DATEN UND CODIERUNG (3.1.1.1)

UNTERRICHTSVERLAUF

B. Berchtold – E-Mail: birgit.berchtold@gmx.net – April 2018

Inhaltsverzeichnis

1. ÜBERSICHT

Der in Klasse 7 durchgeführte Unterrichtsgang zum Thema „Daten und Codierung“ wird in Klasse 8 fortgesetzt. Während in Klasse 7 der Schwerpunkt auf dem neuen Begriff des Codierens und dem Kennenlernen verschiedener Anwendungen auch aus dem Alltag wie beispielsweise KFZ-Kennzeichen, Barcode, Morsecode oder Textcodierung lag, geht es in Klasse 8 schwerpunktmäßig um die Fehlererkennung und Fehlerkorrektur und der damit verbundenen Redundanzen beim Codieren. Dabei bleibt der Fokus auf der für die Rechner wichtigen binären Codierung.

Nachdem die Bedeutung und Relevanz der Fehlererkennung und Fehlerkorrektur anhand von alltäglichen Beispielen (verschmutze Barcodes oder QR-Codes, Kratzer auf einer CD,…) erkannt wurde, lernen die Schülerinnen und Schüler unterschiedliche Prüfverfahren wie Prüfbit und Prüfsumme zur Fehlererkennung kennen. Dabei soll auch auch immer die Güte eines Prüfverfahrens bewertet werden.

Im nächsten Schritt wird die Frage gestellt, ob jeder erkannte Fehler auch gleichzeitig korrigiert werden kann. Hierbei werden die Unterschiede der Begriffe Fehlererkennung und Fehlerkorrektur hervorgehoben. Wichtig dabei ist die Erkenntnis, dass nicht jeder fehlererkennende Code auch fehlerkorrigierend ist. Insbesondere wird hierbei auf 1-bit-, 2-bit- und 3-bit-Fehler eingegangen.

Darüber hinaus sollen die Schülerinnen und Schüler die Güte einer Codierung hinsichtlich der Fehlererkennung und Fehlerkorrektur bewerten können. Die Hamming-Distanz, die nicht Bestandteil des Bildungsplans ist, kann ergänzend als ein Gütekriterium hinzugenommen werden. Anhand von einfachen Beispielen sollen sie entscheiden, ob eine fehlerfreie, aber teure, Übertragung sinnvoll ist oder ob man kleinere Fehler tolerieren kann.

Daran anknüpfend wird schließlich in Klasse 9 schwerpunktmäßig die Datenkompression betrachtet.

2. FACHLICHER HINTERGRUND

Ein Code ist eine Vorschrift, die eindeutig die Zeichen eines Zeichenvorrats (Urbildmenge) den Zeichen eines anderen Zeichenvorrats (Bildmenge) zuordnet. Zeichen können hierbei Signale oder Symbole sein. Der Zeichenvorrat, auch Alphabet genannt, ist die Menge der zulässigen Zeichen.

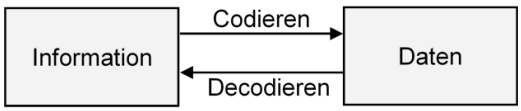
Beispiele für Alphabete:

- {A, B, C, …, Y, Z} — gewöhnliches Alphabet
- {0,1, 2,…, 8, 9} — Dezimalziffern
- alphanumerischer Zeichenvorrat — Alphabet, das die Buchstaben des gewöhnlichen Alphabets und die Ziffern 0 bis 9 enthält.
- Zeichensatz des ASCII-Codes
- { · , −, Pause} — Zeichen des Morse-Codes
- {0, 1} — binäres Alphabet
- ...

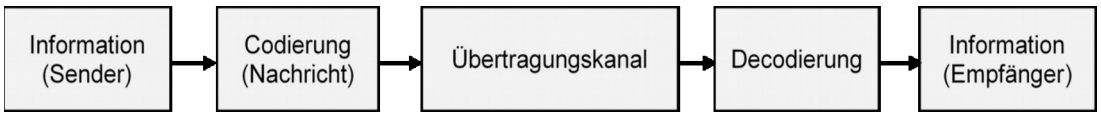
Beispiele für Codierungen:

- KFZ-Kennzeichen
- Strichcode auf Verpackungen von Waren (EAN: European Article Number)
- ISBN-Code bei Büchern
- Morsecode
- ASCII-Code
- Flaggencode
- Fliegeralphabet
- …

Je nachdem, welchem Zweck eine Codierung dient, findet man in der Literatur unterschiedliche Begriffe für die codierte Information. Dient die Codierung dem Zweck der Speicherung oder Berechnung, spricht man von Daten.



