1. PNK-Übungsblatt

Hinweis: Verwenden Sie in den folgenden Aufgaben stets die Hesse'sche Normalenform

$$\mathbf{x} \cdot \mathbf{w}_0 - d = 0,$$

um eine Hyperebene darzustellen. Hierbei ist $\mathbf{x}$ ein Punkt auf der Hyperebenen, $\mathbf{w}_0$ der Normalenvektor der Hyperebene und der Betrag von d der Abstand der Hyperebene zum Koordinatenursprung.

Aufgabe 1 Vektoren und Winkel

Berechnen Sie den Betrag des Vektors sowie die Winkel zwischen Vektor und Koordinatenachsen:

a) $\mathbf{b} = (-2, 11, -10)$,

b) $\mathbf{c} = \overline{AB}$ mit $A(1, -2, -3)$ und $B(4, 2, 9)$.

Aufgabe 2 Hesse'sche Normalform

Bringen Sie folgende Lineargleichungen auf die Hesse'sche Normalform $\mathbf{r} \cdot \mathbf{n}_0 - d = 0$:

a) $-2x + 3y + 0.5 = 0$,

b) $y = mx + n$ mit $n > 0$.

Aufgabe 3 Abstände zu Geraden

a) Welchen Abstand hat $P_1(4, 3)$ von der Geraden, welche die Koordinatenachsen bei $x = \frac{10}{3}$ und $y = 2.5$ schneidet?

b) Welchen Abstand haben die Parallelen $2x - 3y = 6$ und $4x - 6y = 25$ voneinander?

Aufgabe 4 Gradientenverfahren

Suchen Sie das Minimum der Funktion $f(x_1, x_2, x_3) = -3x_1^3 + 0.5x_2 + 1.5x_3^2 + 1.5x_2^2 + 10$ mithilfe des Gradientenverfahrens. Nutzen Sie die Anfangsnäherung $(x_1, x_2, x_3) = (1, 1, 1)$ und die Schrittweite $\gamma = 0.2$.

Aufgabe 5 Boole'sche Ausdrücke

Vereinfachen Sie die folgenden Boole'schen Ausdrücke.

a) $\neg\neg\neg p$

b) $(p \vee \neg q \vee \neg p) \rightarrow q$

c) $(p \wedge \neg q \wedge \neg p) \rightarrow q$

d) $(p \vee q) \wedge (r \wedge q)$

e) $(p \wedge q) \vee (r \vee q)$

f) $\neg(\neg p \vee \neg q)$

g) $\neg(\neg p \wedge \neg q)$

h) $\neg p \rightarrow q$

i) $\neg p \rightarrow \neg q$

j) $\neg(p \wedge q) \rightarrow (p \wedge q)$

Aufgabe 6 Disjunktive Normalform

Konstruieren Sie die disjunktive Normalform der folgenden Boole'schen Ausdrücke.

a) $p \wedge (q \vee r)$

b) $(p \vee \neg q \vee \neg r) \wedge (s \vee \neg t)$

c) $(p \vee \neg q) \wedge (\neg r \vee \neg s \vee t) \wedge (u \wedge \neg v)$

Aufgabe 7 Konjunktive Normalform

Konstruieren Sie die konjunktive Normalform der folgenden Boole'schen Ausdrücke.

a) $p \vee (\neg q \wedge \neg r)$

b) $(p \wedge (\neg q \vee \neg r)) \vee (s \vee \neg t)$

c) $(p \vee \neg q) \vee ((\neg r \vee \neg s \vee t) \wedge (u \vee \neg v))$

2. PNK-Übungsblatt

Dieses Übungsblatt wird im Fuzzy-Labor G29-036 stattfinden. Bitte schicken Sie Ihre funktionierenden Programmierlösungen einen Tag vorher an `mailto:cmoewes@ovgu.de` ein, um eine Vorstellung im Fuzzy-Labor zu garantieren. Das Vorstellen eines Pseudo-Codes ist ebenfalls erlaubt. Für diese Form eines Programms bedarf es natürlich keiner Einreichung am Vortag der Übung. Viel Glück :-)

Aufgabe 8 Newton-Verfahren

a) Informieren Sie sich, z.B. auf Wikipedia, über das Newton-Verfahren.

b) Leiten Sie (sowohl graphisch als auch) rechnerisch für eine stetig differenzierbare, reelle Funktion $f : \mathbb{R} \to \mathbb{R}$ das Newton-Verfahren $x^{(i+1)} \leftarrow x^{(i)} - f(x^{(i)})/f'(x^{(i)})$ zum Finden von Nullstellen von f her.

c) Nutzen Sie das Netwon-Verfahren zum Finden der Quadratwurzel aus 5. Dafür bietet es sich an, die Beziehung $x^2 = 5 \Leftrightarrow x^2/5 = 1 \Leftrightarrow 0 = 1 - 5/x^2$ zu verwenden.

