

**Entwicklung der webbasierten
Kryptografie-Anwendung *KRYPTOKNACK* für die
Sekundarstufe I für den Mathematikunterricht
und die Digitale Grundbildung**

BACHELORARBEIT

in der Studienrichtung
Mathematik und Informatik

zur Erlangung des akademischen Grades
Bachelor of Education

eingereicht an der

**Fakultät für Mathematik, Informatik und Physik
der Universität Innsbruck**

von

Benedikt Maximilian Dornauer

Innsbruck,

Betreuer: Karin Schnass und Tobias Hell

Abstract

Was ist **KRYPTOKNACK**? KryptoKnack ist eine webbasierte Kryptografie-Anwendung, welche im Rahmen des vorliegenden Bachelorprojekts für die Sekundarstufe I entwickelt wurde.

Die Anwendung kann in der *Digitalen Grundbildung* zum Thema Datenschutz eingesetzt werden. Schüler*innen lernen mit Hilfe dieser Applikation, wie Verschlüsselungsverfahren genutzt werden, um Privatsphäre zu erzielen. Dabei erfahren sie, dass Geheimtexte geknackt werden können und es keine Garantie für absolute Sicherheit gibt.

Die Applikation kann auch fächerübergreifend eingesetzt werden, um Theorien aus der Mathematik zu behandeln. Die Themen funktionale Abhängigkeit, Zählen und relative Häufigkeit sowie Computational Thinking werden praxisbezogen behandelt.

Neben den didaktischen Einsatzmöglichkeiten im Unterricht, beschreibt die Arbeit das Konzept des Softwaresystems im Detail. Insbesondere umfasst dies einen stufenbasierten Softwaretest beruhend auf dem Konzept „A new Approach to Software Verification“ von David J.Panzel.

Inhaltsverzeichnis

1 Einleitung	5
2 Kryptografie und <i>KRYPTOKNACK</i> in der Sekundarstufe I	7
2.1 Datensicherheit durch Verschlüsselung (<i>KRYPTOKNACK</i> : Verschlüsselung)	7
2.1.1 Monoalphabetische Substitution	9
2.1.2 Atbasch-Verschlüsselung	10
2.1.3 Caesar-Chiffre	12
2.2 Potentielle Gefahren in Netzwerken (<i>KRYPTOKNACK</i> : CodeKnacker)	13
2.2.1 Gefahr durch Verzicht auf Verschlüsselung	13
2.2.2 Relative Sicherheit durch kryptografische Verfahren	14
2.2.3 Häufigkeitsanalyse um Texte zu „knacken“	15
3 Didaktische Möglichkeiten von <i>KRYPTOKNACK</i> im Unterricht	18
3.1 Digitale Grundbildung	18
3.2 Mathematische Schwerpunkte	20
3.2.1 Funktionale Abhängigkeiten	20
3.2.2 Zählen und Häufigkeit	21
3.2.3 Algorithmisches Denken (Computational Thinking)	22
4 Konzeptentwurf und fachliche Umsetzung der Anwendung	24
4.1 Systemüberblick und Use-Cases	24
4.2 Ausgewählte Technologien	27
4.3 Konzeptentwurf	29
4.3.1 GUI Component	30
4.3.2 Cryptography Component	31
4.3.3 Utility Component	31
4.4 Beschreibung des Konvertierungsprozesses beim Verschlüsseln	32
5 Funktionstests	35
5.1 Integrations- und Systemtest	36
5.2 Akzeptanztest	37
5.2.1 Untersuchungsdesign	37

5.2.2	Durchführung der Untersuchung	38
5.2.3	Auswertung der Ergebnisse	38
6	Reflexion und Nachwort	41
7	Danksagung	43
8	Anhang	51
8.1	Arbeitsmaterial	51
8.1.1	Einleitung und Erklärung	52
8.1.2	Atbasch Verfahren	53
8.1.3	Caesar Verfahren	54
8.1.4	Substitutions-Verfahren	55
8.1.5	Themenschwerpunkt: Stabilisierungscharakter bei der Häufigkeitsanalyse	56
8.1.6	Themenschwerpunkt: Sicherer Datenaustausch	57
8.1.7	Themenschwerpunkt: Sicherheit ist relativ	58
8.1.8	Lösungen	59
8.2	Funktionstest	60
8.2.1	Testergebnisse des Integrations- und Systemtests (01.01.2020)	60
8.2.2	Akzeptanztest	63

1 Einleitung

Täglich verwenden Jugendliche technische Hilfsmittel. Sie senden Nachrichten, laden Bilder auf *Instagram* hoch oder überweisen Geld auf ihr Jugendkonto. Während der Nutzung vertrauen sie darauf, dass ihre Privatsphäre gewahrt und ihre Daten vor unautorisierten Zugriffen geschützt sind. Da die Nutzungsdauer von mobilen Endgeräten in den letzten Jahren zugenommen hat, erscheint es sinnvoll, Jugendliche im Bereich Datensicherheit zu sensibilisieren [30, S. 46].