Dient die Codierung dem Zweck der Übertragung von Informationen, wird der Begriff *Nachricht* verwendet. In diesem Fall kommt, wie in der folgenden Abbildung dargestellt, der Übertragungskanal hinzu.



Die Übertragung kann ggf. verschlüsselt erfolgen (vgl. dazu 3.1.1.4 Informationsgesellschaft und Datensicherheit), worauf hier jedoch nicht eingegangen wird. Der Zweck der Codierung für Datenkompressionen wird im darauf folgenden Schuljahr behandelt (3.2.1.1 Daten und Codierung, IMP9).

Hinweis: Die Begriffe Codieren und Verschlüsseln bzw. Decodieren und Entschlüsseln sollten nicht synonym verwendet werden. In der Codierungstheorie werden ausschließlich die Begriffe Codieren und Decodieren verwendet.

Im weiteren Verlauf wird nicht immer zwischen Daten und Nachricht unterschieden. In diesem Fall wird von der codierten Information gesprochen.

Es stellt sich nun die für diese Einheit zentrale Frage: Kann man erkennen, ob sich bei einer codierten Information Fehler eingeschlichen haben? Und wenn ja, lassen sich diese Fehler auch korrigieren?

Bei der Nachrichtenübertragung ist eine grundsätzliche Idee, dass die Nachricht immer zweimal über den Übertragungskanal geschickt wird und der Empfänger die ankommenden Nachrichten vergleicht. Sind sie nicht identisch, kann der Empfänger die Nachricht noch einmal anfordern. Dass dies nicht die optimale Lösung ist, liegt auf der Hand.

Einfache Verfahren aus der Praxis sind beispielsweise das Bilden einer Quersumme als zusätzliches Prüfbit, Paritätsbits oder Prüfsummenverfahren. Dadurch werden Redundanzen erzeugt, die es ermöglichen, Fehler zu finden oder sie ggf. sogar zu korrigieren. Redundant ist der Teil einer Nachricht, der keine Information enthält.

3. WIEDERHOLUNG AUS KLASSE 7

Das in Klasse 7 erworbene Wissen zum Thema *Daten und Codierung* kann innerhalb der Klasse beispielsweise mit einem (Online-) Quiz wiederholt werden. Geeignet hierfür sind zum Beispiel *kahoot!* (kahoot.com, letzter Aufruf: 31.5.2018) oder LearningApps (learningapps.org, letzter Aufruf: 31.5.2018). Dazu können Fragen zum Binärsystem, zu Speichergrößen oder zu den in Klasse 7 durchgenommenen Codierungen gestellt werden.

Bei einem *kahoot!*-Quiz können mögliche Fragen und Antworten (A1-A4) wie folgt sein:

	A1	A2	A3	A4
Wie viele Ziffern kommen im Binärsystem vor?	1	2	10	keine
Wandle die Dezimalzahl 13 in das Binärsystem um.	1101 b	1110 b	1011 b	1100 b
Wandle die Binärzahl 10101 b in das Dezimalsystem um.	19	23	21	17
Addiere 1010 b mit 0011 b.	1011 b	1101 b	1111 b	1110 b
Wie viele Bits hat ein Byte?	2	8	10	16
Wie viele Kilobyte hat 1 Gigabyte?	0,000001	1.000	0,001	1.000.000
Wofür steht die Abkürzung TB?	Terrabyte	Terabit	Terabyte	Tausend Byte
Welchen Präfix für Speichergrößen gibt es NICHT?	Exa	Yotta	Zetta	Hexa
Mit welcher Codierung lässt sich „INFORMATIK“ NICHT codieren?	ASCII-Code	Morse-Code	Braille-Schrift	Eiercode
Wobei handelt es sich um einen längenvariablen Code?	Morse-Code	Eiercode	ASCII-Code	EAN-13

Hinweis: Diese Fragen sind nicht vollständig und können bzw. sollen beliebig angepasst und ergänzt werden.

