```
function [] = Lab()

% Hritik Kumar

% 202051088

% MA202: Lab 03

*Q1. Difference between approximate value and true value*

%a.
function [] = q1 () trueval=exp(0.1);

n=5; % no of terms

h=0.1;

cv=1;

apv=0;

for i=1:n

apv=apv+cv;

cv=cv*(h/i);

err=abs(apv-trueval);

fprintf('error for approximated value till %d terms with stepsize  %f
is %8.20f\n',i,h,err) ;

end

end

q1();
```

error for approximated value till 1 terms with stepsize 0.100000 is 0.10517091807564771244

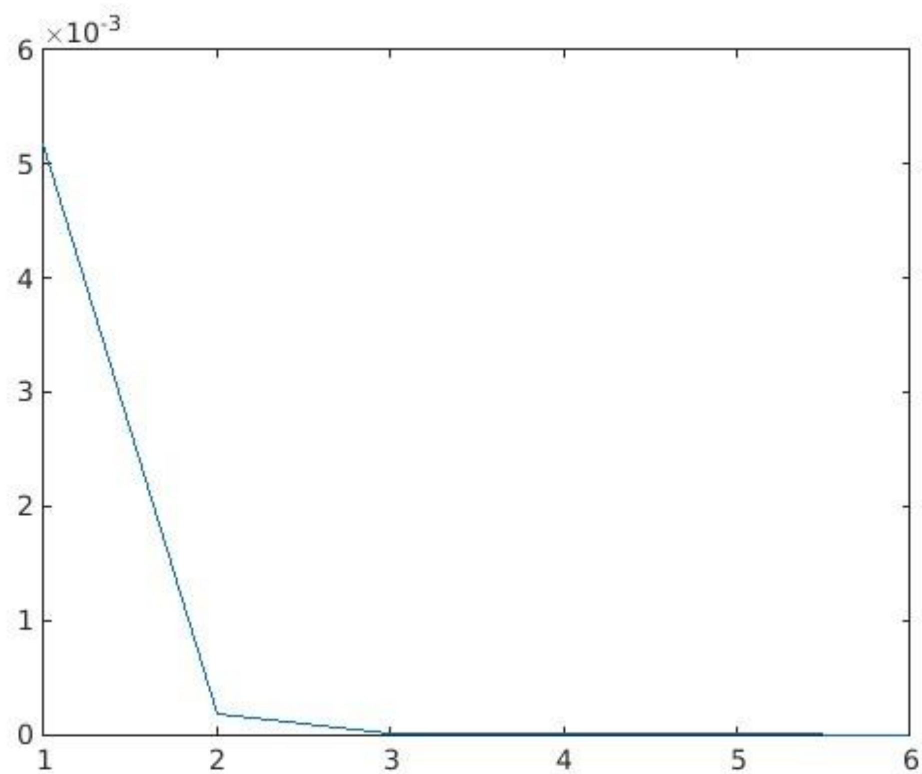
error for approximated value till 2 terms with stepsize 0.100000 is 0.00517091807564762362

error for approximated value till 3 terms with stepsize 0.100000 is 0.00017091807564773021

error for approximated value till 4 terms with stepsize 0.100000 is 0.00000425140898108189

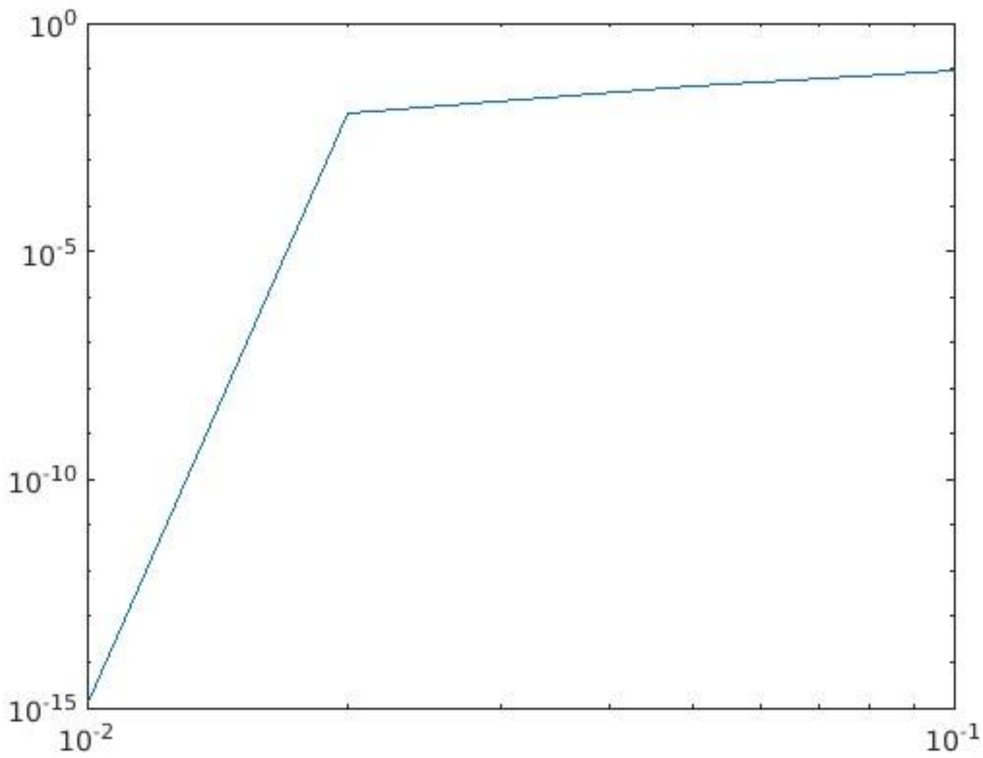
error for approximated value till 5 terms with stepsize 0.100000 is 0.00000008474231449895

```
%b. Plot the error between your approximation and the exact value
function [] = q2()

trueval = exp(0.1);

n=6;

h=0.1;

i=1:n;

cv=h.^i./cumprod(i);

apv=1+cumsum(cv);

err=abs(trueval-apv);

plot(i,err);
end
q2();
```




```
%c. The error using log-log scale for each od the step sizes.
function [] = q3()

steps=[0.1 0.05 0.02 0.01];
```