Die Vermittlung von solchen Kompetenzen sind seit 2018/2019 in der *Digitalen Grundbildung* in den österreichischen Lehrplänen der Neuen Mittelschule, sowie der allgemeinbildenden höheren Schule für die Sekundarstufe I verankert. Die *Digitale Grundbildung* fordert beim Thema *Sicherheit* ein, dass Schüler*innen verstehen, wie durch Kryptografie - Wissenschaft des verborgenen Schreibens - Datensicherheit erreicht werden kann. Außerdem soll vermittelt werden, dass Risiken und Bedrohungen in digitalen Umgebungen bestehen.

Basierend auf dieser Entwicklung wurde die Kryptografie-Anwendung *KRYPTOKNACK* (siehe Logo) für die Sekundarstufe I entwickelt. Die Verwendung dieser Applikation im Unterricht ermöglicht die geforderten digitalen Kompetenzen zu behandeln.

Da die Wissenschaft der Kryptografie eng mit der Mathematik verknüpft ist, liegt es auch nahe, die Anwendung im Mathematikunterricht zu verwenden. Themenschwerpunkte, wie *Funktionale Abhängigkeit*, *Zählen und Häufigkeit*, als auch *Algorithmisches Denken* können gezielt durch die Anwendung erarbeitet werden.



Abbildung 1: Logo

Basierend darauf, ergibt sich folgende Forschungsfrage für das Projekt:

Welchen Mehrwert hat der Einsatz der Kryptografie-Anwendung
KRYPTOKNACK in der Sekundarstufe I in der Digitalen Grundbildung
und im Mathematikunterricht?

Im ersten Kapitel dieser Arbeit wird aufgezeigt, wie durch Verschlüsselungsverfahren Datensicherheit erzielt wird und welche potentiellen Gefahren dabei einhergehen. Diese Themen werden in Bezug zu **KRYPTOKNACK** gebracht.

Anschließend wird dargestellt, welche didaktischen Möglichkeiten sich durch den Einsatz von **KRYPTOKNACK** ergeben. Dazu werden aktuelle Lehrpläne der Sekundarstufe I der Neuen Mittelschulen, als auch der allgemeinbildenden höheren Schule analysiert.

Nachdem erklärt wurde, wie **KRYPTOKNACK** im Unterricht eingesetzt werden kann, wird im Anschluss auf die technische Umsetzung, sowie auf das Testkonzept näher eingegangen.

Zum Schluss reflektiere ich meine Arbeit, zeige auf, welche Schwierigkeiten sich für mich ergaben und welche Erkenntnisse und Kompetenzen ich mir in diesem Projekt aneignete.

2 Kryptografie und **KRYPTOKNACK** in der Sekundarstufe I

Die *Kryptografie* - Wissenschaft des verborgenen Schreibens - versucht mit unterschiedlichen Verfahren Informationen zu verschlüsseln [22, S.12]. Durch mathematische Denkweisen, sowie mathematische Erkenntnisse können (sichere) Verfahren zum Verschlüsseln entwickelt werden. Diese Verfahren werden vielfältig in Netzwerken eingesetzt und tragen wesentlich zur Datensicherheit bei. [34, S.9 f.] Dass dabei Risiken bestehen, zeigen vielfältige Beispiele aus der Vergangenheit. Die Kryptoanalyse versucht verschlüsselte Texte zu „knacken“, um damit mögliche Gefahren aufzuzeigen.

2.1 Datensicherheit durch Verschlüsselung (**KRYPTOKNACK**: Verschlüsselung)

Häufig werden Datenfragmente - z.B. Bilder, Nachrichten - über ein öffentliches Netzwerk übermittelt. Dabei besteht das Kommunikationsschema oft aus zwei Partnern, die kommunizieren:

Sender Dieser ist Aussender eines Datenfragments über eine Netzwerk z.B. eine Nachricht auf *Signal*¹ schreiben.

¹Signal ist ein freier Messenger-Service der Nachrichten Ende-zu-Ende verschlüsselt.

Empfänger Der Empfänger wartet auf Nachrichten des Senders z.B. *Signal* wartet auf eingehende Nachrichten.

Auf dem Weg vom Sender zum Empfänger wird über ein Übertragungsmedium ein Datenfragment übermittelt, wie in Abbildung 2 dargestellt. Damit während des Datenaustausches ungewolltes Mitlesen verhindert wird, werden Verschlüsselungsverfahren angewandt. Der Klartext (meist ein Bit-Stream) wird vor dem Senden, nach einem bestimmten Verfahren, chiffriert und damit unlesbar gemacht, d.h. der Klartext ist nun Geheimtext. Die verschlüsselte Nachricht wird über das Übertragungsmedium an den Empfänger weitergeleitet. Dieser wartet auf eingehende Datenfragmente und entschlüsselt die Nachricht, um diese wieder für sich lesbar zu machen. [19, S.11 f.]

