NitscheSahoo - AD Praktikum für 17.10.13 – Aufgabe 2 - Matrix

Das Interface:

```
public interface Matrix {

	void insert(int i, int j, double value);

	int getM();

	int getN();

	double get(int i, int j);

	void copyFrom(Matrix source);

	int memoryUsage();

	int accessCount();

	void resetAccessCount();

	Matrix add(Matrix m);

	Matrix mul(double skalar);

	Matrix mul(Matrix factor);

	Matrix pow(int exponent);

}
```

Begründungen:
Insert: man braucht eine Methode um ein Element an eine entsprechende Position zu packen.
GetM/getN: Simple getter für Matrixdimensionen.
CopyFrom: eine exakte Kopie von einer Matrix im Speicher, da wior nicht nur eine Referenz wollen.

Aufgabe 7 – Platzaufwand der Listenimplementationen
Siehe Datei „ab2_test/Aufgabe7.java“.
Siehe Excel Tabelle Sheet „Aufgb2“.

Graphische Darstellung: Siehe Excel Tabelle. Da beide Impementationen mit den gleichen zufälligen Matrizen arbeiten, haben beide daher den gleichen Platzaufwand in den Messdaten.

Platzbedarf: Füllt man die beiden Listenimpementationen von Matrizen mithilfe der Generatormoduls, dann können wir feststellen, dass bei geringem p (also geringer Chance auf eine Zahl ungleich Null) beide Listenimplementationen tatsächlich Platz sparen. Erhöht man p um einen Faktor, so wir dauch de rPlatzbedarf um ca. den gleichen Faktor höher.

Overhead: Wir haben die Kosten eines einzigen Elements als Overhead interpretiert.

Aufgabe 8 – Platzaufwand der Listenimplementationen

Siehe Datei „ab2_test/Aufgabe7.java“.
Siehe Excel Tabelle Sheet „Aufgb2“.

Graphische Darstellung: Siehe Excel Tabelle. ListAddAverage und ArrayListMultAverage haben sehr ähnliche Werte. Deswegen sind nicht beide Linien zu erkennen. Das Diagramm ist außerdem logarithmisch.

Zeitbedarf(In Anzahl der Dereferenzierungen): Der Zeitbedarf vermehrfacht sich bei beiden Implementationen und beiden Operationen proportional zu p. Dabei ist die ArrayListImplementation ungefähr 100x effizienter.

Overhead: ?

Aufwandsabschätzung:

Arbeitszeit: 15h gemeinsam
Zeitverteilung: 60% MatrixImplementations - 40% Tests/Experiemente

Vorschau der Javadoc: