Mappeoppgave 2

Informasjon om oppgaven

Når du besvarer oppgaven, husk:

- les oppgaveteksten nøye
- kommenter koden din
- sett navn på akser og lignende i figurene
- skriv hvor du har hentet kodesnutter fra, hvis du gjør det
- bruk engelske variabelnavn og vær konsistent med hvordan du bruker store og små bokstaver
- bruk mest mulig funksjoner for ting som kan repeteres
- En kort kode kan være en bra kode, så ikke gjør det mer komplisert enn det spørres om.

Du kan få full pott uten å svare på oppgaven som er markert "ekstrapoeng". Du blir likevel belønnet for denne (dvs. hvis du har noen feil og får 45 poeng totalt, så kan du få en høyere poengsum hvis du også har svart på "ekstrapoeng".

Innlevering av oppgavene

Du skal levere begge mappene samtidig (det vil si denne oppgaven og mappe 1). Innleveringsfristen er 6 desember kl 13:00. Begge oppgavene skal leveres i github (som jupyter-fil) og wiseflow (som PDF). Bruk navnet "SOK-1003-eksamen-2022-mappe2" på filene.

- For github: Husk å gi meg (brukernavn "okaars") tilgang til github-reposetoriet deres. Hvis dere har satt reposetoriet til public (anbefales ikke), må dere dele lenken til dette på ole.k.aars@uit.no
- For wiseflow: En person fra hver gruppe (for hver mappeoppgave), leverer inn. Ved innlevering kan du krysse av hvem som er på gruppen din

Se generell informasjon om hvordan man leverer oppgaven her.

NB!: En person fra gruppa må fylle ut dette skjemaet for å melde om hvem som er på gruppa. Dere vil i etterkant motta en epost om tidspunkt for presentasjon.

Presentasjon

Presentasjonen innebærer en kort gjennomgang av oppgaven (10-15 min) etterfulgt av kommentarer fra meg (10-15 min). Alle gruppemedlemmer skal bidra til presentasjonen. Det er anbefalt å laste opp besvarelsen på github forut for presentasjonen (helst to dager før) slik at jeg har mulighet til å lese gjennom. Dere vil ha mulighet til å endre besvarelsen etter presentasjonen, frem til endelig innlevering 6 desember.

Oppgave 1 (10 poeng)

a) Vi skal spille et spill der vi kaster en terning 6 ganger. Lag en funksjon med "for-løkke" som printer alle terningene som har blitt kastet. Du kan bruke `np.random.randint()` til å lage tilfeldige tall

In [8]:
```
import numpy as np

dice = []
for i in range(6):
        dice = np.random.randint(1,7)
        print(dice)
```

```
1
6
4
6
3
6
```

b) Juster den samme funksjonen slik at den lagrer tallene i en liste før den printer ut selve listen. Dere kan kalle denne listen for `lot_numbers`. Dere kan vurdere å bruke `append()` som del av funksjonen.

In [14]:
```
dice = []
for i in range(6):
        lot_numbers = np.random.randint(1, 7)
        dice.append(lot_numbers)
        print(dice)
```

```
[6]
[6, 6]
[6, 6, 1]
[6, 6, 1, 3]
[6, 6, 1, 3, 6]
[6, 6, 1, 3, 6, 3]
```

In [ ]:
```
print(dice)
```

```
[3, 2, 1, 3, 3, 1]
```

c) Juster den samme funksjonen slik at den har to argument. Disse argumentene er to terningverdier som du "tipper" blir kastet. Bruk `if`, `else` og `elif` til å generere vinnertall. Resultatet fra funksjonen skal printe ut ulike setninger avhengig av om man får 0, 1 eller 2 rette. Setningene velger du selv, men de skal inneholde tallene som du tippet, og tallene som ble trukket.

In [2]:
```
import numpy as np
import random
number1 = int(input("Choose your first number: "))
number2 = int(input("Choose your second number: "))
count = 0
dice = []
print("The dice fell on:")
for j in range(6):
    rng_num = random.randint(1, 6)
    dice.append(rng_num)
    print(rng_num)


for i in dice:
    if i == number1:
        count += 1

    elif i == number2:
        count += 1
```

```
if count >= 2:
    print("Congrats! You got ", int(count), " correct.")
elif count == 1:
    print("You got ", int(count), " right!")
else:
    print("You got nothing right 😕")
```

```
The dice fell on:
6
6
4
6
4
1
Congrats! You got  2  correct.
```

Oppgave 2 (10 poeng)

a) Du har nå begynt å spille lotto i stedet, og satser alt på ett vinnertall. Lag en while-løkke som printer ut tall helt til du har trukket riktig tall (som du definerer selv). For enkelthets skyld kan du begrense utfallsrommet av trekningene til mellom 0-30.