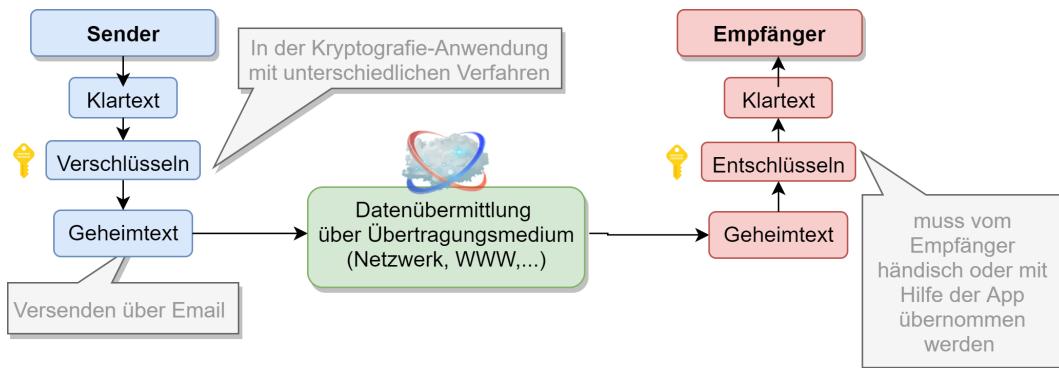


Abbildung 2: Übermitteln und Verschlüsseln eines Datenfragments

Der sichere Kommunikationsweg zwischen Sender und Empfänger kann durch die Kryptografie-Anwendung „simuliert“ werden. Dies erfolgt in drei Schritten:

1. Verschlüsseln Im ersten Schritt wählen die Schüler*innen ein Verschlüsselungsverfahren aus. Mit Hilfe dessen können sie ihre zu verschlüsselnde Nachricht chiffrieren. Dadurch wird die Nachricht unlesbar. Der*Die Anwender*in kennt das Verfahren und den benötigten Schlüssel, um die Nachricht wieder in lesbaren Text umwandeln zu können.

2. Senden Anschließend können die Sender ihre verschlüsselte Nachricht über die Anwendung per Email an einen Empfänger (Ziel der Nachricht) versenden. Die „verschlüsselte“ Datenübermittlung findet statt.

3. Empfangen Der Empfänger (Mail-Inhaber*in) erhält per Mail die verschlüsselte Nachricht - der Inhalt ist nicht lesbar. Wenn der Empfänger das Verfahren und den Schlüssel kennt, kann die verschlüsselte Nachricht entschlüsselt werden.

Beim Verschlüsseln kann zwischen einzelnen Verschlüsselungsverfahren gewählt werden, auf die nun näher eingegangen wird. Dabei charakterisiert die Schlüsselfunktionen c das jeweilige Verfahren. Die Entschlüsselungsfunktion wird mit c^{-1} bezeichnet. Die Buchstaben werden hierzu durch Zahlen ersetzt:

- A entspricht 0
- B entspricht 1
- ...
- Z entspricht 25

2.1.1 Monoalphabetische Substitution

Die monoalphabetische Substitution fasst eine Menge von unterschiedlichen Verschlüsselungsverfahren in einem (Ober-)begriff zusammen. Zum Beispiel handelt es sich beim Atbasch-Verfahren oder bei der Caesar-Verschiebung um Spezialfälle der freien Buchstabensubstitution. [36, S.24 ff.]

Bei der monoalphabetischen Substitution können die Buchstaben des Klartextalphabets nach einem zufälligen Prinzip neu angeordnet werden. Dabei ist darauf zu achten, dass jedes Zeichen des Alphabets **durch genau ein** anderes Zeichen ersetzt wird. [6, S.66]

Die Schlüssel-Funktion, welche den Zusammenhang erläutert, kann als Permutation

$$c : \{0, 1, \dots, 25\} \rightarrow \{0, 1, \dots, 25\}$$

der Zahlen $\{0, 1, \dots, 25\}$ aufgefasst werden und stellt eine bijektive Abbildung dar. Beispielsweise wird $0 \mapsto 5$, d.h. 'A' wird auf 'F' abgebildet. Die Permutation wird häufig in tabellarischer Form dargestellt:

Klartextalphabet	A	B	C	D	E	...	U	V	W	X	Y
Geheimtextalphabet	F	Y	L	R	W	...	Z	J	K	B	N

Tabelle 2: Beispiel eines Klartext- und Geheimtextalphabets einer freien Buchstabensubstitution

Wenden wir eine freie Buchstabensubstitution auf das lateinische Alphabet an, ergeben sich $26!$ mögliche Schlüssel, d.h. $403\,291\,461\,126\,605\,635\,584\,000\,000$ Möglichkeiten der Neuordnung. Das Verfahren erweckt den Eindruck - aufgrund der zahlreichen Möglichkeiten - sicher zu sein, doch mit der Häufigkeitsanalyse lassen sich solche Geheimtexte einfach entschlüsseln (vgl. 2.2.3 *Häufigkeitsanalyse um Texte zu „knacken“*, S. 15).

Hinweis

Übungsbeispiele mit **KRYPTOKNACK** zur Monoalphabetischen Substitution sind im Anhang unter *8.1.4 Substitutions-Verfahren*, S. 55 zu finden.

2.1.2 Atbasch-Verschlüsselung

Im Alten Testament fanden sich Textstellen wieder, welche für den Laien unverständlich waren. Zum Beispiel wurde das Wort Babel durch *Sheshach* ersetzt. Diese nicht-verständlichen Worte sollten der Bibel eine geheimnisvolle Aura verleihen. Bis in das Mittelalter wusste man nicht, was die verschlüsselten Botschaften wirklich bedeuteten. Erst den Mönchen gelang es die Verschlüsselung zu „knacken“. Es handelte sich hierbei um das Atbasch-Verfahren, eine Verschlüsselung aus Palästina 600 v.Chr.. [36, S.42 f.]

2.1 Datensicherheit durch Verschlüsselung (*KRYPTOKNACK*: Verschlüsselung)

Der Name Atbasch hat seinen Ursprung im hebräischen Schriftsystem und dieser gibt einen Hinweis auf die Vorgangsweise des Verfahrens. Das Wort Atbasch (A-T-B-SCH = Alef-Twa-Bet-Schin) stellt einzelne Buchstaben des hebräischen Alphabets dar. *Alef* ist der erste und *Twa* der letzte Buchstabe des Alphabets. Das Paar ist der Hinweis, die beiden Buchstaben zu vertauschen. Gleichermaßen wird der zweite Buchstabe *Bet* mit dem vorletzten Buchstaben *Schin* vertauscht. [36, S.43]

Übertragen wir die Idee aus dem 6. Jh. v.Chr. in das lateinische Alphabet, würde das bedeuten, dass A durch Z, B durch Y, C durch X usw. ersetzt wird. Beim Verschlüsseln eines einzelnen Buchstabens stellt man fest, um wie viele Stellen dieser vom Anfang des Alphabets entfernt ist. Anschließend ersetzt man den Buchstaben durch jenen Buchstaben, der genau gleich viele Stellen vom Ende des Alphabets entfernt ist. [6, S.101] Folgend wird das Klartext- und Geheimtextalphabet von Atbasch dargestellt.

Klartextalphabet	A	B	C	D	E	...	V	W	X	Y	Z
Geheimtextalphabet	Z	Y	X	W	V	...	E	D	C	B	A

Tabelle 4: Klartext- und Geheimtextalphabet beim Atbasch-Verfahren

Allgemein kann dieses Verfahren über folgende Funktion ausgedrückt werden [37, S.5]:

$$c : \{0, 1, \dots, 25\} \rightarrow \{0, 1, \dots, 25\}$$

$$k \mapsto 25 - k$$

Hinweis

Übungsbeispiele mit *KRYPTOKNACK* zum Atbasch-Verfahren sind im Anhang unter *8.1.2 Atbasch Verfahren*, S. 53 zu finden.

2.1.3 Caesar-Chiffre

Gaius Iulius Caesar (100 v.Chr. - 44 v.Chr.) - Kaiser des römischen Reiches - eroberte während seiner Lebzeiten viele Gebiete, um seine Macht auszubauen. Damit er während seiner Belagerungszüge nicht belauscht wurde, griff er auf verschiedene Verschlüsselungsverfahren zurück, um seine Botschaften zu verschleiern. Der Schriftsteller *Gaius Suetonius Tranquillus* beschrieb in seinen Büchern unterschiedliche Verschlüsselungen, die Caesar entwickelt hatte. Im Werk *Divus Iulius* berichtet der Autor vom wohl bekanntesten monoalphabetischen Substitutionsverfahren von Gaius Iulius Caesar. Es ist unter dem Namen Caesar-Chiffre oder kurz Caesar bekannt. [6, S.108 f.]

„Will jemand sie [die geheime Botschaft, Anm. d. Verf.] entziffern und hintereinander lesen, so muss er immer den vierten Buchstaben des Alphabets, also D für A und sofort an die Stelle des wirklich Geschriebenen setzen.“ [40, S.75]

Gaius Suetonius Tranquillus beschreibt hier eine zeichenweise Verschiebung um drei Buchstaben im Alphabet. Es ergibt sich folgende tabellarische Darstellung:

Klartextalphabet	A	B	C	D	E	...	V	W	X	Y
Geheimtextalphabet	D	E	F	G	H	...	D	C	B	A

Tabelle 6: Klartext- und Geheimtextalphabet bei der Caesar-Chiffre

Die ursprüngliche Caesar-Chiffre kann durch ein allgemeines Verfahren ersetzt werden. Dazu wird ein Verschiebe-Schlüssel $k \in \mathbb{N}$ gewählt, welche die Anzahl der Verschiebestellen beschreibt. [37, S.5]

$$c : \{0, 1, \dots, 25\} \times \mathbb{N} \rightarrow \{0, 1, \dots, 25\}$$

$$(l, k) \mapsto (l + k) \bmod 26$$

Hinweis

Übungsbeispiele mit KrytoKnack zum Caesar-Verfahren sind im Anhang unter *8.1.3 Caesar Verfahren, S. 54* zu finden.