4. PRÜFSUMMENVERFAHREN UND PARITÄTSPRÜFUNG

Nach der Wiederholung bekannter Codierungen kann in einem Unterrichtsgespräch diskutiert werden, was passieren könnte, wenn ein Code beispielsweise fehlerhaft (Kratzer auf einer CD) oder nicht mehr erkennbar bzw. verschmutzt ist (Barcode). Welche Reaktion kann man an einer Supermarktkasse erwarten, wenn der Scanner den EAN-13-Code nicht korrekt lesen kann? In der Regel muss der/die Kassierer/in den Code von Hand eingeben. Kann man sich dabei vertippen? Kann es zu Zahlendrehern kommen? Wird dann ein falsches Produkt eingelesen wird? Wie kann dies verhindert werden? Idealerweise kommen die Schülerinnen und Schüler auf die Idee, eine Quersumme zu bilden. Dieses ist ein sehr einfaches Verfahren, hat aber den Nachteil, dass Zahlendreher nicht erkannt werden können. Man kann aber diese Idee aufgreifen und zum Arbeitsblatt `01_duc_ab_pruefsumme_pruefbit` überleiten.

4.1. Prüfsummenverfahren am Beispiel der EAN-13-Codierung

Die Schülerinnen und Schüler sollen hierbei anhand der gegebenen Beispiele zweier EAN-13-Codes die Berechnung der Prüfsumme selbst erkennen.

Die Prüfsumme eines EAN-13-Codes wird wie folgt berechnet: Jede der ersten 12 Ziffern wird abwechselnd mit dem Faktor 1 und dem Faktor 3 multipliziert. Anschließend wird aufsummiert und von diesem Ergebnis zur nächsten Zehnerzahl ergänzt. Dies ist die sogenannte Prüfziffer. Eine Modulo-Rechnung wird an dieser Stelle nicht erwartet.

Da der EAN-13-Code auf fast allen Schulmaterialien wie Bleistifte, Radiergummis, Schulhefte,… zu finden ist, können die Schülerinnen und Schüler an dieser Stelle experimentieren und prüfen. Beispielsweise kann auf der Internetseite *http://www.pruefziffer.de/eantest.php4* (letzter Aufruf: 23.4.2018) ein EAN-13-Code validiert und zu den ersten 12 Ziffern eines EAN-13-Codes die fehlende Prüfziffer berechnet werden.

Interessant ist auch, dass bei der UIC-Kennzeichnung der Triebfahrzeuge auch ein Prüfsummenverfahren eingesetzt wird. Allerdings wird hierbei abwechselnd mit den Faktoren 1 und 2 multipliziert. Von den gewichteten Summanden werden nun allerdings ihre Ziffern addiert.

Für die Schülerinnen und Schüler kann auch die Codierung der Striche beim EAN-Strichcode interessant sein. Hierbei wird jede Ziffer durch ein Modul bestehend aus sieben vertikalen gleich großen Abschnitten dargestellt. Die Abschnitte werden nach einer Codierungsvorschrift entsprechend schwarz und weiß eingefärbt.

Güte der Verfahren mit Beispielen: Beim einfachen Quersummenverfahren wird zur Berechnung der Prüfziffer zu einer Ziffernfolge zunächst die Quersumme der Ziffern dieser Ziffernfolge gebildet und anschließend zur nächsten Zehnerzahl ergänzt.

Beispiel: Der Ziffernfolge 12345 wird demnach die Prüfziffer 5 angehängt, da 1+2+3+4+5=15 ist und 15 mit 5 bis zur nächsten Zehnerzahl 20 ergänzt wird. Somit lautet die Ziffernfolge inklusive der Prüfziffer 123455.

Da aber jede andere Reihenfolge dieser Ziffern wie 54321 oder 15243 auch zur gleichen Prüfziffer 5 führt, können mit diesem Verfahren zwar Einzelfehler, die sehr häufig vorkommen, erkannt werden, jedoch keine Zahlendreher, die genauso oft als Fehler vorkommen.

Um auch Zahlendreher als Fehler erkennen zu können, werden im einfachen Fall die Ziffern abwechselnd mit zwei unterschiedlichen Faktoren gewichtet. Anschließend wird wieder aufsummiert und zur nächsten Zehnerzahl ergänzt. Somit wird erreicht, dass die meisten

Zahlendreher aufgedeckt werden können. Jedoch gibt es auch hier Einzelfälle, bei denen die Fehlererkennung dennoch nicht gelingt.