In [5]:
```
from IPython.display import Markdown, display
import sympy as sp
import numpy as np

lottery = []
x,y=0,30

while y<40:
    y+=x
    x=5*(np.random.rand()-0.5)
    lottery.append(y)

print(lottery)
```

```
[30, 31.565291975889235, 30.84149803942209, 30.902515110288768, 31.554148391734728, 34.04886751101526, 32.40115360028053, 30.323021825623364, 30.903886656490183,
31.71597091240516, 29.60656506900056, 28.59160636708077, 30.611511574568972, 29.260515758172584, 30.193241517085347, 31.29142279999774, 31.40691930131396, 31.69
586550683971, 29.86094542790937, 30.16787502580633, 31.484965794743061, 32.581149600764625, 31.082135825194214, 31.831409803557758, 34.24234424460253, 34.9573648
05138624, 36.30570227431513, 35.03186540049513, 37.27449970262256, 35.4727050385719276, 36.938918889829125, 37.496293547177004, 38.2602140379108, 36.82532709205
132, 35.14425275020211, 32.88725363416918, 30.684666032985717, 30.76157253680258, 29.102318968369318, 29.71050053407313, 29.288497910044107, 31.68894021519978
6, 33.23117472002114, 31.243290355802685, 30.228142273719197, 28.11356715922329, 30.200558563965984, 27.81922647122802, 30.274506743724547, 30.623355343766367,
33.014440549751754, 31.22177792149757, 32.274155474083415, 30.457958301503517, 28.312462325861294, 30.383314231397947, 32.371360900836866, 33.56154234708263, 3
2.43494600208048, 30.073030059975643, 30.104978315031794, 32.034970697134845, 31.887340695091684, 32.392473537923074, 30.066654505469728, 31.189218763861184, 3
1.60589521294788, 29.68294695418019, 29.40632429446037, 29.948560479349563, 28.009722031593267, 29.35162689663841, 27.269155960237242, 26.751009498119302, 28.9
37615234863205, 28.721617466343023, 30.62585776151309, 31.988685735082015, 32.64172853967624, 33.28495810232921, 31.78547585930525, 29.619099525057045, 28.1113
1452591179, 27.647307984903723, 29.32391458256089, 31.514815542645024, 33.09479120316986, 34.761994240252946, 35.075591633387994, 33.27334343374897, 34.3703540
90733464, 34.568247640222416, 33.46888237547322, 35.26327555902253, 33.56946812102964, 31.22255725710957, 31.505479377290560, 30.497440742048983, 28.9060728532376
46, 30.47638874836996, 28.280820412593766, 30.60338485119342, 29.088140390458445, 30.276155578500666, 29.217555131515333, 31.527224144258696, 30.420972984814725,
32.42110001238058, 33.9889207488512285, 35.67816471840948, 33.68415929544326, 32.59209043998203, 34.701244533789652, 32.42472148542542, 33.015900208741904, 30.99
9641998315383, 31.902619941436511, 33.43630550069064, 32.30669425183795, 32.36576725388051, 30.613234744310722, 28.651938672348666, 28.958224073832696, 26.845765148
325103, 24.957626064214022, 26.704681689192178, 28.526978972346686, 27.793721495430642, 25.547454429900228, 24.78916833382984, 24.06226537274729, 23.3250260707
1251, 20.976065142878557, 20.546167753870015, 18.051540152000676, 18.018832336032997, 18.436884445494687, 20.613457993093625, 20.984949837979974, 23.1895703287348
35, 20.84024552558045, 20.124361870341723, 19.146286821571742, 21.645024744275737, 22.33488923671248, 21.639540752589475, 23.774692680435507, 25.99797355375210
7, 26.3157116367161113, 24.752122706392377, 22.303242097810198, 21.862501622608136, 20.445059765122892, 20.3199761388266608, 22.650349035895374, 20.769364658391
1, 22.725374176067223, 23.963428820033815, 24.748017518205777, 26.4600218688594, 24.590972621460888, 24.503208817615413, 23.360218631635558, 23.698746045841467,
23.876027155713793, 21.857731547532037, 23.75824351813825, 22.950958444108903, 25.349953986763353, 23.293174685749822, 23.041790449635403, 25.018781306927433,
23.816025464454984, 23.93582377362829, 23.902792202282576, 25.465696887941153, 24.32413913575746, 22.806086762110006, 24.662176793005463, 24.68041621356782, 27.
1039332982234O3, 27.5030695831673460, 25.585037606357926, 26.932148130087036, 28.887010485774276, 29.6535257458269 7, 27.953819765753 71, 27.752435611011729, 27.33
1264113743515, 24.863631783854323, 25.643212914120 23, 25.441260221824283, 24.154948087450048, 21.778931624067 55, 23.0745934194685876, 21.855611070758933, 23.425
47595678248, 25.036396221952813, 23.706840501232634, 23.278481820029 39, 22.493089724579516, 21.881465866211578, 21.076353232251694, 20.82565289059372, 20.75614
7799848723, 19.950318386216683, 17.747032863060955, 18.216906195697614, 17.334282260730312, 19.098286534103 74, 18.839786558153996, 17.691001574924822, 17.25222
463114532, 14.795508681492098, 15.812507567251963, 14.233802640891163, 12.810090291430164, 14.053693005896 76, 16.544616410394 36, 17.745317535320 17, 17.10126977
5678805, 18.301883266600093, 16.752305478336 45, 15.170148205537886, 16.795849425192 76, 15.922547584676277, 15.398290801904032, 17.402810537585 21, 16.4794056913
8629, 16.753160790352364, 16.837828660249926, 18.192112413297146, 17.040980377325 77, 15.070023590358987, 13.221222992258218, 14.022190937746945, 15.402341991147
7477, 17.795382710194488, 16.672302731264796, 17.700777962149665, 18.2553996635781, 19.548311377943776, 17.093584445164545, 16.889312463814 47, 19.1184487438084
1, 18.440708168808452, 20.191538217395 24, 17.949764138057 24, 17.782298715633065, 17.810872125533812, 15.601677769405045, 16.545862261637804, 15.309221671652 70
5, 13.132204263866395, 13.382169374655449, 11.662022590590832, 14.09737147447564, 14.310953877657706, 13.814446315695426, 12.820555709743928, 11.57650429574349