2.2 Potentielle Gefahren in Netzwerken (**KRYPTOKNACK** : CodeKnacker)

Über das WorldWideWeb (WWW) übermitteln wir Datenfragmente über transparente und nicht-transparente Routen. Dabei passieren unsere Daten zahlreiche Teilnetzwerke, häufig von unterschiedlichen Besitzer*innen, um das Zielnetzwerk zu erreichen. [29, S.488 f.] Auf unserem Weg vom Anfangs- zum Zielnetzwerk besteht die Möglichkeit im Netz, dass unsere Daten abgefangen werden und von nicht autorisierten Nutzern (Geheimdienste, Hacker, etc.) missbraucht werden. Private Daten wie Passwörter, Bilder oder Nachrichten sind in Gefahr und somit unsere Privatsphäre.

2.2.1 Gefahr durch Verzicht auf Verschlüsselung

Wenn Daten auf dem Weg vom Sender zum Empfänger im Netzwerk nicht verschlüsselt sind, können diese direkt mitgelesen werden. Ein bekanntes Beispiel, wenn Verschlüsselung vernachlässigt wird, stammte aus dem Jahr 2014. Damals ereignete sich ein großer Datenskandal bei der Firma *Sony Pictures Entertainment*. Hackern gelang es in das firmeninterne Netzwerk einzudringen und auf dessen Strukturen zuzugreifen. Dabei zeigte sich ein schwerwiegender Fehler im System des Unternehmens. Daten lagen ohne jegliche Verschlüsselung auf den Servern, wodurch es Hackern möglich war ohne jegliche Entschlüsselung auf Filme ungesichert zuzugreifen, Gehaltslisten von Angestellten einzusehen oder sogar deren medizinische Befunde abzurufen. [4]

Dieses Beispiel zeigt, wie essentiell der Einsatz von Schutzmechanismen durch Verschlüsselung ist. Durch die Entwicklung von Verschlüsselungsverfahren gelingt es unsere Daten- und Privatsphäre zu schützen, um Missbrauch wie beim oben genannten Unternehmen zu verhindern. [13, S.88]

2.2.2 Relative Sicherheit durch kryptografische Verfahren

Wird nun Verschlüsselung in einem System eingesetzt, bedeutet dies noch immer nicht, dass es absolut sicher ist. Sicherheit ist ein relativer Begriff, denn es besteht immer die Gefahr, dass Verschlüsselungen geknackt werden können. [32, S.64]

Eine der bekanntesten Beispiele aus der Zeitgeschichte ist die Chiffriermaschine *Enigma* (siehe Abbildung 3). Sie galt während der zweiten Hälfte des 20. Jahrhunderts als unbezwingbare Maschine zum Verschlüsseln und Entschlüsseln von Texten ([5]). Die USA hielt es damals für sehr unwahrscheinlich, dass ohne Wissen über den Schlüssel Geheimtexte geknackt werden können [35]. Trotz der scheinbaren „Unbesiegbarkeit“ gelang es dem britischen Geheimdienst, u.a. durch Alan Turing *Enigma* zu entschlüsseln. Es wird sogar vermutet, dass dadurch der Zweite Weltkrieg verkürzt werden konnte [6, S.153].



Abbildung 3: Exemplar einer Enigma [2]

Auch heute besteht die Gefahr, dass unsere bisherigen Verschlüsselungen geknackt werden. Deshalb beschäftigt sich die Kryptoanalyse, Teilgebiet der Kryptologie mit den einzelnen Verfahren. Das Fachgebiet untersucht inwieweit die Verfahren resistent gegenüber Angriffen („knacken“) sind, um mögliche Schwachstellen zu erheben. [37, S.1] Aktuell stehen die Forscher*innen in diesem Gebiet vor einem Durchbruch in der Kryptoanalyse [16, S.408]. Die Benutzung von Quantencomputer könnte ermöglichen, dass Verschlüsselungen aus diesem Jahrhundert in überschaubarer Zeit geknackt werden. Bekannte, verbreitete Verfahren wären gefährdet und müssten deshalb weiterentwickelt oder durch neue Verfahren ersetzt werden [34, S.325 f.].

2.2.3 Häufigkeitsanalyse um Texte zu „knacken“

Die Häufigkeitsanalyse ist ein wichtiger Bestandteil der Kryptoanalyse, um Chiffriertexte zu „knacken“. Bereits im 9. Jahrhundert beschäftigten sich viele arabische Gelehrte mit der Entzifferung kryptografischer Botschaften. Das erste uns bekannte Werk, das sich mit der Entschlüsselung geheimer Nachrichten beschäftigt, wurde

von Abū Ya‘qūb ibn Ishāq al-Kindī verfasst. [6, S.110] Der Gelehrte beschreibt auf der ersten Seite seines Werkes *Abhandlung über die Entzifferung kryptografischer Botschaften* die Häufigkeitsanalyse wie folgt: „eine Möglichkeit, eine verschlüsselte Botschaft zu entziffern, vorausgesetzt wir kennen ihre Sprache.“ [36, S.35]

