

# Aufgabe a4x1

## Funktionsweise:

- **Zwei Konstruktoren:** Standard-Konstruktor (ohne Parameterübergabe) und Parameterversion mit Aufruf der Methode *hasValidCards()* um sicherzustellen, dass keine Karte ein null-Wert ist.
- In der Methode (*Prozedur*) **add (Karten)**: Erstellen neues Array mit kombinierter Länge und kopieren zuerst alle vorhandenen Karten und dann die, die hinzugefügt werden
- **In der Methode** (*Prozedur*) **add (Hand)**: Erstellen neues Array mit kombinierter Länge und kopieren zuerst alle vorhandenen Karten und dann die, die aus der anderen Hand hinzugefügt werden.
- **In der Methode** (*Funktion*) **isSuited**: Initialisieren eine int-Variable, die speichert die Farbe der ersten Karte im Array. Falls alle Karten dieselbe Farbe haben, wird wahr zurückgegeben, sonst falsch.

# Aufgabe a4x2

UML-Klassendiagramm: Konten A4X2

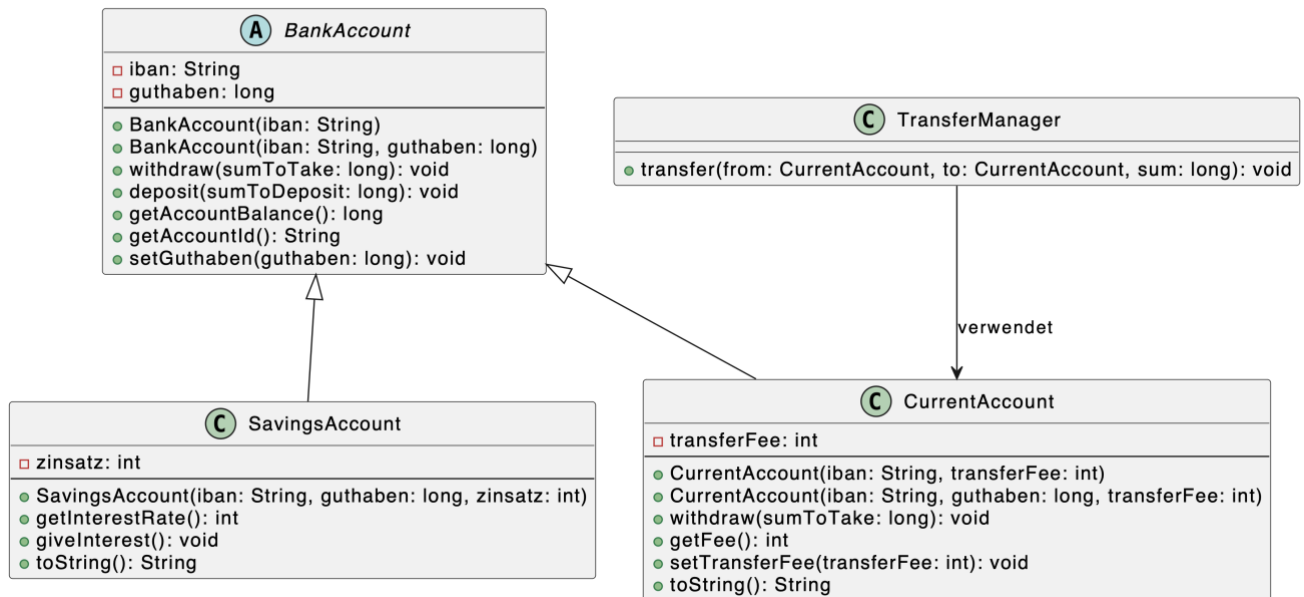


Abb. 1: UML-Klassendiagramm von Konten

## Anmerkungen:

- Spezifische Methoden werden bei Bedarf überschrieben, um die Funktionalität an die jeweiligen Anforderungen der Unterklassen anzupassen.
- *BankAccount* ist eine abstrakte Klasse, da sie eine Oberklasse/Elternklasse für alle Arten von Bankkonten dient.