5, 13.853188994169514, 14.876271640966383, 16.01467693514286, 17.689720897224365, 17.851445237750802, 16.006126002674307, 18.301444351224 27, 16.321835445692231
5, 16.083349602733 47, 14.100329333077715, 16.592857005074826, 15.636189651754801, 14.880065313225 74, 13.443844666656892, 13.185489258625182, 11.961358508141 88
1, 13.704329382049547, 14.005594818580184, 11.758093207490614, 11.018643334731925, 13.041055939957811, 12.089319259360842, 11.499189888205182, 11.364817341804 5
32, 9.775233084682585, 11.251894042059002, 8.794325371349 76, 9.982158543766895, 12.043456920281855, 10.847413729928926, 12.103635233536231, 14.501892166918834,
16.34031901899972, 17.853213561365 43, 19.088396297753 73, 19.898353719891794, 21.137751012657 23, 22.041317404286076, 24.41245474542475, 23.806212511662743, 22.3
4070476640038, 21.557151696326017, 22.648960605639 31, 22.174915898940583, 22.326670169549008, 20.714313530975 04, 21.274884693903786, 19.578427198243894, 17.782
47948005838, 18.077534932892437, 18.159384249559988, 16.603721386014477, 18.491048103913577, 20.188931009117454, 21.359018535551 34, 20.319586811774432, 19.9599
67050906994, 20.366964314467776, 18.123157919579572, 19.877816052101608, 21.554530286821 63, 19.144946700995995, 19.303559086973 44, 18.799038389065956, 17.66830
0858919032, 16.29077600554986, 15.419432162693354, 14.010436009728 97, 16.141539877467437, 14.127042720885775, 14.212741827349527, 16.058827502893916, 17.748458
428401833, 19.337532392760973, 20.638233696084594, 22.797758513431017, 22.3246799852665, 20.792641506729833, 22.396103182841742, 22.418245875046242, 21.7248756
84614815, 21.396218465106095, 21.497908373496443, 19.47956006280151, 19.82371269736109, 20.878892546651258, 20.243937457823368, 19.85106459081766, 17.812803444
59682, 17.417629374725 63, 16.193747775304658, 18.022481495090 36, 19.851020495179583, 21.044116624235528, 19.015144495524385, 18.529541183958727, 20.38500177853
0132, 20.391005311108394, 22.660296779475 67, 20.618054165680526, 19.078910989016904, 21.234618021703 99, 21.768157896289 13, 20.346995420575 71, 21.2211564849857,
21.770032580486564, 19.644747841649227, 17.859682435885293, 17.383719403663974, 16.610750704018315, 16.554774095762 44, 15.209602973072442, 13.211584395929453,
13.640272796057335, 12.061926568625584, 10.283061546517892, 11.040263234566792, 11.119052366004 14, 11.224678831811154, 11.084110883838047, 12.391469470479608,
14.500699744517833, 13.273498390206536, 12.435179621907153, 12.075108991095547, 12.767283165564569, 12.608809680262954, 13.008601400920146, 14.925819602556755,
17.094513480740932, 19.521187808907 67, 18.589838777131252, 16.569016268339276, 17.967917794580 81, 17.456399010960 28, 19.601037789035 97, 21.216341986883 23, 22.5
0175501572225, 24.009576496321 64, 24.828516958410 12, 24.721250577724437, 24.703657570425 13, 27.185880026902 95, 26.638464165780466, 28.125676208849303, 26.46486
9475135703, 28.962471322443676, 27.001786739990 96, 26.984648341118024, 26.529780964493 58, 27.066373139248096, 27.031355005921494, 25.22792875889188, 26.8936976
61119866, 27.739545084287 82, 27.394798397867362, 26.402455773400 54, 25.151499819336962, 25.684228344981 17, 25.650066248333417, 25.644160072152154, 27.74624656
5179874, 29.722771850891373, 30.185555770637848, 28.428225995939606, 25.941022145209 73, 25.076671410711 22, 27.386138712345762, 29.592811913477824, 27.1267474561
01164, 25.492376476699302, 27.507360808432715, 26.275257093329728, 28.699679201368404, 28.680188161907058, 28.138593423711065, 26.379142807969643, 27.818033016
240882, 27.725155881 40702, 29.431272384111438, 28.480759795082307, 28.768821769784367, 30.342929365812942, 30.029840078206 53, 29.896162964302 53, 28.777870547963
6, 29.764838912811303, 29.421822018920057, 29.352883024956 85, 27.387159392144184, 29.654559303664943, 29.974989254250 84, 29.802076915423562, 32.023591274462 39,
31.655652419725754, 32.815311468380 72, 32.845641639650815, 31.249297575188198, 33.124488555072645, 34.343928766873915, 33.314417133213 23, 35.146080104166091, 3
5.436530110386 54, 34.465279458066 71, 32.293233882191936, 33.917011809716016, 35.567323902884866, 33.862672460114936, 33.196389381867 23, 34.435475913605 39, 36.00
95159146766, 37.413973317512 88, 36.891236844184 5, 38.470012485600 97, 36.826666737033 46, 36.443818189267 58, 35.238017602530 69, 35.526263530975 12, 36.535792316270935, 34.887990440454054, 34.995080222946136, 37.175041342120146, 39.00110925601 31, 37.561609430436 46, 39.575552935229 65, 38.019873332760 92, 36.405336194273 19, 33.964422740384805, 36.179467597191 22, 35.925961393402 12, 35.259690358333 95, 34.282314037708 73, 33.140544849673304, 33.728401976162836, 33.049279359193 51, 31.310426665097 44, 30.102289057404043, 27.983199665441127, 30.223012475293956, 29.351593768016613, 28.985977635269606, 26.615889290853 39, 29.016170922643777, 26.755758631107923, 28.082869799668543, 30.064187444725512, 30.173251426891504, 30.815216440620528, 30.613151792406587, 30.092279338827 96, 28.040947133254512, 27.904464881782825, 25.509649100470 75, 27.431093011907787, 29.648396977413714, 32.107359266520 08, 29.608812482325902, 28.891209833717177, 27.138782543504256, 26.814917359673377, 28.025870913519 68, 27.201532617167803, 29.442117543779844, 31.285922661105445, 31.799634407641 89, 31.315724391011393, 30.321330501983 8, 31.142