Aufgabe 9 Newton-Verfahren

Implementieren Sie das Newton-Verfahren in einer Programmiersprache Ihrer Wahl und testen Sie es, um die Quadratwurzel aus 5 zu bestimmen.

Aufgabe 10 Gradientenverfahren

Implementieren Sie das Gradientenverfahren in einer Programmiersprache Ihrer Wahl und Testen Sie es zum Finden eines Minimum für ein beliebiges Polynom 3. Ordnung.

Aufgabe 11 Delta-Regel

Implementieren Sie die Delta

a) für Batch-Lernen und

b) für Online-Lernen

in einer Programmiersprache Ihrer Wahl.

3. PNK-Übungsblatt

Aufgabe 12 Programmablaufplan

a) Informieren Sie sich über Programmablaufpläne im Internet (z.B. auf Wikipedia).

b) Entwickeln Sie ein Programmablaufplan, der den kompletten Vorgang Ihres Frühstücks beschreibt.

c) Entwickeln Sie ein Programmablaufplan, um mit dem Fahrrad von Prof. Henning Scheichs Büro auf dem Medizin-Campus zu Prof. Kruses Büro in der FIN zu kommen.

Aufgabe 13 Süßigkeiten

Ein Bonbonglas ist mit mit vielen Lakritzeschnecken und Gummibärchen gefüllt. Außerdem liegt neben dem Glas eine „Wundertüte“, die unerschöpflich viele Lakritzeschnecken enthält. Der folgende Vorgang soll nun solange ausgeführt werden, bis er sich nicht mehr wiederholen lässt:

- Entnehmen Sie zwei beliebige Süßigkeiten aus dem Glas.
- Falls beide die gleiche Geschmacksrichtung haben, essen Sie sie auf und füllen eine Lakritzeschnecke aus der nebenstehenden Tüte in das Glas.
- Haben Sie eine Lakritzeschnecke und ein Gummibärchen gezogen, dürfen Sie nur die Lakritzeschnecke naschen. Das Gummibärchen müssen Sie wieder in das Glas zurücklegen.

Folgende Fragen drängen sich auf:

a) Terminiert dieser Vorgang?

b) Wie viele Süßigkeiten verbleiben im Glas?

c) Ist der Vorgang determiniert oder deterministisch?

Hinweis: Es wird empfohlen, bei praktischen Untersuchungen des Problems die Werte für n (Anzahl der Gummibärchen) und m (Anzahl der Lakritzeschnecken) nicht zu groß zu wählen, da eine Magenverstimmung das Lösen der Aufgabe erheblich beeinträchtigt.

Aufgabe 14 Primzahlen

Beschreiben Sie einen Algorithmus, der prüft, ob eine gegebene Zahl eine Primzahl ist. Berücksichtigen Sie dabei die Eigenschaft, dass eine Primzahl nur durch eins und durch sich selbst teilbar ist. Verwenden Sie zur Beschreibung ein Struktogramm oder einen Ablaufplan.

Hinweis: Als Lüsungsansatz ist zu untersuchen, ob die Zahl durch einen ihrer Vorgänger teilbar ist.

4. PNK-Übungsblatt

Aufgabe 15 Zinseszins

Ein Kapital von 1000 Euro wird jährlich mit 5% verzinst. Die Funktion `kapital(n)` beschreibe den Kapitalwert nach n Jahren.

Implementieren Sie ein rekursives Programm zur Berechnung des Kapitels nach n Jahren.

