Kapitel 2

Mengenlehre

Verständnisfragen

Sachfragen

1. Wann sind zwei Mengen gleich?
2. Was ist eine Potenzmenge?
3. Was ist der Durchschnitt und die Vereinigung zweier Mengen?
4. Was ist die Differenz und das Komplement zweier Mengen?
5. Wie lautet die Summen- und die Produktregel der Kombinatorik?
6. Was ist eine Permutation?
7. Was ist eine Kombination?
8. Was ist das Pascal'sche Dreieck?
9. Wie lautet die Vandermond'sche Gleichung und wie lautet ihre Interpretation in der Kombinatorik?
10. Beschreiben Sie das Inklusions-Exklusions-Prinzip!
11. Motivieren Sie die Formel für das Inklusions-Exklusions-Prinzip!
12. Was ist ein Derangement?

Methodenfragen

1. Die Potenzmenge einer gegebenen Menge aufstellen können.
2. Eine Mengengleichheit nachweisen können.
3. Die verschiedenen Mengenoperationen durchführen können.
4. Die Rechenregeln für die Mengenoperationen anwenden können.
5. Mengenbeziehungen und Mengenaussagen durch Venn-Diagramme beschreiben können.

6. Mit Hilfe der Charakteristik Mengen auf dem Computer darstellen können und damit die Mengenoperationen durchführen können.
7. Die Summen- und Produktregel anwenden können.
8. Permutation und Kombinationen mit und ohne Wiederholung berechnen können.
9. Binomialkoeffizienten berechnen und anwenden können.
10. Modellprobleme der Kombinatorik beschreiben und identifizieren können.
11. Das Inklusions / Exklusions-Prinzip anwenden können.
12. Derangements berechnen können.

Übungsaufgaben

1. Zählen Sie die Elemente der folgenden Mengen auf: $M_1 = \{x \mid x$ ist eine reelle Zahl mit $x^2 = 1\}$, $M_2 = \{x \mid x$ ist eine natürliche Zahl kleiner als $12\}$, $M_3 = \{x \mid x$ ist das Quadrat einer ganzen Zahl und kleiner als $100\}$.

   Lösung:
   $M_1 = \{1, -1\}$, $M_2 = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$, $M_3 = \{1, 4, 9, 16, 25, 36, 49, 64, 81\}$.

2. Beschreiben Sie die folgenden Mengen mit Hilfe von Eigenschaften: $M_1 = \{0, 3, 6, 9, 12\}$, $M_2 = \{-3, -2, -1, 0, 1, 2, 3\}$, $M_3 = \{1, 3, 5, 7, 9\}$, $M_4 = \{x \mid 2, 4, 6, 8, 10\}$.

   Lösung:
   $M_1 = \{x \mid x = k \cdot 3, k = 0, 1, 2, 3, 4\}$, $M_2 = \{x \mid x \in \mathbb{Z}, |x| \leq 3\}$, $M_3 = \{x \mid x = 2k - 1, k \in \mathbb{N}, k \leq 5\}$, $M_4 = \{x \mid x = 2k, k \in \mathbb{N}, k \leq 5\}$.

3. Gegeben sind die Mengen $A = \{2, 4, 6\}$, $B = \{2, 6\}$ und $C = \{4, 6, 8\}$. Welche dieser Mengen ist eine Teilmenge einer der anderen?

   Lösung:
   Es ist $B \subset A$.

4. Weisen Sie nach, dass aus $A \subset B$ und $B \subset C$ auch $A \subset C$ folgt! Visualisieren Sie diesen Sachverhalt mit Hilfe eines Venn-Diagramms!

   Lösung:
   Es ist zu zeigen, dass für ein $x \in A$ auch $x \in C$ gilt. Wegen $A \subset B$ folgt aus $x \in A$ sofort $x \in B$. Wegen $B \subset C$ folgt die Behauptung.