Die Kernidee des al-Kindī Verfahrens beruht auf der Annahme, dass der am häufigsten vorkommende Buchstabe eines Alphabets auch jener ist, welcher am häufigsten in der verschlüsselten Nachricht vorkommt [6, S.110]. Im deutschen Alphabet zählen zu den "Top drei": E (17.49%), N (9.78%) und R (7.00%). Nun können Kryptoanalytiker versuchen, anhand von Wahrscheinlichkeiten verschlüsselte Buchstaben richtig zu deuten und so schrittweise verschlüsselte Botschaften zu entziffern. Dieses Verfahren eignet sich insbesondere dann, wenn die Anzahl der Buchstaben in der Geheimbotschaft steigt, da die Trefferquote zunimmt. Simon Singh nennt hier einen Richtwert von 100 Buchstaben, ab wann Entzifferung gelingt. [36, S.36 f.]

Eine weitere Möglichkeit zur Entschlüsselung von Texten beruht auf der Analyse der Häufigkeiten anhand Bigrammen. Bigramme - Buchstabenfolge von 2 aufeinanderfolgenden Buchstaben - helfen beim Überprüfen von Vermutungen und geben Hinweise auf die Verschlüsselung.

Bemerkung

Für eine detaillierte Beschreibung der Häufigkeitsanalyse möchte ich auf *Singh, Simon. 2016. Geheime Botschaften: Die Kunst der Verschlüsselung von der Antike bis in die Zeiten des Internet. Kapitel: Die arabischen Kryptoanalytiker (S.30-42)* verweisen.

Damit Schüler*innen sich in die Rolle eines Kryptoanalytikers versetzen können, verfügt **KRYPTOKNACK** den *CodeKnacker*. Der *Codeknacker* ist ein grafisches Tool, das Schüler*innen beim Knacken von monoalphabetisch-verschlüsselten Geheimtexten unterstützt.

Es verfügt über:

- Repräsentation von Häufigkeiten in Diagrammen.
- Wahl unterschiedlicher Verfahren und Schwierigkeitsgrade.
- (Automatisierte) Ersetzung von Buchstaben im Geheimtext.

Mit Hilfe der Häufigkeitsanalyse und durch Verwendung des Codeknackers können Schüler*innen verschlüsselte Texte untersuchen.

3 Didaktische Möglichkeiten von *KRYPTOKNACK* im Unterricht

Im Schuljahr 2018/2019 wurde zum ersten Mal das Schulfach *Digitale Grundbildung* in der Sekundarstufe I flächendeckend in Österreich umgesetzt [9]. In Auseinandersetzung mit dieser neuen Entwicklung wurde die Anwendung *KRYPTOKNACK* konzipiert. Die Anwendung bietet die Möglichkeit Inhalte aus der neu verordneten *Digitalen Grundbildung* zu behandeln. Des Weiteren können mit Hilfe der Applikation die mathematischen Themen *Funktionale Abhängigkeit, Zählen und Häufigkeit*, als auch *Algorithmisches Denken* praxisnah erläutert werden.

Hinweis

Passende Arbeitsmaterialien zu den einzelnen Themen, welche in diesem Kapitel angesprochen werden, sind unter *8.1 Arbeitsmaterial, S. 51* bereitgestellt.

3.1 Digitale Grundbildung

Die neu eingeführte *Digitale Grundbildung* sieht im Lehr-/Lernauftrag vor, dass Schüler*innen Kompetenzen im sicheren Umgang mit den neuen Medien vermittelt bekommen. Die *Änderung der Verordnung über die Lehrpläne der Neuen Mittelschulen* sowie der *Verordnung über die Lehrpläne der allgemeinbildenden höheren Schulen* umfassen:

1. „Im Mittelpunkt steht dabei die **reflektierte Verwendung von Medien und Technik.**“ [10, S.2, Hervorhebung Autor]
2. „Schülerinnen und Schüler verwenden **Software zur Verschlüsselung von Daten.**“ [10, S.10, Hervorhebung Autor]
3. „Freie digitale Informations- und Kommunikationsnetze bieten dazu weitreichende kommunikative, soziale und kreative Möglichkeiten, bergen aber auch **Risiken und Gefahren für den Einzelnen.**“ [10, S.3, Hervorhebung Autor]
4. „Schülerinnen und Schüler sind sich **Risiken und Bedrohungen in digitalen Umgebungen bewusst.**“ [10, S.3, Hervorhebung Autor]

Die Kryptologie - als Teilgebiet der Computersicherheit - bietet das Potential die aufgelisteten Punkte im Rahmen der *Digitalen Grundbildung* zu behandeln. *KRYPTOKNACK* als digitales Unterstützungstool ist dabei von Nutzen.