89414082772, 30.585687240161107, 30.406268253490794, 31.829631075010422, 33.396641235743544, 32.201880231434465, 31.576172444091 12, 33.466866153412724, 34.0092
7070471271, 33.152004206634 59, 33.191140304586 26, 34.397308718000815, 33.120693574154 15, 33.784476365127 31, 35.140204110547 77, 37.214433756116016, 37.848110018
25047, 37.034628234442 97, 35.345868461585 47, 35.744605251012 75, 36.093513740936 25, 36.692285113773 41, 35.046325471530 93, 35.427139721781 295, 34.933366634014 5
2, 35.384863647341 51, 34.570141579176 1, 32.857678057877 18, 31.043004759321 096, 29.650812478564 69, 29.511104866535 955, 30.707532532653573, 28.494401899414974, 2
6.082675856494 39, 24.484674727314427, 23.623307849509196, 25.090298542863554, 26.528750453214037, 27.382996011523502, 25.443838772228105, 23.876914345794688, 2
5.766247663492212, 28.020391476944663, 25.537831936046313, 25.289983421152232, 25.052899450737634, 24.484490289428173, 22.458512155800342, 24.459202980982433,
26.890610573873314, 26.7746646584614, 25.190655381549536, 22.834481290098 47, 23.512399752506614, 23.175726215310895, 24.994408559385377, 22.948689847475 17, 21.
452654654954948, 20.581721822300942, 21.915792971114694, 23.697625505614575, 24.250945261629 97, 23.438220495969638, 22.242312816596545, 20.870547732911838, 18.
2, 18.227535144471 57, 17.588681706413308, 18.747192946882 74, 16.516214753806683, 15.172177148542609, 13.157587645058355, 14.068949928867625, 14.93
```

```
3141429457718, 15.939092857915925, 14.999906866312456, 15.534461498244232, 13.362212006881308, 15.048190365890928, 14.150982898895519, 15.577673528369322, 17.8
2345886668161, 19.789445660645605, 18.326743404722574, 20.65692594534633, 22.959460701708352, 20.679050059803927, 23.03728384716681, 24.59044579189973, 25.6350
8805482682, 27.39716527860987, 29.8337244354922, 28.760422007194094, 29.162102505676238, 28.188077565399652, 30.476651596688036, 29.590139677562597, 31.1119897
72633976, 33.5448828363574, 34.5703386295165, 36.48223580107493, 35.57591685219785, 35.54058610921674, 34.3124346720745 95, 35.60189636748629, 33.6917651636350
7, 32.162475253109, 32.23518886043245, 33.96967159524168, 36.4209077437751 46, 38.21106145901739, 36.87507525723849, 37.662646387007634, 37.39126676829635, 36.2
43134132001614, 36.65624606728467, 34.48725365974687, 36.175680162518105, 34.51264603005463, 32.94972778614235, 32.08918049905542, 30.97083873498897, 30.533532
835947867, 29.650396447857418, 30.289341707807115, 31.58885323106473, 31.308088336853366, 32.10134233636499, 31.365754638605928, 29.775801563277 94, 30.58779910
745787, 31.325505799747067, 30.572544207635346, 29.908077289680328, 27.556060393151 84, 26.608858344381066, 26.878591854024272, 26.109845683797868, 28.496444071
252377, 29.351617071029 94, 30.306967995957507, 30.449876021718143, 30.443722429321614, 31.422120905155332, 30.28633690596389, 32.474025170466 78, 32.038102031052
97, 31.737040868328034, 30.062620804247 88, 32.08672609071051, 32.10468194354844, 33.7158260569333, 33.73474937437241, 32.151125534985894, 32.31014797342107, 3
1.432307999043882, 32.62641955385507, 34.112333056104916, 32.82521945785381, 30.438747024503176, 31.947319521389 56, 33.761637287758134, 36.1148671776689, 33.87
3187237767894, 35.96445485025722, 36.78883993894468, 35.24793060643452, 34.33035461983297, 35.07813019316801, 33.661452864358665, 31.358471568905806, 30.525587
91483183, 31.566881026431226, 32.70174743622443, 31.250346762437385, 30.195494634026275, 31.969350566700513, 31.481522329279 29, 29.202975955441534, 29.71888643
158815, 27.617270104049076, 28.035852981471 84, 28.731931978754 27, 27.652101647988 49, 25.327610978223 12, 27.773841982973593, 29.053271024276466, 29.740378712114
076, 31.660244590087 99, 31.872946894214284, 29.644950247885 96, 29.917413507070826, 29.676289710139805, 27.932433513184 94, 29.035141294070005, 29.5660502960653
6, 28.716184294050 76, 26.421945899294393, 24.365366028878135, 25.851355394511 62, 24.999827109981947, 26.574552202063 99, 27.854986779274 61, 28.827972038587635,
29.990461486258933, 31.322271098510075, 29.953649975722314, 29.969432112737916, 27.759481603406 95, 28.666217476307672, 26.672656366124357, 25.585288589250077,
27.822024369399273, 26.084903822244 72, 27.835026334606 31, 25.699519949799075, 26.090051539742916, 24.641751463217897, 23.473152819912894, 23.339460143303956, 2
3.363585848599 74, 25.445804242506 85, 27.877758133467708, 28.452550847748 89, 27.058667844231 82, 27.170303243165 36, 28.939334121291367, 31.431097807866877, 29.46
2989799864847, 29.845580103332 53, 30.030690700757653, 28.390365985606213, 29.878445522162 97, 31.438199255804328, 29.298539193833335, 31.796559083989653, 31.817
95761208557, 29.662273689732 89, 29.486883354821174, 31.387085938987 95, 32.518440061015795, 32.000194678503796, 30.539379009022 97, 28.508828905921476, 30.718919
352089465, 29.197392058883306, 28.514330025105618, 30.147915597869 13, 32.334736313015796, 32.594461385736 48, 34.937510905349924, 37.084808537273 83, 37.9487882
93483446, 36.808490161988416, 35.762355898307746, 33.759470597412 33, 32.337881943660 01, 33.828554588869 18, 35.935878027912 4, 36.123206183279 95, 34.840543775377
16, 35.340536615236 07, 36.843556994923 81, 35.855704792132 33, 36.451944269886965, 37.100638270187616, 37.374669977623 16, 38.903234418680405, 36.970453565352194,
39.333839477032036, 37.621618857009906, 35.655804579117 56, 37.415040967100 99, 39.890193595069 99, 40.178717082883]
```