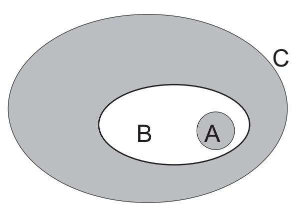

Abbildung 2.1: Venn-Diagramm zu Aufgabe 4

5. Bestimmen Sie für die Mengen $A = \{1, 2, 3, 4, 5\}$, $B = \{0, 3, 6\}$ die Mengen $A \cup B$, $A \cap B$, $A \setminus B$ und $B \setminus A$.

Lösung:
Es ist $A \cup B = \{0, 1, 2, 3, 4, 5, 6\}$, $A \cap B = \{3\}$, $A \setminus B = \{1, 2, 4, 5\}$, $B \setminus A = \{0, 6\}$.

6. Visualisieren Sie die Gleichung $(S \cup T) \setminus V = (S \setminus V) \cup (T \setminus V)$ mit Hilfe eines Venn-Diagramms und führen Sie den Nachweis der Mengengleichheit!

   Lösung:
   Ein $x \in (S \cup T) \setminus V$ liegt in S oder in T, aber nicht in V. Dann gilt aber auch $x \in S \setminus V$ oder $x \in T \setminus V$.

   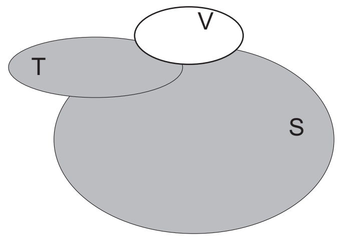

   Abbildung 2.2: Venn-Diagramm zu Aufgabe 6

   Ein $x \in (S \setminus V) \cup (T \setminus V)$ liegt in S oder in T; aber nicht in V.

7. Vereinfachen Sie den Ausdruck $(S^c \cap T^c)^c \cap (S^c \cap T^c)$!

   Lösung:
   Mit den De Morgan'schen Regeln folgt:

$$\begin{aligned}(S^c \cap T^c)^c \cap (S^c \cap T^c) &= (S \cup T) \cap (S^c \cap T^c)\\ &= (S \cup T) \cap (S \cup T)^c\\ &= \emptyset.\end{aligned}$$

8. Bestimmen Sie für $S = \{1, 2\}, T = \{a, b\}$ und $V = \{b, c\}$ die folgenden Mengen: $S \times (T \cap V), (S \times T) \cup (S \times V)$ und $(S \times T) \cap (S \times V)$!

   Lösung:
   $S \times (T \cap V) = \{(1, b), (2, b)\}$,
   $(S \times T) \cup (S \times V) = \{(1, a), (1, b), (1, c), (2, a), (2, b), (2, c)\}$
   $(S \times T) \cap (S \times V) = \{(1, b), (2, b)\}$.

9. Bestimmen Sie die Potenzmenge der folgenden Mengen: $M_1 = \{a\}$, $M_2 = \{a, b\}$, $M_3 = \{\emptyset, a, b\}$!

   Lösung:
   $\mathbb{P}(M_1) = \{\emptyset, \{a\}\}$,
   $\mathbb{P}(M_2) = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}$,
   $\mathbb{P}(M_3) = \{\emptyset, \{\emptyset\}, \{a\}, \{b\}, \{\emptyset, a\}, \{\emptyset, b\}, \{a, b\}, \{\emptyset, a, b\}\}$.

10. Implementieren Sie eine Darstellung einer Menge mit n Elementen mit Hilfe der Charakteristik und beschreiben Sie damit für zwei Teilmengen A und B die Mengen $A \cup B$, $A \cap B$, $A \setminus B$.

    Lösung:
    Hier finden Sie eine Implementierung mit Hilfe der STL-Klasse `vector`:

```
/*!
 * Die Klasse repräsentiert eine Menge und ihre Teilmengen
 * mit Hilfe von Charakteristiken.
 *
 * Die Mengen werden durch Teilmengen der natürlichen Zahlen repräsentiert,
 * beginnend bei 1.
 *
 * Eine Menge mit drei Elementen also durch {1, 2, 3}.
 * Ihre Teilmenge {2,3} durch die Charakteristik (0,1,1).
 *
 * Die Charakteristiken werden mit Hilfe der STL-Klasse vector<bool>
 * realisiert.
 * Die STL-Klasse bitset kann nicht verwendet werden, da diese Klasse
 * es nicht zulässt, die Anzahl der Elemente dynamisch zu spezifizieren.
 */
#ifndef VLSET_H
#define VLSET_H
#include <vector>

class vlSet{
public:
    //! Defaultkonstruktor; Menge mit einem Element.
    vlSet();
    //! Kopierkonstruktor
    vlSet(const vlSet&);
    //! Konstruktor mit Anzahl der Elemente; alle sind enthalten.
    vlSet(int n);
    //! Konstruktor mit Anzahl der Elemente, mit konstantem Wert
    vlSet(int n, bool val);
    //! Konstruktor mit Anzahl der Elemente und Feld
    vlSet(int n, bool*);

    //! Ist Element i in der Teilmenge enthalten?
    bool in(int i);

    //! Funktion, die die Elemente der Teilmenge auf cout ausgibt
    void print(void);

    //! Funktion für das Komplement einer Teilmenge
    vlSet komplement(void);
    //! Funktion für den Durchschnitt zweier Teilmengen
    vlSet schnitt(const vlSet);

    //! Funktion für die Vereinigun zweier Teilmengen
    vlSet vereinigung(const vlSet);

    //! Funktion für den Durchschnitt zweier Teilmengen
    vlSet differenz(const vlSet);

private:
    //! Die Charakteristik; mit einer Defaultlänge von 1
    vector<bool> chi;
    vector<int> elemente;
};
#endif // VLSET_H
```

```
#include "vlSet.h"

vlSet::vlSet()
```

```
{
    chi = vector<bool>(false);
    elemente = vector<int>(1);
}

vlSet::vlSet(const vlSet& kopie)
{
    int i, n = kopie.elemente.size();
    chi = vector<bool>(n);
    elemente = vector<int>(n);

    for (i=0; i<n; i++)
    {
        this->chi[i] = kopie.chi[i];
        this->elemente[i] = kopie.elemente[i];
    }
}

vlSet::vlSet(int n)
{
    int i;

    chi = vector<bool>(n,false);
    elemente = vector<int>(n);

    for (i=0; i<n; i++)
         elemente[i] = i+1;
}

vlSet::vlSet(int n, bool val)
{
    int i;
    chi = vector<bool>(n, val);
    elemente = vector<int>(n);

    for (i=0; i<n; i++)
         elemente[i] = i+1;
}

vlSet::vlSet(int n, bool *values)
{
    int i;

    chi = vector<bool>(n);
    elemente = vector<int>(n);

    for (i=0; i<n; i++)
         elemente[i] = i+1;

    for (i = 0; i<n; i++)
       chi[i] = values[i];
}

bool vlSet::in(int i)
{
    return chi[i];
}
```

```
void vlSet::print(void)
{
    int i, n = chi.size();

    for (i=0; i<n; i++)
    {
        if (chi[i])
           cout << elemente[i] << " ";
    }

    cout << endl;
}

vlSet vlSet::komplement(void)
{
    int i, n = elemente.size();
    vlSet ergebnis(n);

    for (i=0; i<n; i++)
         ergebnis.chi[i] = !chi[i];

    return ergebnis;
}
vlSet vlSet::schnitt(const vlSet b)
{
    int i, n = elemente.size();
    vlSet ergebnis(n);

    if (n != b.elemente.size()) return ergebnis;

    for (i=0; i<n; i++) {
        if (chi[i] && b.chi[i])
            ergebnis.chi[i] = true;
    }
    return ergebnis;
}


vlSet vlSet::vereinigung(const vlSet b)
{
    int i, n = elemente.size();
    vlSet ergebnis(n);

    if (n != b.elemente.size()) return ergebnis;

    for (i=0; i<n; i++) {
        if (chi[i] || b.chi[i])
            ergebnis.chi[i] = true;
    }
    return ergebnis;
}


vlSet vlSet::differenz(const vlSet b)
{
    int i, n = elemente.size();
    vlSet ergebnis(n);
```

```
    if (n != b.elemente.size()) return ergebnis;

    for (i=0; i<n; i++) {
        if (chi[i] && (!b.chi[i]))
            ergebnis.chi[i] = true;
    }

    return ergebnis;
}
```

