

```

function p = polyeval(a, x)
% init
p = zeros(size(x));
n = length(a)-1;
k = 0;
% add up
for k = 0:n
    p = p + a(k+1)*x.^k;
end
end

```

```

function polyplot(p, x1, x2, varargin)
N = 100*(x2-x1);
x = linspace(x1, x2, N);
y = polyeval(p, x);
plot(x,y, varargin{:});
end

```

```

function c = polycombine(a, b, alpha, beta)
c = polyadd(alpha*a, beta*b);
end

```

```

function c = polymult(a, b)
n = length(a)-1;
m = length(b)-1;
c = zeros(1,m+n+1);
for k = 0:n
    for l = 0:m
        c(k+l+1) = c(k+l+1) + a(k+1)*b(l+1);
    end
end
end

```

```

function c = polypow(a, n)
assert(n>=0, 'Power must be non-negative');
c = [1];
for k = 1:n
    c = polymult(c, a);
end
end

```

```

p = [1 2 3 4];
q = [4 3 2];
polyadd(p, q)
polycombine(p, q, 1, -1)
polyeval(q, rand(2))
polymult(p,q)
polypow([1 -1],5)
polyplot([0 0 1],-1,1)

```

```

function c = polyadd(a, b)
% aus Vorlesung kopiert !

```

```

n = length(a)-1; % Laenge des Vektors a entspricht Polynomgrad+1
m = length(b)-1; % Laenge des Vektors b entspricht Polynomgrad+1
MIN = min(m,n); % Minimum von m,n fuer Schleifendurchlaeufer
%
c = zeros(1,max(m,n)); % Speicherreservierung und Initialisierung
for k=0:MIN
    c(k+1)=a(k+1)+b(k+1); % hier gibts sowohl ak als auch bk
end
if n>m % mehr ak's als bk's
    for k=MIN+1:n
        c(k+1)=a(k+1); % der Rest der neuen Koeffizienten sind ak's
    end
else % mehr bk's als ak's
    for k=MIN+1:m
        c(k+1)=b(k+1); % der Rest der neuen Koeffizienten sind bk's
    end
end
end

```