In [ ]:

b) Lag et plot av den while-løkken du nettopp lagde. Man blir belønnet om man;

- bruker `scatter`;
- lager plottet dynamisk (dvs at hver trekning vises hver for seg, og at x-aksen endrer seg etter en gitt verdi);
- viser hvor når siste trekningen blir gjort (dvs at den vises kun når du har trukket vinnertallet).

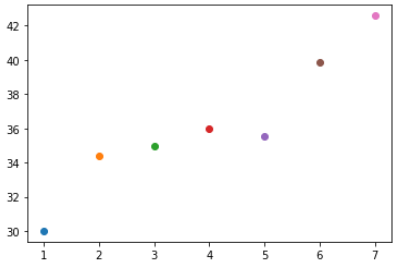
Avhengig av hvordan du lager figuren din kan du får bruk for å importere pakkene `Ellipse`, `display`, `clear_output`.

In [6]:
```
from matplotlib.patches import Ellipse
from IPython.display import display, clear_output
import matplotlib.pyplot as plt
from IPython.display import Markdown, display
import sympy as sp
import numpy as np
lottery = []
dy,y=0,30
x=0

while y<40:
    y+=dy
    dy=5*(np.random.rand()-0.1)
    x += 1
    lottery.append(y)

    print(lottery)
    plt.scatter(x,y)
```

```
[30]
[30, 34.40652704404119]
[30, 34.40652704404119, 34.976217147024784]
[30, 34.40652704404119, 34.976217147024784, 35.9876149555912]
[30, 34.40652704404119, 34.976217147024784, 35.9876149555912, 35.5227370507378]
[30, 34.40652704404119, 34.976217147024784, 35.9876149555912, 35.5227370507378, 39.843154758224124]
[30, 34.40652704404119, 34.976217147024784, 35.9876149555912, 35.5227370507378, 39.843154758224124, 42.586540087556195]
```



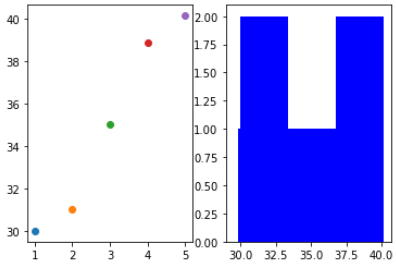
c) Ekstrapoeng: gjør det samme som i (b), men lag et histogram som vises ved siden av. Dette histogrammet skal vise hvor mange ganger de ulike tallene ble trekt. Bruk `plt.hist` til dette. Husk at du må definere figur og akseobjekt først.

In [7]:
```
lottery = []
dy,y=0,30
x=0

while y<40:
    y+=dy
    dy=5*(np.random.rand()-0.1)
    x += 1
    lottery.append(y)

    print(lottery)
    plt.subplot(1,2,1)
    plt.scatter(x,y)
    plt.subplot(1,2,2)
    plt.hist(lottery, bins= 3, color="blue")
```

```
[30]
[30, 31.045446361468695]
[30, 31.045446361468695, 35.01376293927185]
[30, 31.045446361468695, 35.01376293927185, 38.87748098878619]
[30, 31.045446361468695, 35.01376293927185, 38.87748098878619, 40.131567872019765]
```



Oppgave 3 (20 poeng)

En bedrift produserer biler. Produktfunksjonen til bedriften defineres slik $f(L, a, R) = 2RL^{a}$, hvor:

- `L` er arbeidskraft,
- `a` er produktiviteten til arbeiderne og
- `R` er antall robotmaskiner

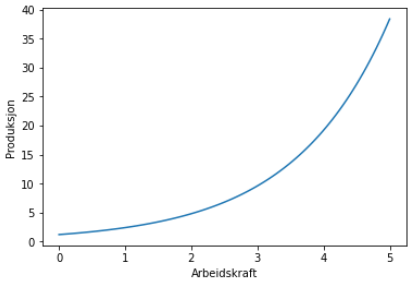
a) Lag en formel for produktfunksjonen til bedriften og plot den grafisk med ulike verdier av `L` på x-aksen. Anta `a=0.6` og `R=2`

In [8]:
```
import sympy as sp
def p(L,a,R):
    return 2*R*L**a
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0,5,100)

plt.plot(x,p(2,x,0.6))
plt.xlabel('Arbeidskraft')
plt.ylabel('Produksjon')
```

Out[8]:
```
Text(0, 0.5, 'Produksjon')
```



b) anta at profittfunksjonen til denne bedriften er ${profit = f(L, a, R)p-wL-cR-K}$, hvor

- `w` er månedslønnen til arbeiderne,
- `c` er kostnaden for robotmaskinene
- `K` er faste kostnader
- `p` er utsalgsprisen på bilene.