Und hier ein Testprogramm:

```
#include <iostream>
#include "vlSet.h"

int main()
{
    bool ba[3] = {true, false, true},
         bb[3] = {false, false, true},
         bc[3] = {false, true, true};

    vlSet a(3, ba), b(3, bb), c(3, bc), resultat(3);

    cout << "a = " << endl;
    a.print();

    cout << "b = " << endl;
    b.print();

    cout << "c = " << endl;
    c.print();

    cout << "Das Komplement von a = {1, 3}" << endl;
    resultat= a.komplement();

    resultat.print();


    cout << "Der Durchschnitt von a und b:" << endl;
    resultat = a.schnitt(b);

    resultat.print();

    cout << "Die Vereinigung von b und c:" << endl;
    resultat = b.vereinigung(c);

    resultat.print();

    cout << "Die Differenz von c und b:" << endl;
    resultat = c.differenz(b);

    resultat.print();
    return 0;
}
```

Hier die Ausgaben des Testprogramms:

```
a =
1 3
b =
```

```
3
c =
2 3
Das Komplement von a = {1, 3}:
2
Der Durchschnitt von a und b:
3
Die Vereinigung von b und c:
2 3
Die Differenz von c und b:
2
```

11. Auf wie viele Arten können Sie aus 9 deutsch-, 3 englisch- und 11 französisch-sprachigen Büchern zwei verschiedensprachige auswählen?

    Lösung:
    Auf $9 \cdot 3 + 9 \cdot 11 + 3 \cdot 11 = 159$ Arten.

12. Auf wie viele Arten können Sie zwei Felder eines Schachbretts auswählen, die in verschiedenen Zeilen und Spalten liegen?

    Lösung:
    Ein Schachbrett hat $8 \cdot 8 = 64$ Felder. Nach der Wahl des ersten Felds, für das es 64 Möglichkeiten gibt, stehen noch $7 \cdot 7$ Felder zur Verfügung. Also gibt es $64 \cdot 49 = 3\,136$ Arten der Auswahl.

13. Auf wie viele Arten können Sie ein Skatspiel mit 32 Karten mischen?

    Lösung:
    Das Mischen entspricht einer 32-Permutation aus 32 Elementenohne Wiederholung, also ist die Antwort gegeben durch 32!.

14. Ein Passwort besteht aus zwei Buchstaben und vier Ziffern, wobei die Ziffern, aber nicht die Buchstaben mehrfach auftreten dürfen. Klein- und Großschreibung ist als signifikant anzusehen. Wie viele Passwörter können Sie bilden?

    Lösung:
    Die zwei Buchstaben auf die sechs Stellen zu verteilen entspricht $P(6, 2) = 6 \cdot 5 = 30$. Für eine getroffene Auswahl für die beiden Buchstaben gibt es dann $52 \cdot 51$ Möglichkeiten, für die Ziffern gibt es 10^4 Möglichkeiten. Insgesamt gibt es dann $30 \cdot (52 \cdot 51 + 10^4) = 382680$ Passwörter.

15. Weisen Sie die Gleichungen $k\binom{n}{k} = n\binom{n-1}{k-1}$, $\binom{n}{k} = \frac{n}{n-k}\binom{n-1}{k}$ nach!

    Lösung:
    Es ist

$$
\begin{aligned}
n\binom{n-1}{k-1} &= n\frac{(n-1)!}{(k-1)!(n-1-(k-1))!} \\
&= \frac{n!}{(k-1)!(n-k)!} \\
&= \frac{kn!}{k!(n-k)!} \\
&= k\binom{n}{k}.
\end{aligned}
$$

und

$$\frac{n}{n-k}\binom{n-1}{k} = \frac{n}{n-k}\frac{(n-1)!}{k!(n-k-1)!}$$
$$= \frac{n!}{k!(n-k)!} = \binom{n}{k}.$$

16. Weisen Sie nach, dass für eine endliche Menge die Anzahl der Teilmengen mit gerader Anzahl gleich der Anzahl der Teilmengen mit ungerader Anzahl ist!