Um Schüler*innen aufzeigen zu können, wie Sicherheit im Netz angestrebt wird, werden einfache kryptografische Verfahren besprochen. *KRYPTOKNACK* unterstützt dabei monoalphabetische Verfahren, da sich diese laut Thomas Borys besonders gut eignen, um Verschlüsselungen einfach zu erklären. Beim Arbeiten mit diesen Verfahren soll eine Vorstellung geprägt werden, wie moderne Systeme funktionieren, um Datensicherheit zu garantieren und das Recht der informationellen Selbstbestimmung zu wahren. [6, S.221]

Die potentiellen Gefahren $\hat{=}$ Risiken, die im Netz bestehen, können in der Kryptologie in Bezug zur Kryptoanalyse gebracht werden. Ein Bewusstsein für diese Sicherheitsproblematik soll bei Schüler*innen geschaffen werden, als präventive Maßnahme gegenüber potentiellen Gefahren. [6, S.41] Eine Maßnahme im Unterricht wäre die Verwendung des *CodeKnackers*. Durch Häufigkeitsanalyse „knacken“ die Anwender*innen Geheimtexte und erfahren, dass „geheim“ relativ ist. Die Fehlvorstellung von absoluter Sicherheit kann dabei behandelt und auf die Gefahren hingewiesen werden.

3.2 Mathematische Schwerpunkte

Die Digitale Grundbildung kann nicht nur als eigenständiges Unterrichtsfach digitale Kompetenzen vermitteln, sondern auch in anderen Fächern umgesetzt werden [9]. Besonders die Mathematik sei ein wichtiges Hilfsmittel der Kryptografie, wie Wissenschaftler Klaus Schmeh festhält. Durch mathematische Denkweisen und durch Hilfe von mathematischen Erkenntnissen wäre es erst möglich Verschlüsselungen zu entwickeln. [34, S.9] Deshalb liegt es nahe mit Hilfe von *KRYPTOKNACK* mathematische Schwerpunkte aus der Sekundarstufe I praxisbezogen im Unterricht zu behandeln. Mögliche Themenschwerpunkte werden im Nachfolgenden aufgezeigt.

3.2.1 Funktionale Abhängigkeiten

Damit Verschlüsselungen charakterisiert und beschrieben werden können, spielen Funktionen eine wichtige Rolle in der Kryptografie. Besonders injektive Funktionen ermöglichen eine Beschreibung von Chiffrierungen [19, S.19]. Die zentrale Rolle der Funktionen legt nahe, dass, wenn im Unterricht mit Verschlüsselungen gearbeitet wird, die fundamentale Idee des funktionalen Zusammenhangs geprägt wird [6, S.186]. Der österreichische Lehrplan der Mathematik NMS, sieht vor, dass Schüler*innen „durch das Arbeiten mit funktionalen Abhängigkeiten einen intuitiven Funktionsbegriff erarbeiten“ [8, S.8]. Im Sinne des Spiralcurriculums kann in der Sekundarstufe I ein intuitives Funktionsverständnis - durch enaktive (Erkenntnisgewinn durch Handlungen) sowie ikonische Handlungen (Erkenntnisgewinn durch Bilder) - geprägt werden [17, S.2].

Aus diesem Grund verfolgt die Anwendung das Ziel eine präformale, anwendungsorientierte Form des intuitiven Funktionsverständnisses durch visuelles Anschauungsmaterial umzusetzen. Im Mittelpunkt der Anwendung steht das Erkunden von funktionalen Zusammenhängen, das Verständnis für unterschiedliche Repräsentationen und die „Verwendung von Funktionen im informellen Sinn“. Deswegen wird

auf eine numerische Darstellung in **KRYPTOKNACK** verzichtet und folgende Formen der Darstellung von funktionalen Zusammenhängen gewählt:

- **grafische Repräsentation:** Tabellen und Bilder
- **textuell:** Beschreibung von Verschlüsselung in Form von verständlichen Texten

Diese nicht numerischen Darstellungsformen seien laut Thomas Borys außerordentlich überzeugende Beispiele für die Erläuterung funktionaler Zusammenhänge [6, S.182].

	<p>Beim Verschlüsseln eines einzelnen Buchstabens stellt man fest, um wie viele Stellen dieser vom Anfang des Alphabets entfernt ist. Anschließend ersetzt man den Buchstaben durch jenen Buchstaben, der genau gleich viele Stellen vom Ende des Alphabets entfernt ist.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Klartextalphabet</th><th>A</th><th>B</th><th>C</th><th>D</th><th>E</th><th>F</th><th>...</th><th>X</th><th>Y</th><th>Z</th></tr> </thead> <tbody> <tr> <th>Schlüsseltextralphabet</th><td>Z</td><td>Y</td><td>X</td><td>W</td><td>V</td><td>U</td><td>...</td><td>C</td><td>B</td><td>A</td></tr> </tbody> </table>	Klartextalphabet	A	B	C	D	E	F	...	X	Y	Z	Schlüsseltextralphabet	Z	Y	X	W	V	U	...	C	B	A
Klartextalphabet	A	B	C	D	E	F	...	X	Y	Z													
Schlüsseltextralphabet	Z	Y	X	W	V	U	...	C	B	A													

