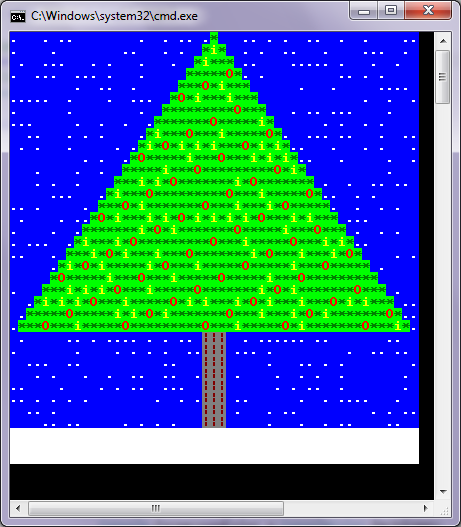
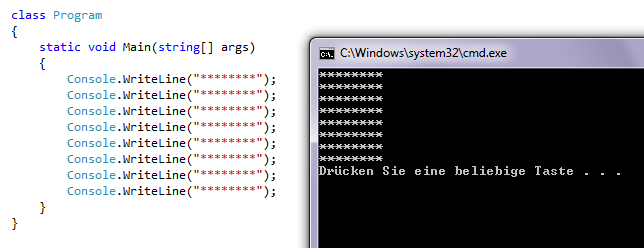
# Weihnachtsbaumtutorial



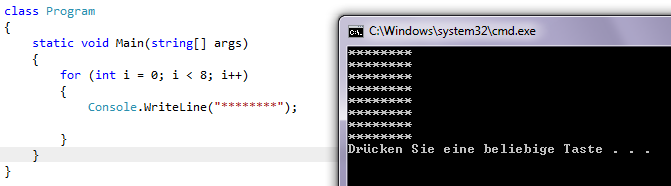
**Erste Schritte:**

Wir erzeugen zuerst ein 8x8 Quadrat aus Sternen:

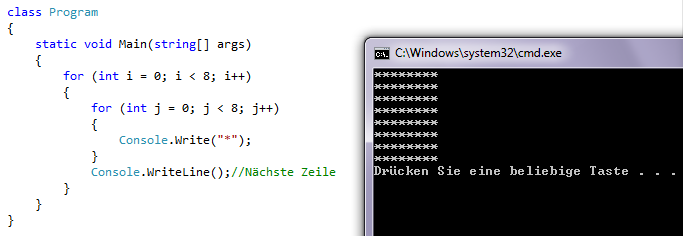


Dass kein wirkliches Quadrat erscheint, liegt an der Schriftart der Console, welche 8x12 standardmäßig eingestellt hat. Experimentiere mit anderen Schriftgrößen.

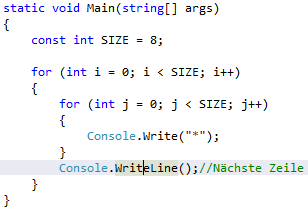
Diese Lösung ist nicht wirklich flexibel, besser wäre eine Schleife:  
(wir machen ja 8 „Zeilen“ und 8 „Sterne“:



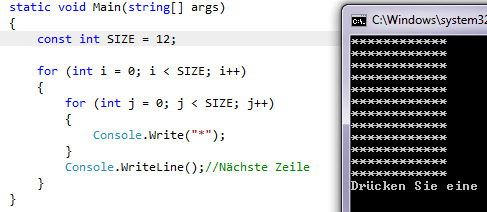
Als nächsten Verbesserungsschritt geben wir nur mehr 1 Zeichen aus;   
nach jeweils 8 Zeichen springen wir in die nächste Zeile:



Nun fällt auf, dass die Zahl 8 2 mal vorkommt. Wenn wir nun ein Quadrat einer anderen Größe zeichnen wollen, muss an 2 Stellen der Code geändert werden. Besser ist es, dafür eine Variable zu nehmen und diese gleich als Konstante zu definieren:

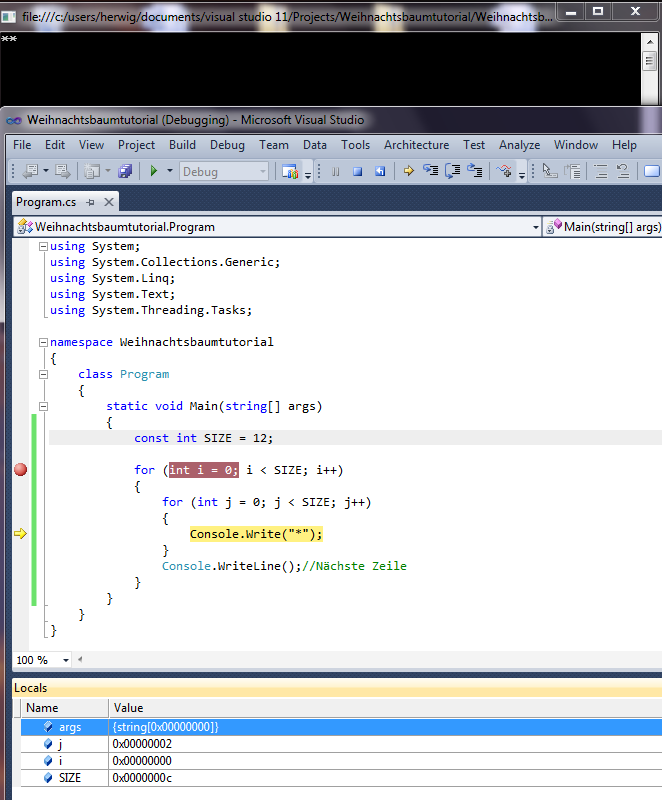


Das Programm liefert immer noch das gleiche Ergebnis; setzt man SIZE = 12 so, sieht die Ausgabe jedoch folgendermaßen aus:

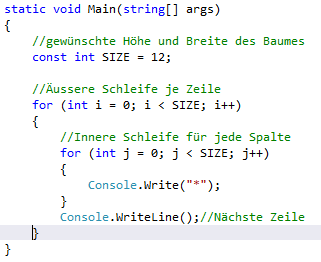


Wir sollten spätestens jetzt mehr kommentieren: Welche Schleife ist für die Zeilen zuständig? Welche Schleife macht die Spalten nebeneinander? Wir setzen einen Breakpoint zur ersten FOR-Schleife und debuggen das Programm:

* Starte nun das Programm mit F5. Sobald es stoppt, springe mit F10 oder F11 weiter. Beobachte die Werte der Variablen im unteren Fenster.
* Konfiguriere während des Debuggens Visual Studio (Menü Debugge -> Fenster -> Lokal) sodass die Variableninhalte angezeigt werden!
* Was passiert?

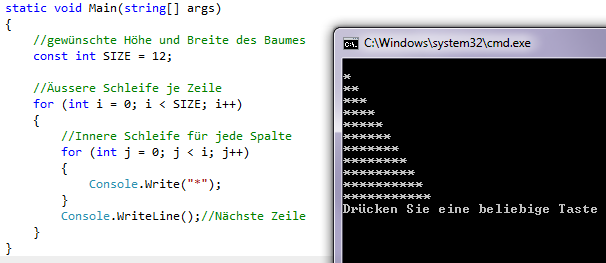


Wir sehen, „i“ ist noch 0, während „j“ auf 2 ist, im nächsten Schritt wird das nächste „\*“ gezeichnet…

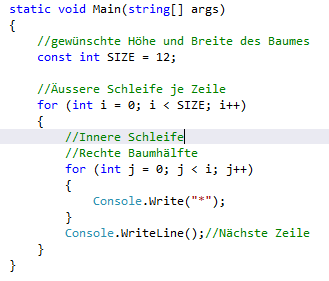


Nun sind wir mit dem Quadrat fertig. Wir wollen nun ein Dreieck zeichnen: die erste Zeile soll aus \* bestehen, die nächste aus \*\*, dann \*\*\* usw.

