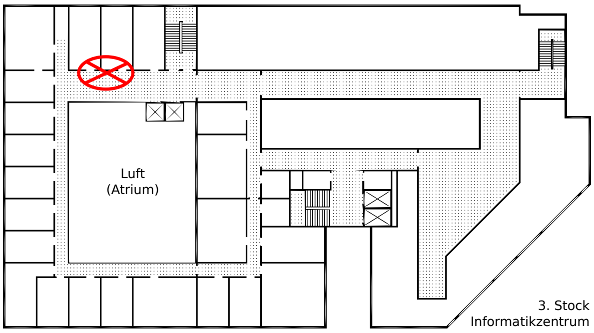
Abteilung Algorithmik **Winter 2018/19**
Institut für Betriebssysteme und Rechnerverbund
TU Braunschweig

Prof. Dr. Sándor Fekete
Dominik Krupke

Mathematische Methoden der Algorithmik
Übungsblatt 4 vom 11.12.2018

Die Abgabe der Lösungen zu Blatt 4 ist bis Dienstag, den 08.01.2019 um 13:15 Uhr im Hausaufgabenrückgabeschrank der Algorithmik möglich.

Bitte die Blätter vorne deutlich mit eigenem Namen sowie Matrikelnummer versehen und zusammenheften!



Aufgabe 1 (Implementiere Simplex): Implementiere in Python mit NumPy den Simplex-Algorithmus der für LPs in Standardform und einer gegebene Startbasis die optimale Lösung berechnet oder erkennt, dass das Problem unbeschränkt ist. Der Code kann per Mail an Alexander (ahill@ibr.cs.tu-bs.de) geschickt werden. Nutze dafür folgende Struktur:

```
import unittest

import numpy as np


class SimplexAlgorithm:
    def __init__(self, A: np.array, b: np.array, c: np.array, start_basis: list):
        """
        Initializes the simplex algorithm
        :param A: The n*m matrix (variable coefficients of the constraints)
        :param b: The constant coefficients of the constraints
        :param c: The objective function vector
        :param start_basis: The indices of the start basis. This might not be the
                            standard basis, so you will have to use the inverse to
                            initialize the tableau.
        """
        raise NotImplementedError()

    def solve(self, maximize=True) -> bool:
        """
        Starts the solution process, i.e., the simplex algorithm.
        The algorithm has to work on the tableau.
```

```
        :param maximize: Should the objective be maximized or minimized?
        :return: Has a solution been computed?
        """
        raise NotImplementedError()

    def get_solution(self) -> np.array:
        raise NotImplementedError()


class Tableau:
    """
    It is recommended but not required that you first implement
    the tableau with the necessary operations.
    """


class SimplexAlgorithmTest(unittest.TestCase):
    """
    You can add additional test cases. Just make sure that their
    names start with "test".
    IDEs such as PyCharm will allow you to directly run the tests.
    """

    def test_1(self):
        A = np.array([[0, 1, 1, 0, 0],
                      [1, 1, 0, 1, 0],
                      [1, -1, 0, 0, 1]])
        b = np.array([5, 7, 3])
        c = np.array([1, 0, 0, 0, 0])

        x = np.array([5, 2, 3, 0, 0])
        sa = SimplexAlgorithm(A=A, b=b, c=c, start_basis=[2, 3, 4])
        success = sa.solve()
        self.assertTrue(success)
        self.assertTrue((sa.get_solution()==x).all())
```

(30 Punkte)

Aufgabe 2 (Simplex-Tableau): Löse die folgende LPs mit dem Simplex-Algorithmus.

$$
\begin{array}{llrcrcrcl}
\min & & -x_1 & + & x_2 & - & 2x_3 & & \\
\text{s.t.} & - & 2x_1 & - & x_2 & + & 3x_3 & \geq & -40 \\
& & x_1 & & & + & x_3 & \leq & 25 \\
& & & & x_2 & + & 3x_3 & \leq & 30 \\
& & x_1 & , & x_2 & , & x_3 & \geq & 0
\end{array}
$$

$$\begin{array}{llllllllll}
\max & 2x_1 & + & 4x_2 & + & 3x_3 & + & x_4 & & \\
\text{s. t.} & 3x_1 & + & x_2 & + & x_3 & + & 4x_4 & \leq 12 \\
& x_1 & - & 3x_2 & + & 2x_3 & + & 3x_4 & \leq 7 \\
& 2x_1 & + & x_2 & + & 3x_3 & - & x_4 & \leq 10 \\
& & & & & x_1, & \ldots, & x_4 & \geq 0
\end{array}$$

$$\begin{array}{lllllllll}
\max & & x_1 & - & x_2 & + & 2x_3 & & \\
\text{s. t.} & - & 2x_1 & - & x_2 & + & 3x_3 & \geq & -40 \\
& & x_1 & & & + & x_3 & \leq & 25 \\
& & & & x_2 & + & 3x_3 & \leq & 30 \\
& & x_1 & , & x_2 & , & x_3 & \geq & \mathbf{1}
\end{array}$$

$$\begin{array}{llllll}
\max & x_1 & + & x_2 & & \\
\text{s. t.} & x_1 & + & x_2 & \geq & 10 \\
& -x_1 & + & 2x_2 & \geq & 10 \\
& 4x_1 & + & 2x_2 & \leq & 15 \\
& x_1 & , & x_2 & \geq & \mathbf{1}
\end{array}$$

Benutze zum Lösen der LPs die Darstellung im Tableau. Argumentiere in jedem Schritt kurz, warum du die gewählte Variable in die Basis aufnimmst, welches das neue Pivotelement ist und warum. Nutze für das dritte und vierte LP den 2-Phasen Simplex. Ihr dürft die Ausgabe eurer eigenen Implementierung nutzen insofern die Zwischenschritte erkennbar ausgegeben werden. **(20 Punkte)**

Aufgabe 3 (Reduzierte Kosten):

(a) Sei $c^T = (4, 6, 0, 0, 0)$ gegeben. Angenommen, das folgende Tableau ist ein Zwischenschritt im Simplex. Berechne die reduzierten Kosten $\overline{c}_i, 1 \leq i \leq 5$ und den aktuellen Zielfunktionswert z.

$\overline{c}_1$	$\overline{c}_2$	$\overline{c}_3$	$\overline{c}_4$	$\overline{c}_5$	$-z$
-1	1	1	0	0	11
2	0	-1	1	0	16
7	0	-5	0	1	35

(b) Sei $c^T = (3, 2, 1, 0, 0, 0)$ gegeben. Angenommen, das folgende Tableau ist ein Zwischenschritt im Simplex. Berechne die reduzierten Kosten $\overline{c}_i, 1 \leq i \leq 6$ und den aktuellen Zielfunktionswert z.

$\overline{c}_1$	$\overline{c}_2$	$\overline{c}_3$	$\overline{c}_4$	$\overline{c}_5$	$\overline{c}_6$	$-z$
1	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	0	0	$\frac{15}{2}$
0	$\frac{5}{2}$	$\frac{1}{2}$	$-\frac{1}{2}$	1	0	$\frac{90}{2}$
0	$\frac{7}{4}$	$\frac{11}{4}$	$-\frac{1}{4}$	0	1	$\frac{65}{2}$

(10 Punkte)

Frohe Weihnachten und einen guten Rutsch!