 Lösung:
 Es gibt für ein $k \leq n$ genau $\binom{n}{k}$ Teilmengen mit k Elementen. Es ist also zu zeigen, dass

$$\sum_{k=0,\, k \text{ gerade}}^{n} \binom{n}{k} = \sum_{k=0,\, k \text{ ungerade}}^{n} \binom{n}{k}$$

 gilt. Wenn Sie die Summe auf der rechten Seite nach links bringen, erhalten Sie die Gleichung

$$\sum_{k=0}^{n} (-1)^k \binom{n}{k} = 0.$$

 Diese Formel stimmt allerdings, mit der binomischen Formel gilt

$$\sum_{k=0}^{n} (-1)^k \binom{n}{k} = (1-1)^n = 0.$$

17. Wie viele 10-stellige Dezimalzahlen enthalten die Ziffern 2 und 5, aber nicht 0, 1 oder 9?

 Lösung:
 Mit dem Inklusions-Exklusions-Prinzip folgt ähnlich der Argumentation im Buch als Lösung $7^{10} - 2 \cdot 6^{10} + 5^{10}$.

18. Eine Menge M mit 100 Elementen hat 3 Teilmengen A_1, A_2, und A_3 mit $|M_1| = 60$, $|A_2| = 65$, $|A_3| = 20$ mit $A_1 \cup A_2 \cup A_3 = M$ und $\forall i, j\ |A_i \cap A_j| = 20$. Bestimmen sie $|A_1 \cap A_2 \cap A_3|$.

 Lösung:
 Laut Aufgabentext ist $N_2 = |A_i \cap A_j| = 20$.

 Es ist $M = A_1^c \cup A_2^c \cup A_3^c$. Die Mächtigkeit von $A_1^c \cup A_2^c \cup A_3^c$ kann mit Hilfe des Inklusions-Exklusions-Prinzip berechnet werden als

$$100 = |A_1^c \cup A_2^c \cup A_3^c| = |A_1| + |A_2| + |A_3| - \binom{3}{2} N_2 + N_3.$$

 Gesucht ist N_3. Durch Auflösen ergibt sich

$$N_3 = 100 - (60 + 65 + 20) + 3 \cdot 20 = 100 - 145 + 60 = 15.$$

19. Implementieren Sie eine Funktion, die für eine natürliche Zahl n die Anzahl der Derangements D_n berechnet, und vergleichen Sie Ihre Ergebnisse mit den Werten aus Abschnitt 2.4!

 Lösung:
 Man könnte die Implementierung einfach mit der im Buch angegebenen Formel und mit Hilfe der Fakultät durchführen. Aber wenn Sie $n!$ mit jedem Summanden multiplizieren, dann erkennt man, dass

$$D_n = (-1)^n + \sum_{k=1}^{n-2} (-1)^{n-k} \prod_{j=0}^{k-1} (n-j)$$

gilt.

Dies führt zu einer Implementierung, die keine Fakultät benötigt. Das Produkt kann bei jedem Summanden mitberechnet werden.

```
long derangement(long n)
{
        long k, sign, faktor, result;

    if (n % 2 == 0)
         sign = 1;
    else
             sign = -1;
    result = sign;
    faktor = n;
    for (k=1; k<n-1; k++) {
        sign = -sign;
        result += sign*faktor;
        faktor *= n-k;
    }
    return result;
}
```

Und hier die Ausgabe für $n = 6$:

```
------------------------------------------
 Lösung der Aufgabe 19 zu Kapitel 2
------------------------------------------
D_n fuer n=6 ist: 265
```

Eine Java-Implementierung:

```
private static long derangement(long n) {
     long k;
     long result, sign, faktor;

     if (n % 2 == 0)
        sign = 1;
     else
        sign = -1;
     result = sign;
     faktor = n;
     for (k=1; k<n-1; k++) {
        sign = -sign;
        result += sign*faktor;
        faktor *= n-k;
     }
     return result;
}
```