## Sicherheit:

- **Der Konstruktor** unterscheiden zwei Eingabemöglichkeiten: mit oder ohne Guthaben auf dem Konto. Falls kein Guthaben angegeben ist, wird automatisch 0 gesetzt. Die Korrektheit der IBAN-Nummer wird dadurch sichergestellt, dass überprüft wird, ob sie nicht *null* ist. Ist die IBAN *null*, so wird Assertion ausgelöst und das Programm beendet.
- **In der Methode (Prozedur) *withdraw*** wird auf folgende Aspekten geachtet:
  1. Der Betrag, der abgebucht werden muss, muss positiv sein.
  2. Auf dem Konto liegende Guthaben muss größer sein als der abzubuchende Betrag.
  3. Nur in der überschriebenen Methode (Prozedur) *withdraw* der Unterklasse *CurrentAccount*: Das Guthaben muss größer sein als die Summe aus Betrag und Gebühr
- **In der Methode (Prozedur) *deposit*** muss die einzuzahlende Summe positiv sein.
- **Die Methode (Prozedur) *giveInterest*** ist nur auf Konten anwendbar, bei denen ein Guthaben vorhanden ist.
- **In der Methode (Prozedur) *transfer*** muss Quell- und Ziel-Giro-Konto unterschiedlich sein.

## Aufgabe a4x3

### Funktionsweise:

- In jeder Iteration der While-Schleife werden alle Buchstaben innerhalb des aktuellen Musters verglichen.
- In der ersten Iteration entspricht das Muster der Länge des gegebenen Strings.
- Sollte kein Palindrom gefunden werden, wird das Muster jedes Mal um eins verkleinert und dabei wird die Anzahl der Iterationen um eins erhöht, dabei werden mehr Startpositionen geprüft.
- Die While-Schleife wird so lange wiederholt, bis die Länge des Musters kleiner als eins ist, also *aktuelleLangeMuster* < 1, oder bis das erste Palindrom gefunden wird.

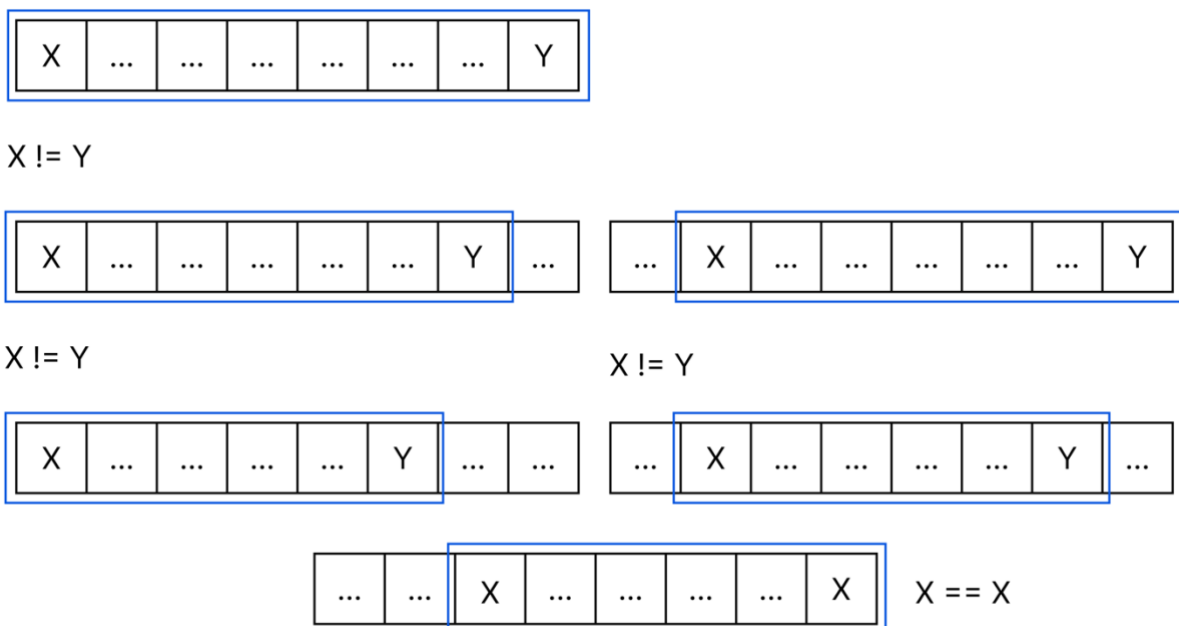


Abb. 2: Funktionsweise der Methode *getLongestPalindrome*

### Datenorganisation:

- **While-Schleife (äußere)**  
**Grund:** unbekannte Anzahl der Iterationen  
läuft bis Palindrom gefunden oder Musterlänge < 1
- **For-Schleife (mittlere):**  
**Grund:** bekannte Anzahl der Iterationen  
alle Startpositionen für aktuelle Musterlänge durchführen
- **While-Schleife (innere):**  
**Grund:** dynamischer Abbruch bei Palindrom-Prüfung von außen nach innen

