: 100%;
    blockquote {
      font-style: italic;
      font-weight: normal;
      font-family: Georgia, Serif;
      font-size: 15px;
      padding: 0 10px 20px 2px;
      position: relative;
      margin-top: 25px;

      blockquote:after {
        position: absolute;
        content: "
      }
    }
  }
}

blockquote p {margin-bottom: 10px;
  strong, b {font-weight: bold }
  em, i, cite {
    font-style: normal;
    font-family: monospace;
  }
  small {font-size: 100%;
    figure {margin: 10px 0;
      code, pre {
        font-family: monospace, consolas, sans-serif;
        font-weight: normal;
        font-style: normal;
      }
    }
  }
  pre {
    margin: 5px 0 20px 0;
    line-height: 1.3em;
    padding: 5px 10px;
    overflow: auto;
  }
  img {height: 15px;
    width: 8px;
  }
}

code {font-size: 10px;
  pre {font-size: 10px;
    block
  }
}

```

Man använder texteditor eller IDE (integrerad utvecklingsmiljö) för att skriva programtext / källkod

```
// Created by Myclipse on 7/28/2013.
// Copyright © 2013 Myclipse. All rights reserved.

#import "ViewController.h"

#import <UIKit/UIKit.h>
#import <MobileServices/MobileServices.h>

@interface ViewController ()

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    [self setupSearchLightSearch];
}

- (void)setupSearchLightSearch {
    NSArray *attributes = [NSArray arrayWithObjects:@"id", @"initializedName", @"name", @"password", @"role", @"username", @"password", nil];
    NSString *url = [NSString stringWithFormat:@"http://localhost:8080/Service/FindUser?%s", [NSString stringWithFormat:@"id=%d", [NSNumber numberWithInt:1]]];
    NSData *responseData = [NSData dataWithContentsOfURL:[NSURL URLWithString:url]];
    NSDictionary *dict = [NSDictionary dictionaryWithObject:[NSJSONSerialization JSONObjectWithData:responseData options:0 error:nil] forKey:@"data"];
    NSArray *items = [NSArray arrayWithObjects:[NSDictionary dictionaryWithObject:[dict[@"data"][@"id"]] forKey:@"id"], [NSDictionary dictionaryWithObject:[dict[@"data"][@"initializedName"]] forKey:@"initializedName"], [NSDictionary dictionaryWithObject:[dict[@"data"][@"name"]] forKey:@"name"], [NSDictionary dictionaryWithObject:[dict[@"data"][@"password"]] forKey:@"password"], [NSDictionary dictionaryWithObject:[dict[@"data"][@"username"]] forKey:@"username"], [NSDictionary dictionaryWithObject:[dict[@"data"][@"password"]] forKey:@"password"], nil];
    NSArray *indexPaths = [NSArray arrayWithObjects:[NSIndexPath indexPathForRow:0 inSection:0], [NSIndexPath indexPathForRow:1 inSection:0], [NSIndexPath indexPathForRow:2 inSection:0], [NSIndexPath indexPathForRow:3 inSection:0], [NSIndexPath indexPathForRow:4 inSection:0], [NSIndexPath indexPathForRow:5 inSection:0], nil];
    [self.tableView reloadData];
}

- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
}
```

programtext / källkod

kompiator

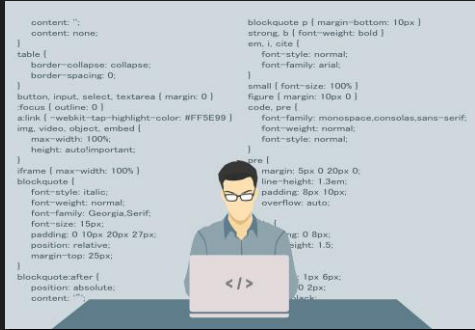
Kompilator är ett program som översätter källkoden till en maskinkod. Maskinkoden består av nollor och ettor.

Körbar fil (exe-fil): output

Länkare är ett program som skapar en exekverbar fil. Den filen kan köras på operativsystem (OS)

länkare

Interpreterade språk (Python, Matlab, ...)



Man använder texteditor eller IDE (integrerad utvecklingsmiljö) för att skriva ett skript