Anta `a=0.6` , `R=6` , `p=300 000` , `w=100 000` , `c= 1 000 000` og `K=90 000 000` . Plot profittfunksjonen figurativt for antall arbeidere (`L`) mellom 0 og 10 000. Vis profitten i millioner (dvs at du må dele på 1 000 000)

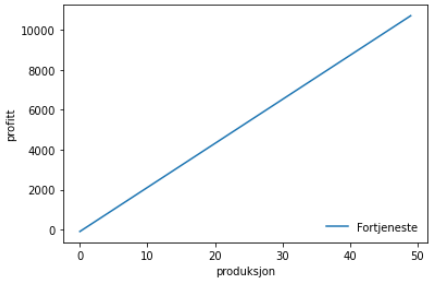
In [9]:
```
import matplotlib.pyplot as plt
import numpy as np

def profittfunksjon(L,a,w,p,c,K,R):
    return (L*a*R)*p-(w*L)-(c*R)-K

#creating the plot
x = np.linspace(0,10000)
fig,ax=plt.subplots()
ax.set_ylabel('profitt')
ax.set_xlabel('produksjon')


#plotting the function
plt.plot(profittfunksjon(x,0.6,6,300000,1000000,90000000,6)/1000000,label='Fortjeneste')
ax.legend(loc='lower right',frameon=False)
print(np.sum((profittfunksjon(89,0.6,6,300000,1000000,90000000,6)/1000000)))
```

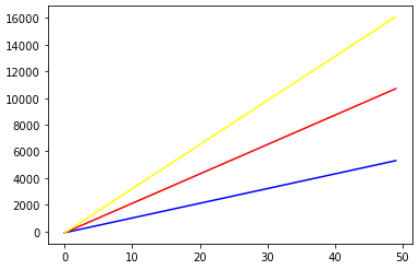
```
0.119466
```



c) Plot profitttfunksjonen for antall robostmaskiner `R=[3, 6, 9]` i samme plot (dvs at tre profittfunksjoner vises sammen). Bruk av "for loops" for å gjøre dette belønnes

In [116…
```
def profittfunksjon(L,a,w,p,c,K,R):
    return (L*a*R)*p-(w*L)-(c*R)-K
plt.plot(profittfunksjon(x,0.6,3,300000,1000000,90000000,3)/1000000,label='Fortjeneste', color = 'blue')
plt.plot(profittfunksjon(x,0.6,6,300000,1000000,90000000,6)/1000000,label='Fortjeneste', color = 'red')
plt.plot(profittfunksjon(x,0.6,9,300000,1000000,90000000,9)/1000000,label='Fortjeneste', color = 'yellow')
```

Out[116]:
```
[<matplotlib.lines.Line2D at 0x7f85cb20fa00>]
```



d) finn profittmaksimum og optimal antall arbeidere ved hjelp av derivasjon med samme forutsetninger som i (1b). Bruk `sympy` -pakken til dette

In [122…
```
#pakker du kan få bruk for
import sympy as sp
from sympy.solvers import solve

d_profitt=sp.diff(profittfunksjon(L,a,w,p,c,K,R))
d_profitt
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [122], in <cell line: 5>()
      2 import sympy as sp
      3 from sympy.solvers import solve
----> 5 d_profitt=sp.diff(profittfunksjon(L,a,w,p,c,K,R))
      6 d_profitt

NameError: name 'L' is not defined
```

e) vis figurativt med bruk av `fill_between` arealet hvor man taper penger (i rødt) og hvor man tjener penger (i grønt). Marker også profittmaksimum og antall arbeidere i profittmaksimum - gjerne ved bruk av `vlines` . Bruk ellers samme forutsetninger for argumentene som i oppgave (1b)

In [ ]:

f) Plot nå to figurer sammen der du viser hva optimal antall arbeidere gir i profitt (slik som i (2e)) og produksjon av antall biler (som du får fra produktfunksjonen). Marker optimum med vlines. Ha grafen med profittfunksjonen over grafen med produktfunksjonen. Du kan bruke `fig, (ax1, ax2) = plt.subplots(2)` når du skal gjøre dette.

Hint: Du kan finne antall biler som blir produsert ved å bruke antall arbeidere i profittmaksimum, i produktfunksjonen.

In [ ]:

Oppgave 4 (10 poeng)

I denne oppgaven skal vi hente ut et datasett fra eurostat på investeringer i hosholdningen. Bruk koden under til å hente ut dataene.
NB!: Husk at dere må ha innstallert pakken `eurostat` . Dette gjør dere med å åpne "Terminal" og kjøre `pip install eurostat` .

In [10]:
```
import eurostat
import pandas as pd

inv_data = eurostat.get_data_df('tec00098')
inv_data
```

Out[10]:

	freq	unit	sector	na_item	geo\TIME_PERIOD	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
0	A	PC	S14_S15	IRG_S14_S15	AT	8.44	8.60	8.47	8.58	8.30	8.36	8.29	8.69	8.81	9.03	9.21	9.99
1	A	PC	S14_S15	IRG_S14_S15	BE	9.82	9.45	9.37	9.03	9.45	9.31	9.33	9.28	9.35	9.78	9.15	9.94
2	A	PC	S14_S15	IRG_S14_S15	CH	6.83	6.48	6.29	6.31	6.20	6.26	6.13	6.07	6.06	5.69	5.42	NaN
3	A	PC	S14_S15	IRG_S14_S15	CY	13.72	10.73	8.74	7.53	7.15	6.69	8.02	8.97	11.30	12.98	13.12	13.26
4	A	PC	S14_S15	IRG_S14_S15	CZ	11.14	9.82	8.74	8.77	8.88	8.84	9.18	7.86	9.00	9.45	9.34	9.24
5	A	PC	S14_S15	IRG_S14_S15	DE	8.72	9.51	9.61	9.56	9.65	9.32	9.62	9.48	9.68	9.71	9.95	10.55
6	A	PC	S14_S15	IRG_S14_S15	DK	8.50	8.51	7.87	7.35	7.60	7.50	7.46	8.10	8.33	8.57	8.69	9.24
7	A	PC	S14_S15	IRG_S14_S15	EA19	9.30	9.21	8.80	8.37	8.24	8.09	8.35	8.52	8.71	8.77	8.53	9.51
8	A	PC	S14_S15	IRG_S14_S15	EE	6.14	6.47	6.84	7.52	7.79	7.94	8.52	8.99	8.68	9.26	9.94	10.04
9	A	PC	S14_S15	IRG_S14_S15	EL	9.12	8.54	6.28	4.90	3.16	2.72	2.75	2.69	2.44	2.59	2.97	3.40
10	A	PC	S14_S15	IRG_S14_S15	ES	9.66	7.85	6.48	4.84	4.72	4.57	4.60	5.17	5.42	5.54	5.34	6.68
11	A	PC	S14_S15	IRG_S14_S15	EU27_2020	9.07	8.95	8.56	8.12	8.06	7.97	8.22	8.42	8.54	8.60	8.33	9.19
12	A	PC	S14_S15	IRG_S14_S15	EU28	8.44	8.34	7.94	7.67	7.65	7.55	7.85	8.16	8.24	8.39	NaN	NaN
13	A	PC	S14_S15	IRG_S14_S15	FI	11.66	12.18	12.12	11.49	10.88	10.49	11.57	12.09	12.50	12.25	11.91	12.50
14	A	PC	S14_S15	IRG_S14_S15	FR	9.31	9.46	9.31	9.27	9.07	8.88	8.99	9.41	9.47	9.62	8.57	9.95
15	A	PC	S14_S15	IRG_S14_S15	HR	5.96	5.66	5.53	5.01	4.78	4.91	4.96	5.01	5.35	6.37	6.40	6.45
16	A	PC	S14_S15	IRG_S14_S15	HU	6.67	5.11	4.80	4.92	5.17	5.68	5.86	6.53	7.19	7.52	8.87	8.30
17	A	PC	S14_S15	IRG_S14_S15	IE	6.01	4.96	4.12	4.29	4.41	4.72	5.26	5.50	6.27	4.92	4.27	5.29
18	A	PC	S14_S15	IRG_S14_S15	IS	4.45	4.58	4.81	5.19	5.57	NaN	NaN	NaN	NaN	NaN	NaN	NaN
19	A	PC	S14_S15	IRG_S14_S15	IT	10.33	9.82	9.20	8.42	7.78	7.58	7.68	7.74	7.77	7.64	7.20	8.69
20	A	PC	S14_S15	IRG_S14_S15	LT	4.73	5.11	4.70	5.56	6.04	6.68	6.89	6.51	6.78	7.09	7.04	7.39
21	A	PC	S14_S15	IRG_S14_S15	LU	9.69	11.26	10.43	11.34	12.75	12.34	12.48	12.35	11.45	10.34	9.71	11.33
22	A	PC	S14_S15	IRG_S14_S15	LV	3.56	4.92	6.17	4.62	4.98	5.76	4.87	4.83	5.68	5.85	5.30	4.62
23	A	PC	S14_S15	IRG_S14_S15	NL	10.05	9.54	8.47	7.46	8.10	9.08	10.60	10.80	11.52	12.15	12.22	12.96

24	A	PC	S14_S15	IRG_S14_S15	NO	9.71	10.98	11.74	12.32	11.82	11.35	12.34	13.02	12.22	11.89	11.29	11.11
25	A	PC	S14_S15	IRG_S14_S15	PL	7.99	7.92	8.24	7.59	7.86	7.88	7.52	6.88	5.84	5.93	5.18	6.59
26	A	PC	S14_S15	IRG_S14_S15	PT	6.27	5.81	5.07	4.37	4.51	4.50	4.71	5.05	5.48	5.65	5.69	6.10
27	A	PC	S14_S15	IRG_S14_S15	RS	NaN	NaN	NaN	NaN	NaN	2.63	3.25	3.17	3.21	3.35	2.97	NaN
28	A	PC	S14_S15	IRG_S14_S15	SE	5.86	5.48	4.51	4.61	4.69	5.69	6.15	6.77	6.24	5.88	6.48	6.84
29	A	PC	S14_S15	IRG_S14_S15	SI	6.92	6.30	5.78	5.50	5.65	5.81	5.85	6.28	6.51	6.33	5.64	6.20
30	A	PC	S14_S15	IRG_S14_S15	SK	6.32	7.00	6.65	7.08	6.39	6.10	6.75	6.71	6.76	6.79	6.93	7.04
31	A	PC	S14_S15	IRG_S14_S15	UK	4.96	5.02	4.95	5.27	5.67	5.84	6.16	6.77	6.62	6.81	NaN	NaN

a) Bytt navn på kolonnen "geo\time" til "country" ved bruk av en av kodene under. Fjern så alle kolonner utenom "country" og alle årstallene.
NB!: Noen vil få en ekstra første kolonne som heter "freq" eller noe annet. Da må dere bruke versjon 2 av koden under.

In [18]:
```
#inv_data.columns = ['unit',   'sector',  'na_item', 'country'] + list(range(2010, 2022)) #v1
```

In [11]:
```
inv_data.columns = ['freq', 'unit',   'sector',  'na_item', 'country'] + list(range(2010, 2022))
inv_data = inv_data.drop(columns=['freq', 'unit', 'sector', 'na_item'])
inv_data.fillna(' ', inplace=True)
```

b) fjern radene med nan verdi. Sett deretter indeksen til "country".
Hint: En metode er å bruke `isna()` og `any()` over radaksene (dvs. `axis=1`)