Beispiel: Die zu betrachtende Ziffernfolge lautet 1678 und es sollen die Gewichte 3 und 1 verwendet werden. Die gewichtete Quersumme beträgt 3 · 1 + 1 · 6 + 3 · 7 + 1 · 8 = 38. Demnach ist die Prüfziffer 2. Es werden nun die ersten beiden Ziffern der Ziffernfolge vertauscht, sodass nun die fehlerhafte Ziffernfolge 6178 lautet. Die gewichtete Quersumme beträgt in diesem Fall 3 · 6 + 1 · 1 + 3 · 7 + 1 · 8 = 48 und somit ist die Prüfziffer wieder 2 und der Fehler wird nicht erkannt.

Mithilfe zahlentheoretischer Kenntnisse können derartige Verfahren noch weiter verbessert werden.

4.2. Paritätsprüfung (Paritätsbit)

Die Paritätsprüfung ist ein Verfahren zur Fehlererkennung, das auf binären Codes basiert. Hierbei wird zu einer zu übertragenden Bitfolge als Prüfbit ein sogenanntes Paritätsbit angehängt. Ist eine gerade Parität vereinbart, so muss nach Anhängen des Paritätsbits die Summe aller Bits (inklusive des Paritätsbits) gerade sein. Ist eine ungerade Parität vereinbart, gilt entsprechend, dass die Summe aller Bits ungerade sein muss. Entspricht die Bitsumme nicht der vorgegebenen Parität, so liegt ein Übertragungsfehler vor. Sind allerdings zwei Bits fehlerhaft, so kann dies nicht zuverlässig erkannt werden. Umgekehrt kann man daher auch nicht ausschließen, dass eine scheinbar korrekt übertragene Bitfolge fehlerfrei ist. Eine Diskussion darüber kann bei Aufgabe 7 und 8 geführt werden.

5. DAS XO-SPIEL

Das XO-Spiel geht auf die Idee von Jens Gallenbacher[1] zurück. Das, was hierbei als „Zauberei in der Informatik“ gezeigt wird, beinhaltet eine sehr schöne Erklärung der Begriffe „Redundanz von Informationen“, „Fehlerkorrekur“ und „Fehlererkennung“. Man benötigt für dieses Spiel lediglich beidseitig beschriftete Kärtchen, wobei eine Seite eines Kärtchens mit dem Symbol „X“ und die andere Seite mit dem Symbol „O“ gekennzeichnet ist. Eine Kopiervorlage findet man unter `02_kopiervorlagen/02_duc_kopiervorlage_xo_spiel.docx`. Idealerweise werden die X- und O-Kärtchen auf unterschiedlich farbigem Papier gedruckt und anschließend zusammen mit Vorder- und Rückseite laminiert. Auch im Buch von Jens Gallenbacher, Abenteuer Informatik, ist im Anhang eine kartonierte Vorlage zu finden. Benötigt werden mindestens 36 solcher Kärtchen. Hat man mehr dieser Kärtchen, lässt sich die „Zauberei“ etwas überzeugender durchführen. In der Arbeitsphase bekommt jede Gruppe ein solches Kartenset. Zum Vorführen vor der gesamten Gruppe eignen sich auch Wendemagnete, die an Tafeln haften bleiben. Diese können im Internet bestellt werden.

5.1 Ablauf des Spiels

X	X	O	X	O
X	X	X	X	X
O	X	O	O	X
X	X	O	X	X
O	O	O	O	O

Abb. 1: 5x5-Ausgangs-muster

Der „Zauberer“ (beim ersten Vorführen dieses Spiels ist dies die Lehrkraft) gibt an, magische Fähigkeiten zu besitzen. Um dies demonstrieren zu können, bittet der Zauberer ein oder mehrere Schülerinnen und Schüler mithilfe der Kärtchen ein beliebiges, möglichst kompliziertes 5x5-Muster zu legen. Dabei sieht er absichtlich nicht genau zu, was gelegt wird. Er behauptet, er könne anschließend mithilfe seiner magischen Kräfte jedes geheim umgedrehte

X	X	O	X	O	X
X	X	X	X	X	X
O	X	O	O	X	O
X	X	O	X	X	O
O	O	O	O	O	O
X	O	X	X	X	O

Abb. 2: 6x6-Muster

1 Jens Gallenbacher, Abenteuer Informatik, Kapitel 11 „InformaGik“

Kärtchen finden, ohne sich das Muster vorher einprägen zu müssen. Zum Beispiel könnte das von den Schülerinnnen und Schülern gelegte Muster wie in Abb. 1 aussehen. Der Zauberer beschließt spontan, den Schwierigkeitsgrad des Spiels zu erhöhen und – damit es angeblich etwas schneller geht – ergänzt selber rasch und scheinbar zufällig das gelegte Muster zu einem 6x6-Muster. Beispielsweise wird jeweils rechts und unten eine Spalte bzw. Zeile ergänzt (vgl. Abb. 2). Der Zauberer bittet nun, dass heimlich eine Karte umgedreht werden soll. Er selber dreht sich dabei mit dem Rücken zum Spiel oder verlässt kurz den Raum. Dann kehrt er zurück zum Spiel und lässt seine Magie spielen – und findet die umgedrehte Karte! Um die Glaubwürdigkeit des Zauberers zu unterstreichen, kann das Spiel wiederholt werden.

