**IMPLEMENTIERUNG**

**Polynomdarstellung**

Um in der Programmiersprache Python Polynome darzustellen, wird die Datenstruktur Tupel verwendet. In einem Tupel können mehrere Inhalte unveränderlich gespeichert werden. Da in der Charakteristik Zwei lediglich Zahlen aus dem Binärsystem Vorfaktoren der verschiedenen Monome sein können, werden die Tupel nur mit Nullen und Einsen befüllt. Dadurch wird bestimmt, ob ein bestimmtes Monom in dem Polynom vorkommt (in diesem Fall mit einer „1“ an der entsprechenden Stelle dargestellt) oder ob das Monom nicht in dem Polynom vertreten ist (Mit einer „0“ dargestellt).

In diesen Tupeln steht das Monom mit dem höchsten Exponent an der ersten stelle (entspricht der Stelle ganz links) und die Exponenten nehmen dann schrittweise nach rechts ab, sodass die letzte Stelle (ganz rechts) dem Monom mit dem Exponent „0“ entspricht. Diese Darstellung durch Tupel vereinfacht den Umgang mit Polynomen, da die ausführliche mathematische Schreibweise zusätzliche, unnötige Informationen wie die Variable „x“ oder das „+“-Zeichen. Für den Umgang mit Polynomen ist jedoch nur wichtig, welcher Vorfaktor an welcher Stelle des Polynoms steht, und diese beiden Informationen sind in der Schreibweise als binäre Tupel enthalten, wodurch sie sich anbietet für das Arbeiten mit Polynomen.

Um solch ein binäres Tupel in Python wieder in die mathematische Schreibweise eines Polynoms mit Vorfaktor, Variable „x“ und Exponent für jedes Monom darzustellen wurde die Funktion „TupleToPolynom“ erstellt. Wie der Name schon sagt, bekommt die Funktion ein Tupel als Funktionsparameter und gibt einen String zurück, der das vollständige Polynom umfasst. Dabei iteriert die Funktion über die einzelnen Elemente des Tupels und bestimmt daraus einen Teilstring. Bei der Erstellung dieser Teilstrings gibt es drei verschiedene Fälle. Bei dem letzten Monom eines Tupels, also dem Monom ganz rechts, wird der Teilstring lediglich eine „1“, da das Letzte Monom den Exponenten Null hat, und alles hoch Null gleich Eins ergibt. Bei dem vorletzten Monom wird der Teilstring zum einem „X“, da dieses Monom den Exponenten Eins hat. Für alle anderen Fälle, bei denen der Exponent größer als Eins ist, wird der Teilstring zu „X^n“, wobei n dem Exponenten des Monom entspricht. Im Zuge dieser Funktion wird das Tupel vor der Erstellung des Lösungsstrings einmal umgekehrt, sodass nun das letzte Monom an erster Stelle steht, das vorletzte an zweiter Stelle steht, und so weiter. Dadurch gewinnt man den Vorteil, dass jetzt die Position des Monoms in dem Tupel gleich dem Exponenten des Monoms entspricht. Da bei Tupeln, ähnlich wie bei Arrays das Zählen bei der Null anfängt steht nach der Umkehrung das erste Monom (was ursprünglich das Letzte war) an der Position Null in dem Tupel, was genau dem Exponenten dieses Monoms entspricht. All diese einzelnen Teilstrings werden dann aneinandergereiht, wobei zwischendrin immer ein „+“ beigefügt wird. So erhält man schließlich einen String der das vollständige Polynom, wie man es auch in der mathematischen Schreibweise aufschreiben würde.

Diese Funktion, die aus einem Tupel wieder ein Polynom macht, erfüllt keinen direkten Zweck, der für weitere Operationen wichtig ist, sondern diese Funktion dient lediglich dazu, aus einem Tupel eine für den Anwender lesbare Darstellung zu bieten, da die Tupel, vor allem bei steigender Länge unübersichtlich werden können.

**Arithmetische Operationen**

Da die Charakteristik Zwei auf dem Binärsystem basiert, verhalten sich die arithmetischen Operationen innerhalb von Körpern der Charakteristik Zwei analog zu den Berechnungen im Binärsystem. Im Folgenden wird die Implementierung der Arithmetischen Rechenoperationen für binäre Tupel, welche Polynome der Charakteristik Zwei darstellen, genauer erläutert.

Addition von Polynomen

Bei der Addition von Polynomen in Körpern der Charakteristik Zwei gelten die Rechenregeln wie bei der Addition im Binärsystem. Diese lauten folgendermaßen. 0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1 und 1 + 1 = 0 (mit Übertrag 1).

Um nun diese Addition nun in Python zu implementieren, werden der Funktion „AddPolynoms“ zunächst zwei Polynome übergegeben, die in Form von binären Tupeln dargestellt werden. Um diese beiden binäre Tupel zu addieren, wird über die Stellen der Tupel iteriert, und jede Stelle einzeln, nach den oben gegebenen Rechenregeln, binär addiert. Die Ergebnisse werden dann schrittweise dem Lösungspolynom hinzugefügt.

Bevor die Iterative Addition der Stellen der Polynome durchgeführt werden kann, muss jedoch sichergestellt werden, dass auch jeweils die passenden Stellen miteinander addiert werden. Als Voraussetzung dafür, dass immer die passenden Stellen miteinander addiert werden, gilt, dass die beiden Polynome, beziehungsweise die beiden Tupel, gleichviele Stellen haben. Um sicherzustellen, dass die beiden Tupel gleichviele Stellen haben, wird bevor der iterativen Addition geprüft, welches Tupel das längere der beiden ist, und im Anschluss wird das kürzere Tupel von links aus mit Nullen ergänzt, bis es so viele Stellen wie das jeweils andere hat. Diese Nullen dienen lediglich zur Richtigen Zuordnung der Stellen bei der Addition. Zudem ändern sie nichts an den Rechnungen, da die Zahl Null das neutrale Element in der Addition darstellt. Demnach wird durch diese Auffüllung mit Nullen das Ergebnis der Addition nicht beeinflusst.

Schließlich wird am Ende das Lösungspolynom in Form eines binären Tupels zurückgegeben.

Multiplikation von Polynomen

Polynomdivision