Oppgave 1a)

N=100; %Velger en tilfeldig verdi for N, ettersom det skal gjelde for alle verdier

Sum = 0; %Definerer sum

for i = 1:N %Får løkken til å kjøre for hvert heltall opp t.o.m. N

Sum = Sum + 1/i; %Summerer alle brøkene

end %Stopper for-løkka

disp(Sum); %Printer ut svaret

Oppgave 1 b)

sum=0; %Definerer sum

N=0; %Starter å telle N fra 0

while sum <= 5 %Løkken skal kjøre så lenge summen er mindre enn 5

N=N+1; %Teller antall ganger løkken kjører/verdi for N

sum=sum+1/N; %Sumerer brøkene

end %Stopper løkken

disp(N); %Printer svaret, her får vi svaret 83.

%Vi kan vite at 256 er større enn 5 ved å del opp og summere i grupper.

%Vi gjør om 1/3+1/4 = 1/2 også følger vi dette systemet, vi får da at 2^n, 2, 4, 8, 16..

%Dette er en måte å finne et raskt svar på. Dette er en rask måte å regne på. Da

%får vi 1+1/2+1/2+1/2+1/2+1/2+1/2+1/2+1/2, dette blir 5. Her må vi da også

%regne ut 2+4+8+16.. siden vi har 8 led må vi regne ut 2^1+2^2.., dette

%blir 255.

Oppgave 1 c)

function sA = sumAlternerende(N) %Navn på funksjon og inputverdi og output

N=5 %Til spørsmålet om hva det blir når N=5

sum=0; %Definerer sum

for i = 0:N %Antall ganger den skal kjøre, begynner på 0

sum=sum+(-1)^i \*(1/2^i) %Formelen som skal summeres

end

sA=sum %Definere en outputverdi

format rat;

sA %Dette gir oss svaret 21/32 når N=5.

Oppgave 2a)

function A = trapes(a,b,h) %Definerer funksjonen, med output og input verdier

A=((a+b)\*h)/2 %Formelen for areal av trapes

Oppgave 2b)

a=4; %Setter a=5

h=2; %h=2

b=10; %Velger et tilfeldig tall for b

sum=0; %Definerer sum

for i = 1:b %Løkken skal kjøre hvert heltall fra 1 til b

sum=sum+trapes(a,i,h); %Henter funksjonen og sumerer

end %Slutter løkken

disp(sum) %Printer ut svaret

%En formel for å regne ut når b/n=10 er ((n+a)\*(n+a+1))/2-(1+2+3..+a). Det vi

%gjør her er å fjerne h siden den er lik nevneren. Vi lager så en lik

%formel som om man skal plusse alle tallene fra 1 til 10, så må vi ta bort

%de første tallene. Dette blir da ((n+5)\*(n+g))/2-(1+2+3+4+5), da får vi

%svaret 105.

Oppgave 2c)

A0 = trapes(5,0,2) %Dette er arealet av trekant, vi får svaret 5.

Oppgave 3a)

function mx = gjsn(x) %Definerer funksjonen og input og output verdi

n=1; %Definerer n og setter en startverdi

sum=0; %Definerer sum

a=length(x); %finner lengde til vektoren

for i = 1:a %Antall ganger løkken skal kjøre

sum=sum+x(n); %Sumerer alle tallene i vektoren

n=n+1; %legger til n hvergang for å plusse neste tall i vektoren

end

mx = sum/a; %Deler på antallet for å få gjennomsnitt.

ov = gjsn(nfpht)

nfpht=[1 2 3 4...n]

ov=(n\*(n+1))/(2\*n) %Dette er en enkel formel for å regne gjennomsnitt av en stigende vektor med hvert heltall

Oppgave 3b)

function vx = varians(x); %Definere en funksjon med outputverdi og input verdi

sum=0; %Definerer sum

a=length(x); %Finner lengden til vektoren

mx=gjsn(x); %Henter funksjonsfilen

for i = 1:a %Lager løkke og antall ganger den skal kjøre

sum=sum+(x(i)-mx)^2; %Summere sammen alle tallene

end

vx=sum/a; %Deler så alle tallene på antall tall i vektoren

vt = varians(t)

%Når t=5, vil summen bli 0. Man tar gjennomsnitt minus tallet, her er

%gjennomsnitt og tallet det samme, vi vil da får 0

Oppgave 3c)

x=[1 2 3 4] %Valgte en tilfeldig x vektor

mx = gjsn(x) %Henter funksjoner fra tidligere oppgaver

vx = varians(x)

sx= sqrt(vx) %Definere sx

a=length(x) %Finner lengden for å finne ut av hvor mange ganger for løkken skal kjøre

svar=1 %Definer svar, må ha 1, fordi vi ganger

for i = 1:a %Løkka kjører for hvert tall i vektoren

svar=svar\*((x(i)-mx)/sx); %Regner ut y

end

y=svar %Setter y = svaret sitt

Oppgave 4

function [x0, n] = hemmeligH2020(g, V, H, fm) %Definerer funksjon og input og output

n=0; %Definerer n skal begynne på 0

while V-H < -fm %Når intervall bredden overstiger feilmarginen halveres intervallet

m= (V+H)/2; %Beregner midtpunktet i intervallet

if g(m)\*g(V) <= 0 %Hvis dette er sant ligger det et nullpunkt i intervallet

H = m; %Intervallet halveres da ved å sette midtpunktet som øvre grense

else

V = m; %Hvis det ikke stemmer blir det halvert og midtpunktet blir ny nedre grense

end

n = n+1; %Teller opp at hvert halverte intervallet.

end

x0 = (V+H)/2; %Midtpunktet mellom intervallet velges som endelig nullpunkt.

%Dette er bisection metoden, den finner 0 punkt i en graf.