5.2 Fehlererkennung und Fehlerkorrektur (1-bit, 2-bit, 3-bit-Fehler)

Gemeinsam oder in kleineren Gruppen soll nun der Zaubertrick gelüftet werden. Die Idee, die hinter diesem Trick steckt, ist, dass das Legen der scheinbar zufälligen Karten zu einem 6x6-Muster einer strikten Regel folgt. Jede Zeile und jede Spalte wird so ergänzt, dass die Anzahl der „X“-Karten und die Anzahl der „O“-Karten in allen Zeilen und Spalten stets gerade ist.

Die XO-Karten können natürlich auch binär interpretiert werden. Ein X entspricht dabei einer 1, ein O entspricht einer 0. Es liegt nun die Frage auf der Hand, ob sich der Informationsgehalt der zunächst gelegten 25 Karten nach dem Ergänzen zu einem 6x6-Muster mit 36 Karten auch erhöht hat. Die Antwort ist: nein. Denn man kann jederzeit die zusätzlichen Karten wieder wegnehmen und die ursprüngliche Information bleibt erhalten. Genauso kann auch jederzeit das Muster nach der vorgegeben Regel wieder ergänzt werden. Man kann feststellen, dass sogar jede beliebige Spalte und jede beliebige Zeile entfernt werden kann, um daraus wieder eindeutig das Muster zu ergänzen. Hierbei lässt sich der Begriff der „Redundanz“ erklären. Die zusätzlichen Karten (in der Informatik entsprechen diese den Bits) ändern den Informationsgehalt nicht.

Damit stellt sich aber nun die Frage, wozu man dann diese zusätzlichen Karten (Bits) benötigt. Dieser Fragestellung soll nun im Folgenden und mithilfe des Arbeitsblattes (`02_duc_ab_xo_spiel.odt`) nachgegangen werden.

In Partnerarbeit oder in Kleingruppen kann das Arbeitsblatt von den Schülerinnen und Schülern erarbeitet werden. Nachdem in den Gruppen das Spiel selbstständig und mit unterschiedlichen 5x5-Ausgangsmustern durchgeführt wurde, sollen nun zu einem nicht mehr zu ändernden 6x6-Muster entsprechende Fragen gelöst werden.

Das Ziel dabei ist, dass die Schülerinnen und Schüler feststellen, dass mithilfe der Redundanzen manche Fehler erkannt werden können. In bestimmten Fällen kann ein Fehler nicht nur erkannt, sondern sogar korrigiert werden. Auf den Arbeitsblättern sollen die Schülerinnen und Schüler diejenigen Positionen in den vorgedruckten Rastern farbig markieren, die jeweils für die Lösung der Aufgabe relevant sind, vergleichbar mit den Abbildungen 3 bis 7.

Der erste Fall entspricht dem oben durchgeführten Zauberspiel. Egal wie das 5x5-Ausgangsmuster gelegt und nach der vorgegeben Regel zu einem 6x6-Muster ergänzt wurde, es lässt sich an jeder Position eine falsch gelegte Karte finden und durch Umdrehen korrigieren. Das bedeutet, dass hierbei 1-bit-Fehler stets erkannt und korrigiert werden können.

1-bit Fehler sind fehlererkennend und fehlerkorrigierend.

Im zweiten Fall soll überprüft werden, ob sich auch 2-bit-Fehler, also das Umdrehen von zwei Karten, ebenfalls finden und korrigieren lassen. Bei dem 6x6-Muster aus Abb. 2 werden nun hierfür zwei beliebige Karten umgedreht. Diese sind in der Abb. 3 mit einem roten Rahmen markiert. Man kann nun feststellen, dass sowohl in der 2. und 5. Zeile als auch in der 2. und 4.

