Aufgabe 1

1. Für den obigen Algorithmus benötigt man n Additionen und $n + \frac{n(n+1)(2n+1)}{6}$ Multiplikationen, denn man addiert jeder Mitglied des Polynoms n-mal und man berechnet jede Potenz mit dem Formel $\sum_{i=0}^{N} i^2$. Dann gibt es auch n für die Koeffizienten-Multiplikation.

2. a) Wir wissen dass n^2 gleich mit der Summe der ungeraden Zahlen bis einem Indizes ist, deswegen können wir das Polynom folgendes mit dem Horner Schema schrieben:
   $(...(a_n x^{2n-1} + a_{n-1})x^{2n-3} + a_{n-2})...)x + a_0$

   b) Mit dem Ergebnbis von 2 a) berechnet man effizienter das Polynom, weil man nicht die Quadrat jeder Potenz berechnen muss. Das impliziert, dass die Nummer der Multiplikationen kleiner ist. Ein Beispiel wäre für $n = 3$, wenn man 17 Multiplikationen mit dem classichen Algorithmus im Vergleich zu 12 macht.

   c)

```
    q=1;
for(i=n;i>0;i--)
 {
  potenz=0;
    for(j=1;j<2*i-1;i--)
     potenz=potenz*x;
   q=a[i]*potenz+a[i-1]+q;
  a[i-1]=q;
  }
```

---

d) Der verbesserte Algorithmus berechnet das Polynom mit einer Komplexität von O(n^2) und macht $n(n+1)$ Multiplikationen und $n-1$ Additionen.

3. a) Mit dem Hinweis können wir jede Potenz in Bezug auf die vorherige Potenz berechnen. Zusätzlich, wissen wir dass die Differenz zwischen $n^2 - n^{n-1}$ glecih $2n-1$ ist. Zum Beispiel, können wir $(x^3)^2$ als $(x^2)^2$ mal $(x^2)^{3 \cdot 2 - 1}$

   b)

   #Codeblock

```
  q=x[0]+x[1]*x;
  lastpotenz=x;
  for(i=2;i<=n;i++)
 {
  potenz=1;
  for(j=1;j<2*i-1;i--)
   potenz=lastpotenz*x;
  lastpotenz=potenz
  q=a[i]*potenz+q;

 }
```

---

c) Diese verbesserte Version macht insgesamt n Additionen und $(3n + 1 + n \cdot (n + 1))$ Multiplikationen.