Was muss getan werden? Wir probieren, die innere Schleife umzuschreiben, sodass diese nicht bis „SIZE“ läuft, sondern bis (wohin eigentlich?) - probieren wir einmal „i“:



Das sieht schon fast wie ein halber Baum aus ;-) also kommentieren wir das entsprechend!



Bevor wir weitermachen, schauen wir uns in einer Skizze den Aufbau des Baumes genauer an:

Baum   
rechts

Baum   
links

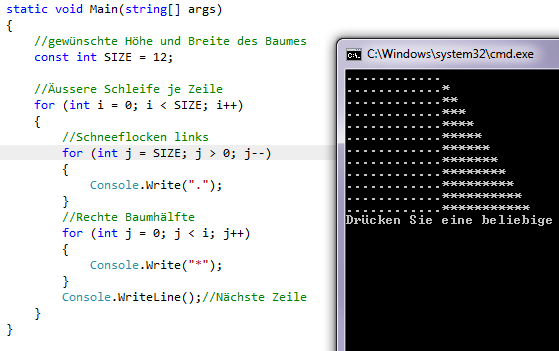
Schnee-flocken  
links

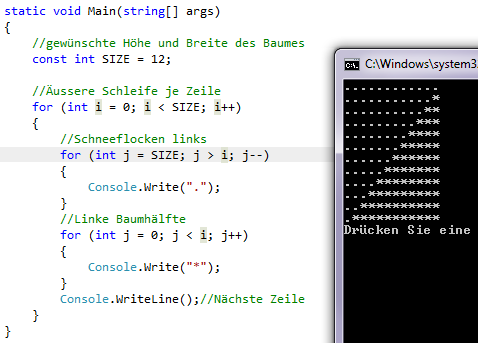
Schnee-flocken  
rechts

Man kann sich vorstellen, aus 4 Dreiecken den Baum zu erstellen:

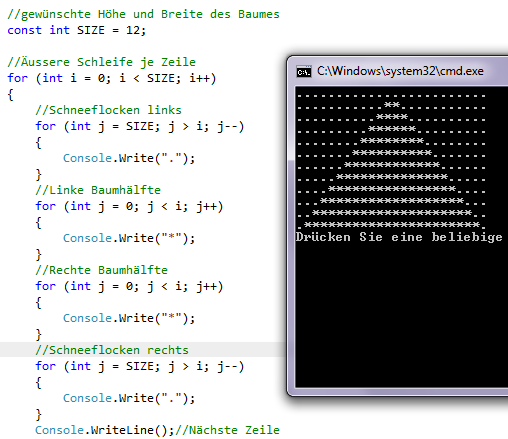
Dabei sind offenbar die linken und rechten Teile spiegelgleich zu lösen.

Wie kann das Dreieck nun am Kopf stehen?

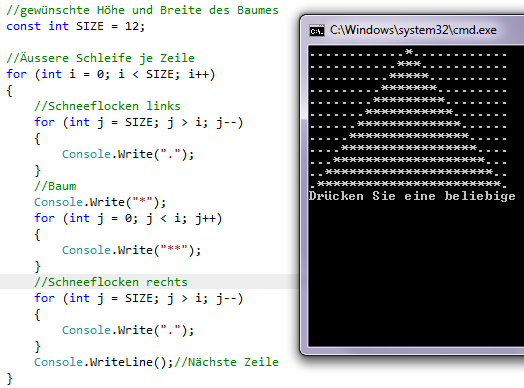
Wir probieren eine Schleife, die anfangs viele „.“ zeichnet, danach je Zeile immer wieder um 1 weniger: ( ACHTE AUF DIE KOMMENTARE !!)