Spalte die Anzahl der X-Kärtchen und die Anzahl der O-Kärtchen nicht mehr gerade ist. Man weiß nun also, dass in der Tat zwei Fehler vorliegen. Das bedeutet, dass Fehlererkennung bei zwei falschen Bits möglich ist. Lässt sich dieser 2-bit-Fehler aber auch korrigieren? Dies wird in der Abb. 4 verdeutlicht. Es gibt nämlich zwei mögliche Paare von Bits, die für eine Korrektur infrage kommen, sodass nach deren Umdrehen die geforderte Regel wieder eingehalten wird: das rote Pärchen und das gelbe Pärchen. Die vier Bits bilden zusammen die Ecken eines Rechtecks. Somit kann es also passieren, dass bei einem 2-bit-Fehler falsch korrigiert und damit alles verschlimmert wird.

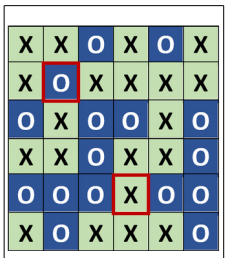


Abb. 3: 2-bit-Fehler

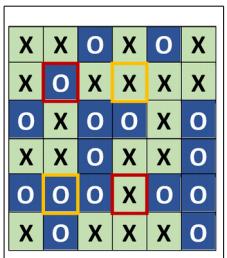


Abb. 4: 2-bit-Fehler

Werden zwei direkt nebeneinander liegende Kärtchen umgedreht, beispielsweise in einer Zeile, so kann man zwar in den beiden betreffenden Spalten erkennen, dass zwei Fehler vorliegen müssen, jedoch bleibt die Anzahl der X- und O-Kärtchen in der betroffenen Zeile gerade, so dass jede Zeile für eine Korrektur infrage kommen kann.

2-bit-Fehler sind fehlererkennend, jedoch nicht fehlerkorrigierend.

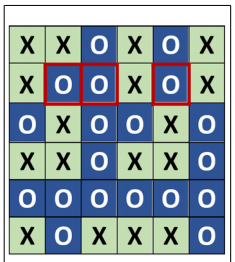


Abb. 5: 3-bit-Fehler

Im dritten Fall beschäftigt man sich mit 3-bit-Fehlern. Es können wieder verschiedene Szenarien durchgespielt werden, die sich darin unterscheiden, wo die fehlerhaften Bits im Muster verteilt sind. Ein Szenario wird in den Abbildungen 5 und 6 gezeigt. Wieder sind die fehlerhaften Bits rot markiert. In den Spalten 2, 3 und 5 sowie in der Zeile 2 wird die Regel verletzt. Eine Vermutung könnte sein, dass somit die Fehler gefunden und korrigiert werden können. Jedoch zeigt die Abbildung 6 ebenfalls ein fehlerhaftes Muster, wieder in den Spalten 2, 3 und 5 und in der Zeile 2. Offensichtlich ist dies aber ein anderer Fehler.

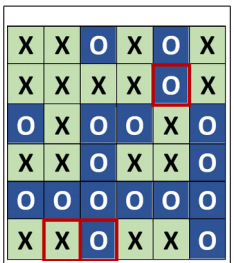


Abb. 6: 3-bit-Fehler

Somit ist klar, dass dieses fehlerhafte Muster nicht korrigiert werden kann. Der 3-bit-Fehler ist in diesem Fall erkennend, aber nicht korrigierend.

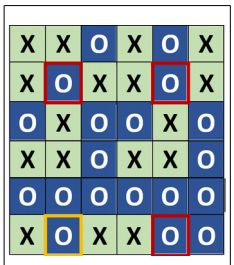


Abb. 7: 3-bit-Fehler

Noch viel dramatischer zeigt sich das Szenario, das in Abb. 7 gezeigt wird. Die drei Fehler liegen so, dass sie drei Ecken eines Rechtecks bilden. Hier liegt zwar ein 3-bit-Fehler vor, er wird jedoch fälschlicherweise als 1-bit-Fehler erkannt und somit auch entsprechend falsch korrigiert (gelb umrandet). In diesem Fall wird der 3-bit-Fehler als solcher gar nicht erkannt.

Zusammengefasst kann man daher sagen:

- Will man Fehlerkorrektur betreiben, so können 1-bit-Fehler erkannt und korrigiert werden, 2-bit-Fehler können hingegen nur erkannt werden.
- Nur wenn man auf die Fehlerkorrektur verzichtet, ist es möglich bis zu drei Fehler zu erkennen.

Als Zusatzaufgabe im Rahmen der Binnendifferenzierung kann überlegt werden, wie man mithilfe zusätzlicher Redundanzen die Zahl der korrigierbaren bzw. erkennbaren Fehler steigern könnte.