Aufgabe 16 Zinseszins

Was leistet die folgendermaßen definierte Funktion?

```
def anzahl(element, liste):
    if len(liste) == 0:
        return 0
    else:
        if liste[0] == element:
            return (1 + anzahl(element, liste[1:]))
        else:
            return anzahl(element, liste[1:])
```

Testen Sie diese Funktion mit verschiedenen Aufrufen wie dem folgenden und erklären, wie die Ergebnisse zustande kommen.

```
>>>anzahl('b', ['a', 'b', 'd', 'a', 'b'])
...
```

Aufgabe 17 Potenzberechnung

Berechnen Sie die Potenz $pot(x, y) = x^y$ zweier natürlicher Zahlen $x > 0$ und $y \geq 0$ durch wiederholte Multiplikation. Implementieren Sie hierfür jeweils

a) einen applikativen und

b) einen imperativen

Algorithmus.

Aufgabe 18 Perfekte Zahlen

Eine Zahl n heißt perfekt oder vollkommen, wenn die Summe ihrer Teiler gleich n ist. Dabei sind die Teiler echt kleiner als n. Beispielsweise ist $6 = 1 + 2 + 3$ eine perfekte Zahl.

a) Implementieren Sie einen Algorithmus, der für eine gegebene Zahl bestimmt, ob sie perfekt ist.

b) Schreiben Sie ein Programm, das alle perfekten Zahlen bis 1000 bestimmt.

Aufgabe 19 Das Sieb des Eratosthenes

Mit Hilfe des Siebs des Eratosthenes können sehr effizient alle Primzahlen bis zu einer gegebenen Obergrenze gefunden („herausgesiebt“) werden. Stellen Sie sich eine Tafel vor, auf der alle natürlichen Zahlen bis zu einer gewählten Obergrenze angeschrieben sind. Auf dieser werden folgendermaßen alle Nicht-Primzahlen gestrichen:

Beginnen Sie mit der 2. Alle Vielfachen von 2, beginnend mit 4, sind keine Primzahlen. Streichen Sie diese Zahlen auf der Tafel aus. Nehmen Sie nun die 3. Streichen Sie alle Vielfachen von 3, beginnend mit 6. Nun die 4. Sie ist bereits von der Tafel gestrichen und kann daher übersprungen werden. (Alle Vielfachen von 4 sind auch Vielfache von 2 und daher schon gestrichen.) Streichen Sie die Vielfachen von 5, beginnend mit 10.

Indem Sie so fortfahren und am Ende noch die 1 streichen, die per Definition keine Primzahl ist, bleiben auf der Tafel nur die Primzahlen ungestrichen. Schreiben Sie ein Programm, das diesen Algorithmus benutzt, um alle Primzahlen bis 1000 zu bestimmen.

5. PNK-Übungsblatt

Aufgabe 20 Klassen, Objekte und Vererbung

Informieren Sie sich unter

- `http://de.wikipedia.org/wiki/Klasse_%28objektorientierte_Programmierung%29`,
- `http://docs.python.org/2/tutorial/classes.html` und
- `http://de.wikipedia.org/wiki/Klassendiagramm`

über die Entwicklung und Darstellung von Klassen (in Python und in UML).

a) Welche Vorteile bringen Klassen bei der Entwicklung von Software?

b) Entwickeln Sie ein Klassendiagramm für das Objekt `Lehrveranstaltung`. Berücksichtigen Sie dabei Objekte (z.B. `Professor`, `Student`, `Vorlesung`, Prüfung) und Methoden (z.B. `fuege_student_hinzu()`, `besucht_vorlesung()`).

c) Implementieren Sie die Klassen Ihres Klassendiagramms in Python. Kreieren Sie die Instanz `neuronale_netze` z.B. unter Ausnutzung von `kruse = Professor()`.

Aufgabe 21 Die Klasse der Neuronalen Netze

a) Entwickeln Sie ein Klassendiagramm für das Objekt `NeuralNetwork`. Berücksichtigen Sie dabei notwendige Objekte und Methoden.

b) Implementieren Sie Ihr Klassendiagramm in Python. Kreieren Sie die Instanz
`mlp = NeuronalesNetz('mlp', units=[2, 5, 3], training='backprop', ...)`.

c) Testen Sie Ihr Netz anhand eines beispielhaften 2-dimensionalen 3-Klassenproblems. Die Daten sollen dabei eine Stichprobe der Größe 300 umfassen, die aus von 3 unterschiedlichen zweidimensionalen Normalverteilungen gezogen wurde. Nehmen Sie an, dass die Normalverteilungen mit A, B und C bezeichnet seien und keinerlei Kovarianzen ausdrücken.