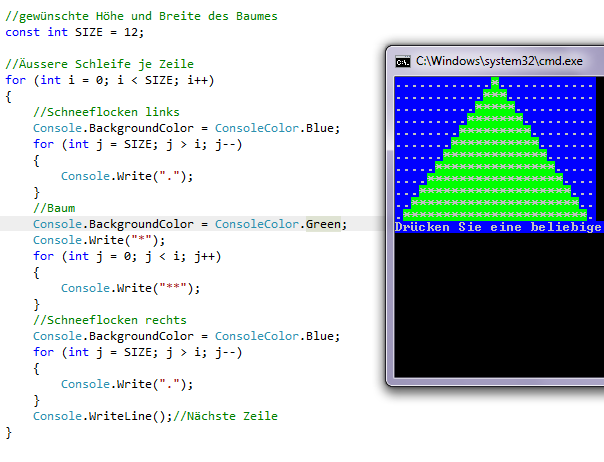
Jetzt brauchen wir nur mehr den Code zu „spiegeln“ und fertig ist ein einfacher Baum!



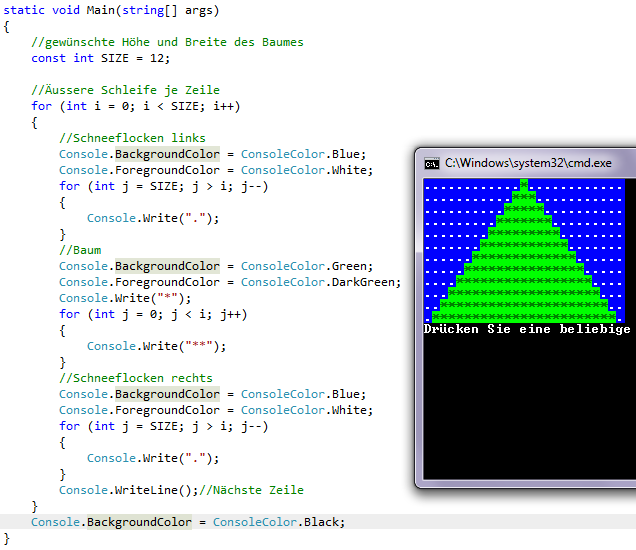
Die linke und rechte Hälfte können wir gleich zusammenfassen, dann wird’s übersichtlicher!



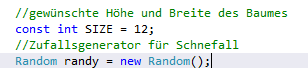
Ein bisserl Farbe gefällig? Warum nicht!



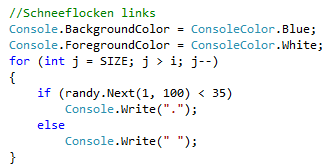
Noch besser:



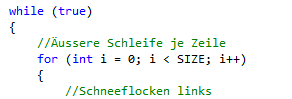
Wir lassen es jetzt weniger „schneien“, also wir zeichnen nicht immer einen „.“ Sondern ein Leerzeichen, abhängig von Zufallswerten:



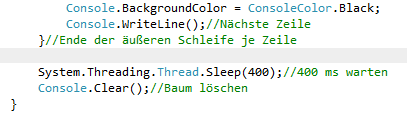
…

Eine nette Animation:



(…)



Nächste Aufgabe: Ergänze einen Stamm! Nach dem Ende der Baumschleife wird ein Stamm gezeichnet, der 1/3 der Baumhöhe hat. Natürlich soll es links und rechts schneien.   
KOMMENTIERE alle Programmabschnitte WIE OBEN!

SIZE

SIZE

Baum   
rechts

Baum   
links

Schnee-flocken  
links

Schnee-flocken  
rechts

SIZE

Schneeflocken  
rechts

stamm

Schneeflocken  
links

SIZE/3

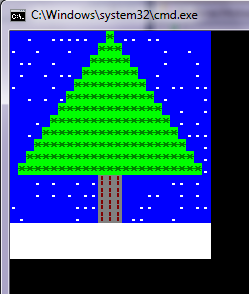
SIZE - 1

3

SIZE - 1



Nächste Aufgabe: Am Boden soll Schnee liegen…



Nächste Aufgabe: (sei kreativ … !)

* Platziere Kugeln o … etwa regelmäßig
* Platziere Kerzen i … etwa komplett zufällig

