Aufgabe 1:

1 a) worst case kann man beobachten wenn die Liste absteigend sortiert ist. Also in diesem Fall muss man jedes neue Element mit allen Elementen vorne vergleichen, weil sie alle größer sind.

Die Funktion insert(xs, i) wird n-1 Mal aufgerufen. Zum ersten Mal ist nur ein Vergleich nötig, zum zweiten – 2 Vergleiche, ..., zum n-1 Mal benötigt man dann n-1 Vergleiche. Also könnte man dann ~~...~~ aufschreiben:

$$1 + 2 + 3 + \ldots + n-2 + n-1 = \sum_{i=1}^{n-1} i = \frac{n^2-n}{2} =$$

$$= \frac{1}{2}n^2 - \frac{1}{2}n \Rightarrow O(n^2)$$

b)

42 13			17 121		
erg = 0	a = 42	b = 13	erg = 0	a = 17	b = 121
erg = 0	a = 21	b = 26	erg = 121	a = 8	b = 242
erg = 26	a = 10	b = 52	erg = 121	a = 4	b = 484
erg = 26	a = 5	b = 104	erg = 121	a = 2	b = 968
erg = 130	a = 2	b = 208	erg = 121	a = 1	b = 1936
erg = 130	a = 1	b = 416	erg = 2057	a = 0	b = 3872
erg = 546	a = 0	b = 832	return 2057		
return 546					

c) ~~sorted_zip~~: list_sum:

Für jedes Element in der Liste wird eine Additionsoperation durchgeführt. Also n = len(xs) Operationen insgesamt.

$O(n)$

sorted_zip:

Laufzeit der Funktion mergesort(xs) ist gleich $2n\log(n)$, wobei n = len(xs) ist.

Laufzeit der Funktion mergesort(ys) ist gleich $2m\log(m)$, wobei m = len(ys) ist.

Dann in der Schleife ~~sind~~ werden jedes Mal zwei Elemente kopiert

Falls xs länger als ys ist dann die Laufzeit muss gleich $2 \cdot n = len(xs) \cdot 2$ sein. Andernfalls ist sie $2 \cdot m = len(ys) \cdot 2$

Wenn man alle Laufzeiten addiert:

$2n \cdot \log(n) + 2m \cdot \log(m) + 2n$ (oder $2m$)

Wenn m und n unbegrenzt groß werden, dann alle Faktoren und den Summanden 2n kann man vernachlässigen.

$O(n \cdot \log(n))$, wo n die Länge der größeren Liste ist.

bauernmultiplikation:

Hier ist es sinvoll, die Anzahl von Vergleichen zu rechnen

Es werden solange die Vergleiche gemacht, bis die Zahl (die erste Zahl a) gleich 0 wird. Also:

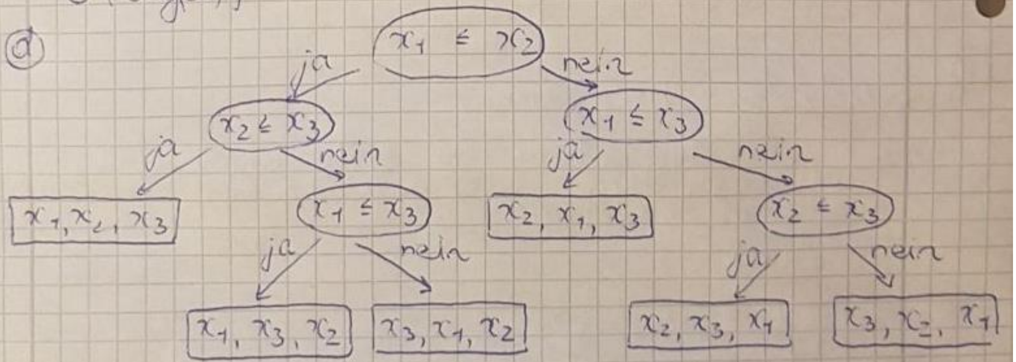
$$\frac{a}{2^{n-1}} \geq 1$$

$$a \geq 2^{n-1} \mid \log$$

$$\log a \geq n-1 \Rightarrow n \leq \log a + 1$$

$$O(\log a))$$

d)



Bei der Aufgabe 1c) wäre es vielleicht sinnvoller *die Laufzeit für den schlimmsten Fall zu* Schreiben, *wenn die Länge der kleineren Liste der Länge der größeren Liste ist, also n = m. Dann würde das so aussehen:*

2nlog(n) + 2nlog(n) + 2n = 4nlogn + 2n

O(n) = nlogn, wobei n Länge der Listen ist.