6. DAS SENDER-EMPFÄNGER-SPIEL

Beim Sender-Empfänger-Spiel[2] simulieren die Schülerinnen und Schüler das Übertragen eines binär codierten Textes, wobei sich auf dem Übertragungsweg, der sogenannten Leitung, Fehler einschleichen sollen. Anhand zweier verschiedener Codetabellen wird die Güte eines Codes hinsichtlich der Fehlerkorrektur untersucht und beurteilt (zugehöriges Arbeitsblatt: `03_duc_ab_sender_empfaenger_spiel.odt`). Die Untersuchung der Codes führt zur Hamming-Distanz, die auf Seite 4 des Arbeitsblattes behandelt wird.

Hinweis: Die Hamming-Distanz ist nicht Bestandteil des Bildungsplans und kann problemlos gestrichen werden. Sie wird hier als mögliches Zusatzmaterial aufgeführt.

Abschließend zur Unterrichtseinheit „Daten und Codieren“ können die Schülerinnen und Schüler die Vor- und Nachteile der Fehlerkorrektur auf unterschiedliche Situationen übertragen.

Für das Sender-Empfänger-Spiel wird die Klasse in Gruppen aufgeteilt. Dabei besteht eine Gruppe aus drei Positionen: Sender  [3], Leitung  , Empfänger  . Die Position Leitung wird doppelt besetzt.

Es wird empfohlen, das Spiel gut vorzubereiten und es den Schülerinnen und Schülern ausführlich zu erklären, damit ein reibungsloser Ablauf auch bei größeren Schülergruppen gewährleistet ist.

6.1 Ablauf des Spiels

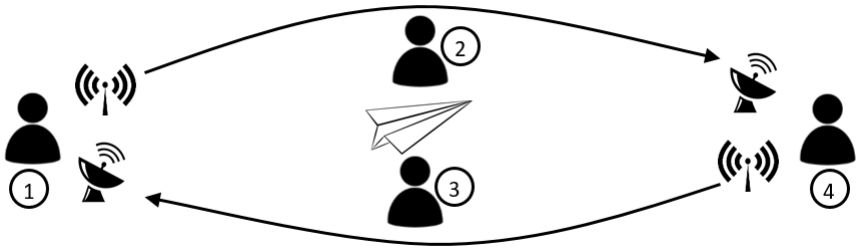
Alle Schülerinnen und Schüler bekommen die folgende Tabelle (vgl. Arbeitsblatt, Seite 2), in die sie ihre Eintragungen machen sollen. In Zeile 0 ist bereits ein Beispiel zur Erläuterung eingetragen.

	Sender-Information	Sender-Bitschlange	Empfänger-Bitschlange	Empfänger-Information	Fehler
Wer?					
0	D A C H	011 000 010 111	011 000 010 **100**	D A C E	0 0 0 2
1					
2					
3					
4					
5					
6					

2 In Anlehnung an Jens Gallenbacher, Abenteuer Informatik.

3 Alle Cliparts in diesem Kapitel sind der Seite openclipart.org (23.04.2018) entnommen. (*https://openclipart.org/unlimited-commercial-use-clipart* – All Clipart on Openclipart are available for unlimited commercial use. That means you may use the clipart commercially, for education, for church, for school, for your job, or even to manufacture products globally.)

Die Positionen 1-4 innerhalb einer Gruppe sind wie folgt besetzt:



Des Weiteren werden zwei Codetabellen „Alpha“ und „Beta“ benötigt, die als auch Vorlage unter `03_duc_kopiervorlage_Codetabellen.odt` zu finden sind:

Code Alpha	
Information	**Codewort**
A	000
B	001
C	010
D	011
E	100
F	101
G	110
H	111

Code Beta	
Information	**Codewort**
A	000000
B	000111
C	011001
D	011110
E	101010
F	101101
G	110011
H	110100

Jeweils die Positionen 1 und 4 (Sender und Empfänger) bekommen zunächst den Code Alpha und im zweiten Teil des Spiels den Code Beta.

Die Positionen 2 und 3 (Leitung) benötigen je einen Würfel, mit dem zufällige Fehler simuliert werden sollen. Wahlweise kann anstatt eines Würfels auch die Random-Funktion eines WTRs verwendet werden.