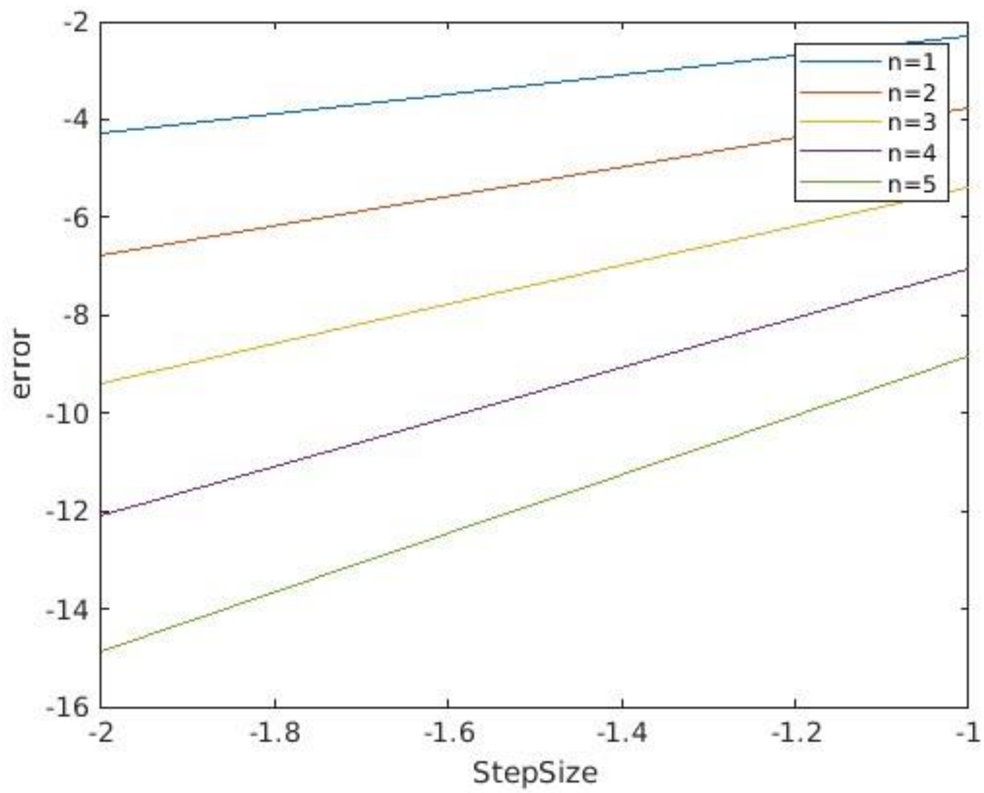
```
trueval=exp(steps);

n=5;

for j=1:length(steps)

i=1:n;

cv=steps(j).^i./cumprod(i); % cumprod is factorial
apv=1+cumsum(cv); % cumsum is addition

err = abs(trueval-apv(n));

loglog(steps,err)

    end

 end

q3();
```



d. What is the slope with respect to the accuracy?

```
    function [] = q4()
n=5;
vec=(1:n);
```