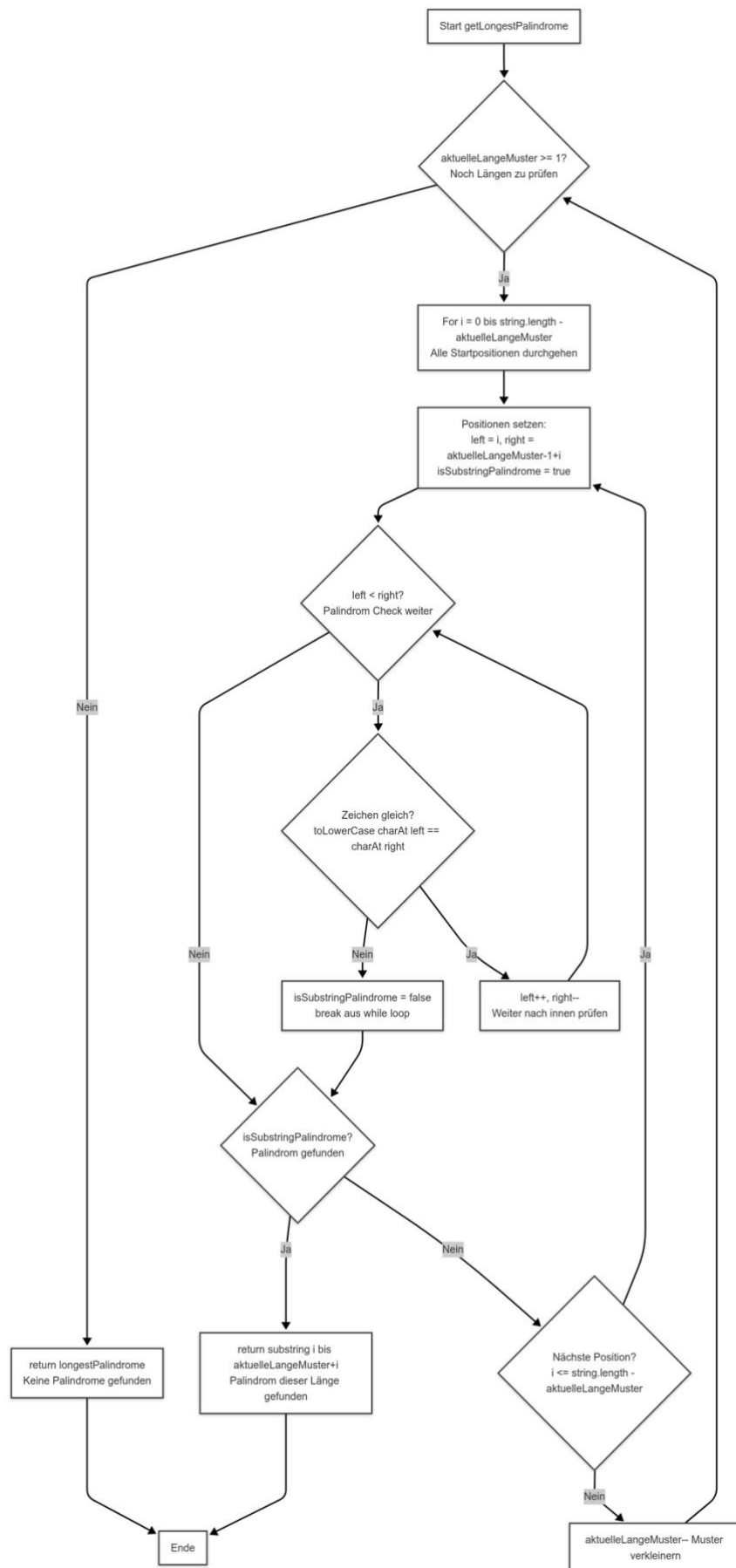


Abb. 3: Flussdiagramm der Methode getLongestPalindrome

# Aufgabe a4x3v2

## Funktionsweise:

- In der Methode mit dem Rückgabewert String wird ein Eingabestring übergeben, der auf Palindrome untersucht werden soll.
- Dabei wird geprüft, ob die Anzahl der Buchstaben in String *gerade oder ungerade\** ist, um alle möglichen Palindromzentren zu finden.
- Für jedes Zeichen im String wird versucht, das längste Palindrome mit diesem Zeichen aus Zentrum (sowohl bei gerader als auch bei ungerader Länge zu finden).
- Wird bei der Überprüfung eine Ungleichheit festgestellt, wird die Suche für dieses Zentrum abgebrochen.
- Am Ende wird das längste gefundene Palindrom zurückgegeben.

1\*: Der Algorithmus prüft zwei Fälle pro Position:

1. **Ungerade Länge** (z.B. "aba" mit dem Zentrum bei "b")
2. **Gerade Länge** (z.B. "abba" mit dem Zentrum bei "b" und "b")