In [12]:
```
inv_data=inv_data.set_index('country')
inv_data
```

Out[12]:

country	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
AT	8.44	8.6	8.47	8.58	8.3	8.36	8.29	8.69	8.81	9.03	9.21	9.99
BE	9.82	9.45	9.37	9.03	9.45	9.31	9.33	9.28	9.35	9.78	9.15	9.94
CH	6.83	6.48	6.29	6.31	6.2	6.26	6.13	6.07	6.06	5.69	5.42	
CY	13.72	10.73	8.74	7.53	7.15	6.69	8.02	8.97	11.3	12.98	13.12	13.26
CZ	11.14	9.82	8.74	8.77	8.88	8.84	9.18	7.86	9.0	9.45	9.34	9.24
DE	8.72	9.51	9.61	9.56	9.65	9.32	9.62	9.48	9.68	9.71	9.95	10.55
DK	8.5	8.51	7.87	7.35	7.6	7.5	7.46	8.1	8.33	8.57	8.69	9.24
EA19	9.3	9.21	8.8	8.37	8.24	8.09	8.35	8.52	8.71	8.77	8.53	9.51
EE	6.14	6.47	6.84	7.52	7.79	7.94	8.52	8.99	8.68	9.26	9.94	10.04
EL	9.12	8.54	6.28	4.9	3.16	2.72	2.75	2.69	2.44	2.59	2.97	3.4
ES	9.66	7.85	6.48	4.84	4.72	4.57	4.6	5.17	5.42	5.54	5.34	6.68
EU27_2020	9.07	8.95	8.56	8.12	8.06	7.97	8.22	8.42	8.54	8.6	8.33	9.19
EU28	8.44	8.34	7.94	7.67	7.65	7.55	7.85	8.16	8.24	8.39		
FI	11.66	12.18	12.12	11.49	10.88	10.49	11.57	12.09	12.5	12.25	11.91	12.5
FR	9.31	9.46	9.31	9.27	9.07	8.88	8.99	9.41	9.47	9.62	8.57	9.95
HR	5.96	5.66	5.53	5.01	4.78	4.91	4.96	5.01	5.35	6.37	6.4	6.45
HU	6.67	5.11	4.8	4.92	5.17	5.68	5.86	6.53	7.19	7.52	8.87	8.3
IE	6.01	4.96	4.12	4.29	4.41	4.72	5.26	5.5	6.27	4.92	4.27	5.29
IS	4.45	4.58	4.81	5.19	5.57							
IT	10.33	9.82	9.2	8.42	7.78	7.58	7.68	7.74	7.77	7.64	7.2	8.69
LT	4.73	5.11	4.7	5.56	6.04	6.68	6.89	6.51	6.78	7.09	7.04	7.39
LU	9.69	11.26	10.43	11.34	12.75	12.34	12.48	12.35	11.45	10.34	9.71	11.33
LV	3.56	4.92	6.17	4.62	4.98	5.76	4.87	4.83	5.68	5.85	5.3	4.62
NL	10.05	9.54	8.47	7.46	8.1	9.08	10.6	10.8	11.52	12.15	12.22	12.96
NO	9.71	10.98	11.74	12.32	11.82	11.35	12.34	13.02	12.22	11.89	11.29	11.11
PL	7.99	7.92	8.24	7.59	7.86	7.88	7.52	6.88	5.84	5.93	5.18	6.59
PT	6.27	5.81	5.07	4.37	4.51	4.5	4.71	5.05	5.48	5.65	5.69	6.1
RS						2.63	3.25	3.17	3.21	3.35	2.97	
SE	5.86	5.48	4.51	4.61	4.69	5.69	6.15	6.77	6.24	5.88	6.48	6.84
SI	6.92	6.3	5.78	5.5	5.65	5.81	5.85	6.28	6.51	6.33	5.64	6.2
SK	6.32	7.0	6.65	7.08	6.39	6.1	6.75	6.71	6.76	6.79	6.93	7.04
UK	4.96	5.02	4.95	5.27	5.67	5.84	6.16	6.77	6.62	6.81		

c) Lag et nytt datasett hvor du kun har med de nordiske landene (dvs. "NO", "SE", "DK", "FI"). Det kan være nyttig å bruke `isin` til dette. Bytt så om på kolonner og rader ved hjelp av `transpose`.

In [13]:
```
inv_data2 = eurostat.get_data_df('tec00098')
inv_data2.columns = ['freq', 'unit',   'sector',  'na_item', 'country'] + list(range(2010, 2022))
inv_data2 = inv_data2.drop(columns=['freq', 'unit', 'sector', 'na_item'])
inv_data2.fillna(' ', inplace=True)
inv_data2 = inv_data2.loc[inv_data2['country'].isin(['DE', 'FI' , 'NO', 'SE'])]
inv_data2=inv_data2.set_index('country')
inv_data2=inv_data2.transpose()

inv_data2
```

Out[13]:

country	DE	FI	NO	SE
2010	8.72	11.66	9.71	5.86
2011	9.51	12.18	10.98	5.48
2012	9.61	12.12	11.74	4.51
2013	9.56	11.49	12.32	4.61
2014	9.65	10.88	11.82	4.69

2015	9.32	10.49	11.35	5.69
2016	9.62	11.57	12.34	6.15
2017	9.48	12.09	13.02	6.77
2018	9.68	12.5	12.22	6.24
2019	9.71	12.25	11.89	5.88
2020	9.95	11.91	11.29	6.48
2021	10.55	12.5	11.11	6.84

d) Lag en ny kolonne som du kaller "mean". Denne skal være gjennomsnittet av alle de nordiske landene for hvert av årene (dvs at du må ta gjennomsnittet over radene). Plot så dette og kall y-aksen for "investering"

In [16]:
```
inv_data2_mean = inv_data2[['DE', 'FI', 'NO', 'SE']].mean()
```

In [17]:
```
import matplotlib.pyplot as plt

plt.plot(inv_data2_mean)
plt.grid(color = 'green', linestyle = '--', linewidth = 0.5)
font1 = {'family':'serif','color':'blue','size':20}
font2 = {'family':'serif','color':'darkred','size':15}
plt.title("Median Investering Norden", fontdict = font1)
plt.ylabel("investering", fontdict = font2)
plt.xlabel("Land", fontdict = font2)

inv_data2_mean
```

Out[17]:
```
country
DE     9.613333
FI    11.803333
NO    11.649167
SE     5.766667
dtype: float64
```