Aufgabe 1 (Speicherbedarf)

Das sogenannte *15 Puzzle* ist ein Spiel, bei dem Felder auf einem quadratischen Spielfeld der Größe 4×4 in die richtige Reihenfolge gebracht werden müssen. Für jedes Feld soll im folgenden gespeichert werden, ob es bereits an der richtigen Stelle liegt oder nicht. Die benötigten Informationen können entweder als Liste vom Typ `Integer` oder als Liste vom Typ `Bool` gespeichert werden. Der Speicherbedarf beträgt abhängig von der Länge n der Liste entweder $S_{Int}(n) = 1 + 4 \cdot n$ bzw. $S_{Bool}(n) = 1 + n$. Dabei gibt S_{Int} bzw. S_{Bool} den Platzbedarf in Byte an.

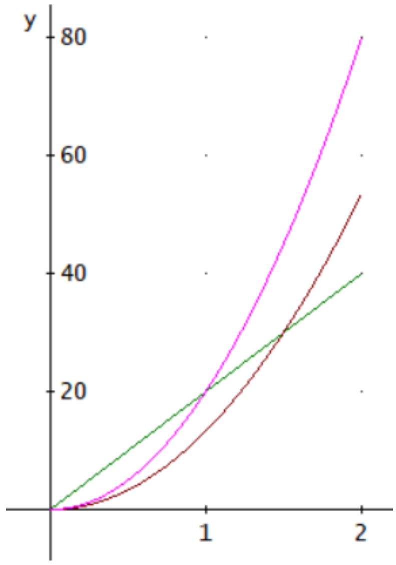
(a) Zeigen Sie, dass der Platzbedarf bei Verwendung von `Integer`-Variablen genau 65 Byte beträgt.

(b) Ermitteln Sie die prozentuale Ersparnis an Speicherplatz wenn stattdessen Werte vom Typ `Bool` zum Speichern der Information genutzt werden.

(c) Die Spielfeldgröße soll nun variabel gestaltet werden, d.h. das Spielfeld besteht aus $m \times m$ Feldern. Stellen Sie die quadratische Funktion $S_{Int}(m)$ auf, die den Speicherbedarf in Abhängigkeit von m angibt und berechnen Sie, wie groß m höchstens sein darf, wenn der maximal verfügbare Speicher 401 Byte beträgt.

Aufgabe 2 (Sortierverfahren)

Ein einfaches Sortierverfahren ist das sogenannte *Insertion Sort*[1]. Die Laufzeit des Verfahrens ist im günstigsten Fall (Best Case) linear. Im durchschnittlichen und schlimmsten Fall (Average Case bzw. Worst Case) ist diese quadratisch in Abhängigkeit von der Anzahl der zu sortierenden Elemente. Betrachten Sie dazu die folgende Graphik. Ein Schritt auf der x-Achse entspricht hier 100 Elementen. Auf der y-Achse ist die Laufzeit in Millisekunden aufgetragen.



(a) Ordnen Sie die drei Graphen den Funktionen zum Best Case, Average Case und Worst Case zu.

(b) Eine Messreihe für eine Implementierung in `Haskell` ergibt für den Worst Case die Gleichung $wc(n) = 20 \cdot n^2$. Durch neue Hardware soll sich die Laufzeit für alle Eingabelängen um 10% verbessern. Ermitteln Sie die verbesserte Gleichung für $\overline{wc}(n)$.

(c) Bestimmen Sie die Funktionsgleichung für den Best Case aus der obigen Graphik.

(d) Zum alternativen Sortierverfahren *Quicksort*[2] ergibt sich die folgende Messreihe.

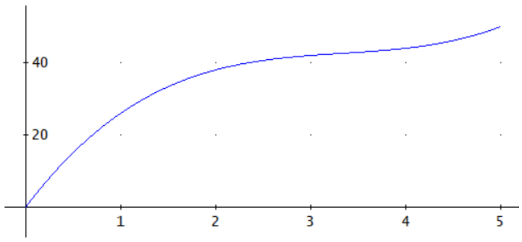
x	$\frac{1}{3}$	$\frac{2}{3}$	1	$\frac{4}{3}$	$\frac{5}{3}$	2
y	2,38	8.06	15.93	25.44	36.26	48.18

Zeichnen Sie die Punkte in der Graphik ein und begründen Sie, warum die zugehörige Funktion weder linear noch quadratisch ist.

[1] Details siehe: `https://de.wikipedia.org/wiki/Insertionsort`
[2] Details siehe: `https://de.wikipedia.org/wiki/Quicksort`

Aufgabe 3 (Kubischer Spline)

In der Graphikprogrammierung verwendet man für die Darstellung komplexer Figuren in der Regel sogenannte *Splines*. Dabei handelt es sich aus mathematischer Sicht häufig um kubische Funktionen. (In der Praxis werden sogenannte Bezier-Kurven verwendet.)



Der in der Abbildung gezeigte Spline hat die Gleichung $S(x) = x^3 - 10x^2 + 35x$.

(a) Zeigen Sie, dass für $0 \leq x \leq 5$ durch $g(x) = 10 \cdot x$ eine Linearisierung des Splines, d.h. eine Approximation durch eine lineare Funktion gegeben ist.

(b) Weisen Sie rechnerisch nach, dass $S(x)$ keine weiteren Nullstellen außer $x = 0$ besitzt.

(c) Für die Darstellung einer Bewegung soll die Veränderung der Spline-Funktion $S(x)$ erfasst werden. Bestimmen Sie hierfür die Ableitungsfunktion $S'(x)$.