```
bicycles.py  x  motorcycles.py  x  hrs_worked.py
1 hrs = int(input("Enter Hours:"))
2 rph = float(input("Enter Rate per Hour:"))
3
4 if hrs <= 40 :
5     total_pay = hrs * rph
6     print(total_pay)
7 else :
8     ot_pay = ((hrs - 40) * (1.5 * rph))
9     base_pay = 40 * rph
10    total_pay = base_pay + ot_pay
11    print(total_pay)
```

Skript

interpretator



output

En interpretator / tolk är ett program som utför kommandon enligt skriptet, rad efter rad

En programtext/källkod för interpreterade programmeringsspråk heter ett skript. Därför kallas interpreterade programmeringsspråk skriptspråk istället för programmeringsspråk.

Kompilerade vs interpreterade programmeringsspråk

Kompilerade programmeringsspråk

Interpreterade programmeringsspråk

Exekverbar fil körs utan kompilator

Interpreterator måste installeras på en dator

Körbarhet av exekverbar fil beror på OS

Hög kompatibilitet, beror inte på OS

Kompilering och länkning tar tid

Kan köras omedelbart

Hög effektivitet (snabb exekvering)

Långsammare hastighet vid exekvering

Ganska ofta lägre nivå programmering

Högnivåprogrammering

Fel hanteras efter kompilering

Fel kan hanteras rad efter rad i ett skript

Python installation

Översikt och jämförelse av olika Python IDE och keditorer

<https://www.programiz.com/python-programming/ide>

- Det rekommenderas att använda Visual Studio Code
- Python på denna IDE (VS Code) kan köras både i skript- och interaktiv mod. Den interaktiva moden involverar *REPL (read-eval-print loop)*.
- Instruktion för installering av VS Code kan hittas i kursboken eller här <https://code.visualstudio.com/docs/languages/python>
https://www.youtube.com/watch?v=7EXd4_ttluw

Visual Studio Code



Mitt första skript

```
# Mitt första program
```

```
a = 3
```

```
b = 5
```

```
c = a * b
```

```
print(f'Multiplikation av {a} och {b} är {c}')
```

```
Multiplikation av 3 och 5 är 15
```

Multipl tilldelning

```
# Mitt första program
```

```
a,b = 3,5
```

```
c = a * b
```

```
print(f'Multiplikation av {a} och {b} är {c}')
```

- 1) Variablerna skapas vid initialiseringen (tilldelningssatserna)
- 2) I Python behöver man inte deklarerat datatyp. Variabler får datatyp beroende på vad man tilldelar dem (dynamisk typning).
- 3) Variabelnamn får bara innehålla bokstäver, siffror och understreck `_`. Namnet får inte börja med en siffra.
- 4) Variabelnamn är skiftlägeskänsliga, dvs. `a` och `A` är två olika variabler.

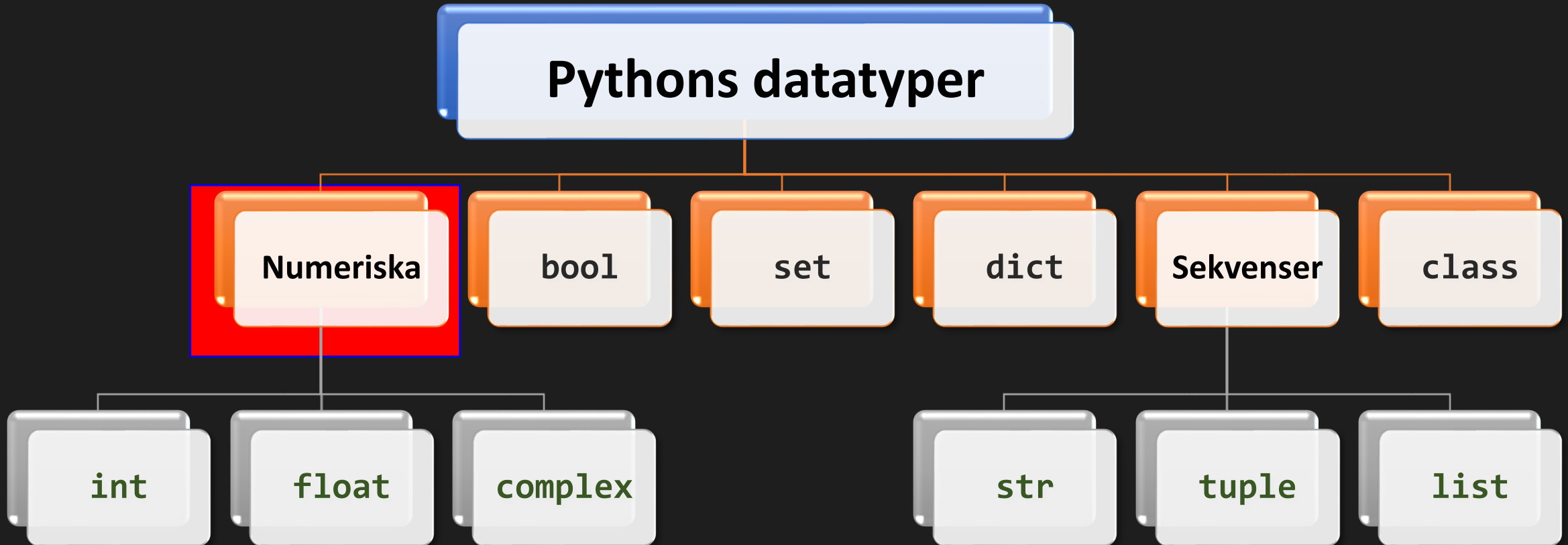
`print(text1, text2, text3, ...)` anropas för att utskriva texten som ges i argument

`print(text1, text2, text3, ..., end='')` skriver ut nästa text på samma rad

`print(f'...')` – f sträng, formaterad sträng. I f-strängar placeras objekt som man vill skriva ut i klammerparanteser/måsvingar `{}`

Pythons datatyper

Variabler (mer precis objekt) i Python är databehållare. Varje variabel(objekt) tillhör en specifik datatyp, som bestämmer värde och beteende hos variabeln.



Mitt första skript