Der Sender beginnt in der Zeile 1 und notiert in der ersten Spalte „Sender-Information“ ein Wort (sinnvoll oder nicht) mit vier Buchstaben. In der zweiten Spalte „Sender-Bitschlange“ codiert er das Wort buchstabenweise entsprechend Codetabelle Alpha.

Der Sender übergibt das Blatt an die Leitung. Nun wird für jedes Bit der Sender-Bitschlange gewürfelt:

- Bei einer 6 wird statt einer 0 eine 1 bzw. statt einer 1 eine 0 eingetragen.
- Bei einer 1-5 wird das Bit unverändert übernommen.

Diese ggf. geänderte Bitfolge wird von der Leitung in die Spalte „Empfänger-Bitschlange“ eingetragen. Anschließend wird das Arbeitsblatt links der grauen Spalte gefaltet, sodass die drei rechten Spalten nach oben liegen. Die Leitung übergibt das gefaltete Arbeitsblatt an den Empfänger weiter.

Der Empfänger hat nun die Aufgabe, die Bitschlange zu decodieren und das empfangene Wort in die Spalte „Empfänger-Information“ der Tabelle einzutragen. Dabei soll in folgenden Schritten vorgegangen werden:

1. Ist das Decodieren eindeutig möglich, dann ist alles prima.
2. Ist das Decodieren nicht eindeutig möglich, dann soll „bestmöglich“ decodiert werden. Das bedeutet, dass derjenige Buchstabe gewählt werden soll, der sich an den wenigsten Stellen (Bits) vom Code unterscheidet. Gibt es hierbei zwei Möglichkeiten, soll zufällig entschieden werden.

Anschließend wird innerhalb der Gruppe das Geheimnis gelüftet und gezählt, wie viele Fehler bei der Übertragung gemacht wurden. Dies wird in die letzte Spalte eingetragen.

Um am Ende des gesamten Spiels für das Zusammentragen und Auswerten im Hinblick auf die Güte eines Codes genügend Ergebnisse zu sammeln, kann diese Runde mit der Codetabelle Alpha noch einmal wiederholt werden. Dazu tauschen die Positionen 1 und 2 sowie 3 und 4 ihre Rollen und tragen ein neues Wort in Zeile 2 ein.

Zeile 3 dient lediglich als Puffer für besonders schnelle Gruppen.

Wenn auch auch diese Runde vorbei ist, wird obiges analog mit der zweiten Codetabelle Beta durchgeführt.

6.2 Güte eines Codes

Nachdem alle Fehlerzahlen in der letzten Spalte eingetragen wurden, werden nun in der gesamten Klasse die Ergebnisse gesammelt und in die Tabelle auf Seite 5 des Arbeitsblattes eingetragen. Dabei wird deutlich, dass bei gleicher Anzahl versendeter Wörter sich die Positionen „korrekt erkannte Buchstaben“ und „versendete Signale (Bits)“ bei den beiden Codes Alpha und Beta unterscheiden.

	Code Alpha	Code Beta
Versendete Buchstaben		
Korrekt erkannte Buchstaben		
Anzahl versendeter Bits		

Das gesammelte Ergebnis dient nun als Gesprächsgrundlage über einen Austausch über die Güte eines Codes. Die Schülerinnen und Schüler sollen erläutern, warum es zu dieser unterschiedlichen Zahl von Fehlern kommt.

Nachdem die Schülerinnen und Schüler zu dem Schluss gekommen sind, dass Code Beta wohl der „bessere“ Code sein muss, kann nun die daran folgende Aufgabe des Arbeitsblattes betrachtet werden, so dass die Schülerinnen und Schüler auch die Nachteile des Codes Beta erkennen sollen.

Die Entscheidung, wann eine hohe Fehlerkorrektur von großer Bedeutung ist und wann man auch auf Fehlerkorrektur verzichten kann, hängt immer von der Situation ab. Ein paar exemplarische Situationen sind in der letzten Aufgabe des Arbeitsblattes aufgeführt, zu denen die Schülerinnen und Schüler ihre Einschätzung abgeben können.

6.3 Die Hamming-Distanz

Als Ergänzung kann die Hamming-Distanz eingeführt werden. Wie bereits erwähnt, ist diese nicht Bestandteil des Bildungsplans. Ein Tutorial zur Bestimmung der Hamming-Distanz (auch Hamming-Abstand genannt) findet man beispielsweise unter dem Link *https://www.youtube.com/watch?v=61kcq2RM8bo* (letzter Aufruf: 30.04.2018). Codes mit einer höheren Hamming-Distanz sind robuster gegenüber Fehlern.