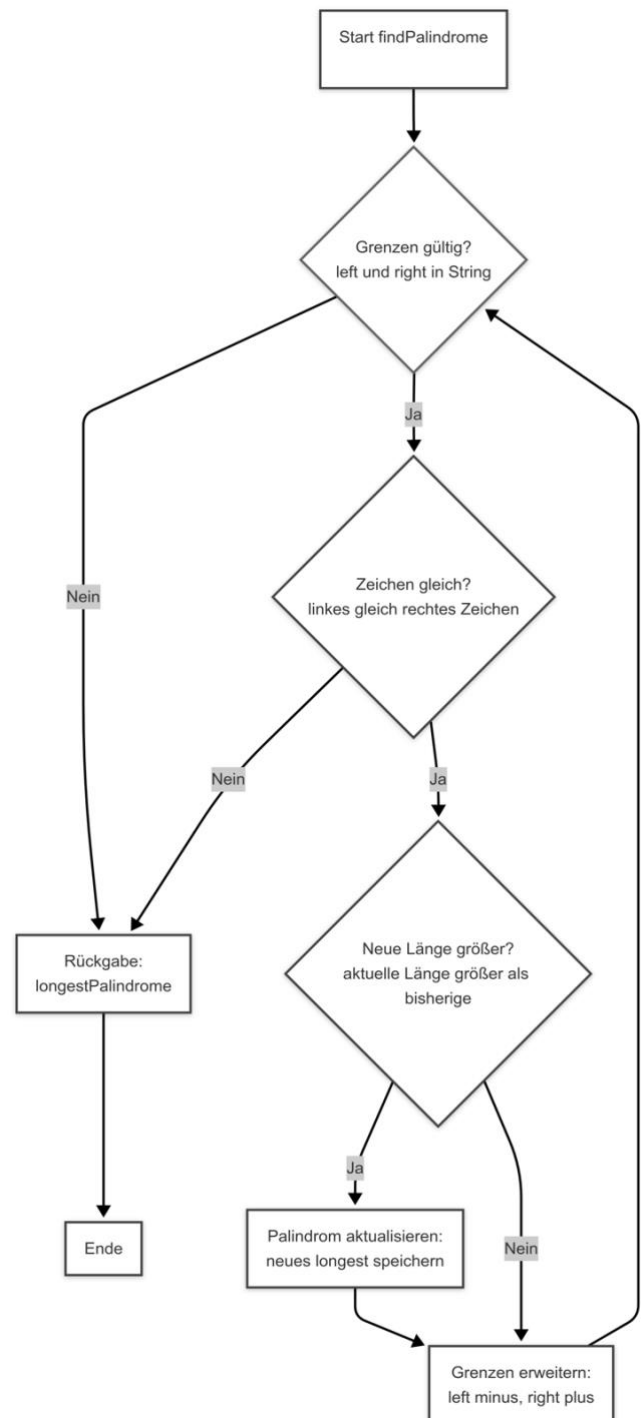


Abb. 4: Flussdiagramm der Methode findPalindrome v2

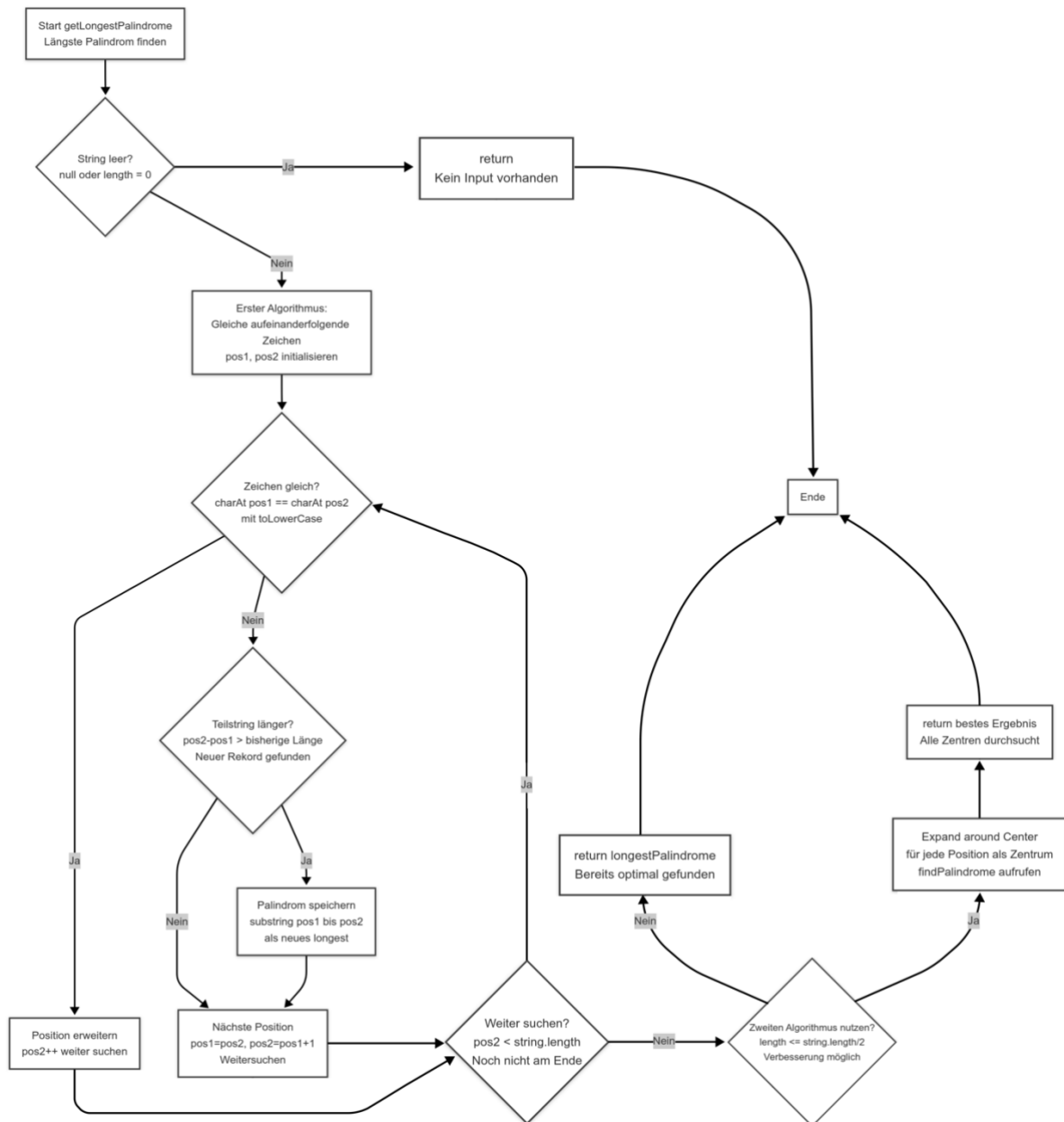


Abb. 5: Flussdiagramm der Methode getLongestPalindrome v2

### Funktionsweise:

- Diese Methode sucht die längste Sequenz identischer Symbole
- Wenn diese Sequenz >60% der Gesamtlänge ausmacht, wird sie als Ergebnis zurückgegeben.
- Andernfalls wird die vollständige Palindrom-Suche mit der *findPalindrome*-Methode durchgeführt.