```
# Mitt första program

a = input('Ange ett tal : ')
b = input('Ange ett annat tal : ')

a = int(a)
b = int(b)

c = a * b

print(f'Multiplikation av {a} och {b} är {c}')
```

s = input(text)

text skrivs ut

s av typ **str** innehåller svaret från användaren

int(arg) omvandlar (kastar) argument **arg** till **int** datatyp

```
Ange ett tal : 3
Ange ett annat tal : 5
Multiplikation av 3 och 5 är 15
```

```
# Mitt första program

a = int(input('Ange ett tal : '))
b = int(input('Ange ett annat tal : '))

c = a * b

print(f'Multiplikation av {a} och {b} är {c}')
```


Formatering, f-sträng

I f-strängar anges en utskrivningsformat i klammerparenteser efter objektsnamn följt av kolon.

```
# Mitt första program
```

```
a = input('Ange ett tal : ')\nb = input('Ange ett annat tal : ')
```

```
a = int(a)\nb = int(b)
```

```
c = a * b
```

```
print(f'Multiplikation av {a:5} och {b:7} är {c:05}')
```

```
a = 1_000_000_000
```

```
print(f'Värdet är {a:,}')
```

```
Värdet är 1,000,000,000
```

```
Ange ett tal : 3\nAnge ett annat tal : 5\nMultiplikation av      3 och      5 är 00015
```

Fältet består
av 5 tecken

Fältet består
av 7 tecken

Fältet består av 5
tecken och tomma
tecken ersätts med 0

```
Ange ett tal : 2.56\nAnge ett annat tal : 3.45\nTraceback (most recent call last):\n  File "c:\\Python\\program1.py", line 6, in <module>\n    a = int(a)\nValueError: invalid literal for int() with base 10: '2.56'
```

Formatering, f-sträng för flyttal

{obj:X.Yf}

X – längd på fält

Y – antal siffror efter decimalpunkten

Om **e** används istället för **f**, skrivs det flyttalet i exponentiell form

```
# Mitt första program

a = input('Ange ett tal : ')
b = input('Ange ett annat tal : ')

a = float(a)
b = float(b)

c = a * b

print(f'Multiplikation av {a:5.2f} och {b:07.3f} är {c:f} eller {c:e}')
```

Ange ett tal : 1.367

Ange ett annat tal : 6.43

Multiplikation av 1.37 och 006.430 är 8.789810 eller 8.789810e+00

Numeriska objekt. Aritmetiska operatorer

Operator	Beskrivning
**	potens
*	multiplikation
/	division (returnerar flyttal)
//	division avhuggen till heltal
%	rest vid division
+	addition
-	subtraktion

```
>>> 4**3
64
>>> 9/4
2.25
>>> 9//4
2
>>> 9%2
1
```