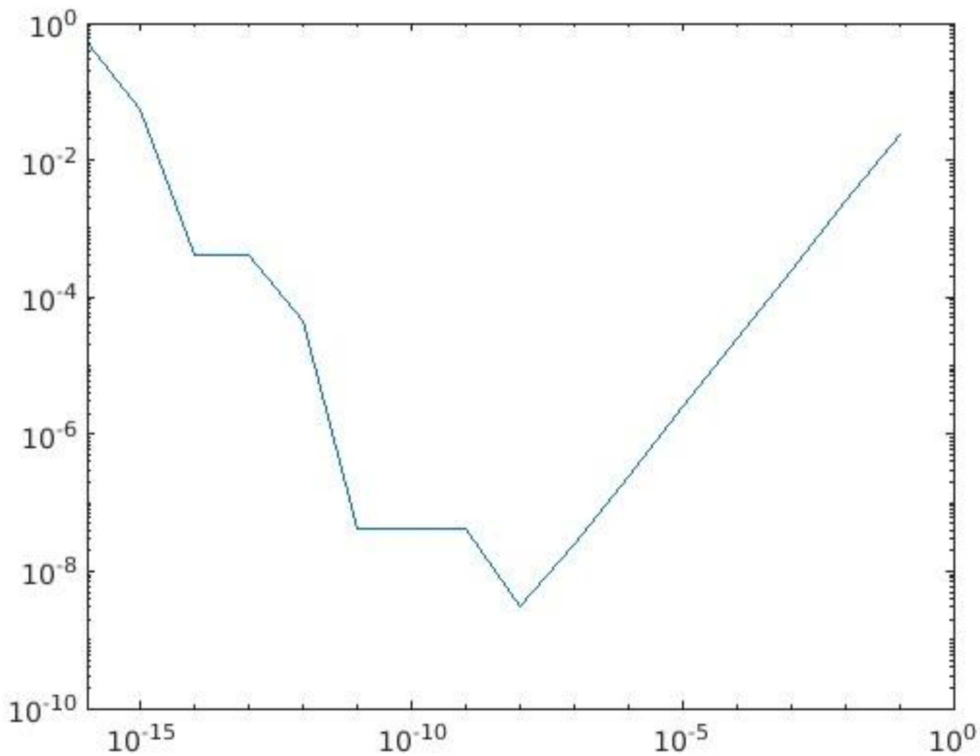
```
steps=[0.1 0.05 0.02 0.01];

e=[];

for i=1:length(steps)
a=steps(i);
term=a.^vec./cumprod(vec);
expval=1+cumsum(term);
trueval=exp(a);
e=[e;abs(trueval-expval)];

end

plot(log10(steps), log10(e))

xlabel('StepSize');

ylabel('error');

legend('n=1' , 'n=2' ,'n=3' ,'n=4' ,'n=5'  )
end

q4();
```



Q2 Calculate numerical derivative of tan^(-1)(x)

```
    function [] = Q2()
x=1;
der=1/(1+x^2);
fprintf('%8.20f\n',der);
 x=1;
h=[-1:-1:-16];
steps=10.^h;
trueval=1/(1+x^2);
fwd=(atan(x+steps)-atan(x))./steps;
error=abs(trueval-fwd);
loglog(steps,error);
    end
Q2();
```

0.50000000000000000000



Q3 Consider a linear system

```
    function [] = Q3()

% a) A=[1.01 0.99;0.99 1.01];

b1=[2;2]; x1 = A\b1 ;
```

```
fprintf("The Solution of equation 4 = \n")
fprintf("%f\n",x1)

% b) b2=[2.02;1.98];

x2= A\b2;

fprintf("The Solution of equation 5 = \n")
fprintf("%f\n",x2)

% c)

 k = cond(A);

fprintf("The condition Number using Function = %f\n",k)
y=norm(A).*(norm(inv(A)));

r=y;

fprintf("The condition Number using equation norm(A).*(norm(inv(A)))
=  %f\n",y)

 q = (norm(x2-x1)./(norm(x1)))./(norm(b2-b1)./(norm(b1))); t=q;
fprintf("The condition Number using equation (norm(x2-x1)./
(norm(x1)))./(norm(b2-b1)./(norm(b1))) = %f\n",q)

  end

  Q3();
```

```
The Solution of equation 4 =
1.000000
1.000000
The Solution of equation 5 =
2.000000
-0.000000
The condition Number using Function = 100.000000
The condition Number using equation norm(A).*(norm(inv(A))) =
 100.000000
The condition Number using equation (norm(x2-x1)./(norm(x1)))./
(norm(b2-b1)./(norm(b1))) = 100.000000
```

Q4. To realize this, find the backward error

```
    function [] = Q4()

A=[1.01 0.99;0.99 1.01];

 b1=[2;2];

x=A\b1;

 b2=[2.02;1.98];
```

```
 x2=A\b2;

fprintf('Backward Error is \n');

norm(b1-b2)    %

    if norm(b1-b2)==0

    abs(x2-x)

    end

  end

Q4();

Backward Error is

ans =

0.0283

end
```