Abbildung 4: Beispiele zu den Darstellungsformen von Funktionen in **KRYPTOKNACK**: Tabellen, Texte und Bilder

3.2.2 Zählen und Häufigkeit

Um das Jahr 815 n.Chr. stellten Religionsgelehrte der theologischen Schulen fest, dass bestimmte Buchstaben in Texten besonders häufig vorkommen. Diese Entdeckung war der Ursprung der Kryptoanalyse, welche auf der Idee des **Zählens** oder der **relativen Häufigkeit** beruht. [36, S.31 ff.] Die Anwendung umfasst das Unterstützungstool *CodeKnacker*. Dieses Werkzeug unterstützt Schüler*innen bei der statistischen Analyse von Geheimtexten, um diese zu entschlüsseln und bietet die Möglichkeit eines anwendungsorientierten Einstiegs in die Statistik. Orientiert am österreichischen Lehrplan der NMS und AHS Sekundarstufe I können folgende Themen mit Hilfe der Anwendung abgedeckt werden:

Darstellung von Daten

Der *CodeKnacker* spielt beim Darstellen von Daten die Hauptrolle. Er übernimmt die Aufgabe des Zählens und repräsentiert die Daten in Form von Säulendiagrammen. Schüler*innen arbeiten mit statistischen Darstellungsformen, können die Daten untersuchen und darüber hinaus interpretieren.

Relative Häufigkeit und Stabilisierungscharakter

Damit Texte entschlüsselt werden können müssen Häufigkeiten miteinander verglichen werden (vgl. 2.2.3 *Häufigkeitsanalyse um Texte zu „knacken“*, S. 15). Dadurch liegt es nahe, über die relative Häufigkeit beim Knacken von Codes im Unterricht zu sprechen und praxisbezogen zu behandeln.

Des Weiteren kann die Kryptoanalyse genutzt werden, um die Stabilisierungseigenschaft von relativen Häufigkeiten zu besprechen. Umso mehr Buchstaben ein sinnvoller Geheimtext besitzt, desto aussagekräftiger werden die relativen Häufigkeiten der einzelnen Buchstaben. Nun können die erhobenen Werte durch die Analyse des Geheimtextes mit den statistischen Erhebungen verglichen werden, um einen möglichen Schlüssel zu finden (siehe Tabelle 7).

Anzahl Buchstaben	5	30	1000	10 000
Absolute Häufigkeit von 'h'	1	3	166	1690
Relative Häufigkeit von 'h'	0.20	0.10	0.166	0.169

Tabelle 7: Ein Text wurde mittels Caesar verschlüsselt. Dabei wurde 'e' durch 'h' ersetzt. Anschließend wurde der Text analysiert. Dazu wurden 5, 30, 1000 und 10000 Zeichen zur Analyse herangezogen. Die erwartete Häufigkeit des Buchstabens 'e' liegt bei 17.4%. Mit zunehmender Anzahl an analysierten Buchstaben stabilisiert sich die Häufigkeit.

3.2.3 Algorithmisches Denken (Computational Thinking)

Damit Schüler*innen den richtigen Schlüssel beim „Knacken“ von Texten finden, benötigt es **algorithmisches Denken**. Dieser Begriff beschreibt die Fähigkeit

gegebene Probleme zu lösen und eine neue Art des Denkens in der modernen Welt zu etablieren. Durch eine Folge von Schritten, Vermutungen, Fehlschlägen oder Erkenntnissen erarbeiten die Schüler*innen eine Lösung. [12, S.3ff]. „Computational Thinking“ ist ein zentraler Begriff der „neuen“ *Digitalen Grundbildung*.

4 Konzeptentwurf und fachliche Umsetzung der Anwendung

Im Rahmen des Bachelorprojekts wurde eine responsive, webbasierte Kryptografieanwendung zur Unterstützung von Lehrer*innen und Schüler*innen für die Sekundarstufe I entwickelt. Dieses Kapitel beschreibt im Detail die Konzeption der Software, geht näher auf die eingesetzten Technologien ein und erläutert die verwendete Softwarearchitektur.

4.1 Systemüberblick und Use-Cases

KRYPTOKNACK ist eine über das Internet zugängliche Web-Anwendung. Wenn User*innen auf die Webseite zugreifen, gelangen sie auf die Startseite. Es wird eine Übersicht der unterschiedlichen Funktionalitäten gezeigt. Die Anwender*innen können die verschiedenen Verschlüsselungsverfahren wählen, genaue Informationen über die Verfahren erhalten, sowie zum Unterstützungstool *CodeKnacker* wechseln.

In Form eines Use-Case-Diagramms werden die diversen Ziele der Anwendung erfasst und aus Sicht der Akteure (Schüler*innen, Lehrer*innen, etc.) beschrieben. Auf diesem Diagramm (vgl. Abbildung 5) beruht die Modellierung der konzipierten Software. Außerdem werden die Use-Cases in der Testphase (siehe 5.2 *Akzeptanztest*, S. 37) herangezogen, um zu ermitteln, ob die beschriebenen Anwendungsfälle erfüllt sind [7, S.74 f.].

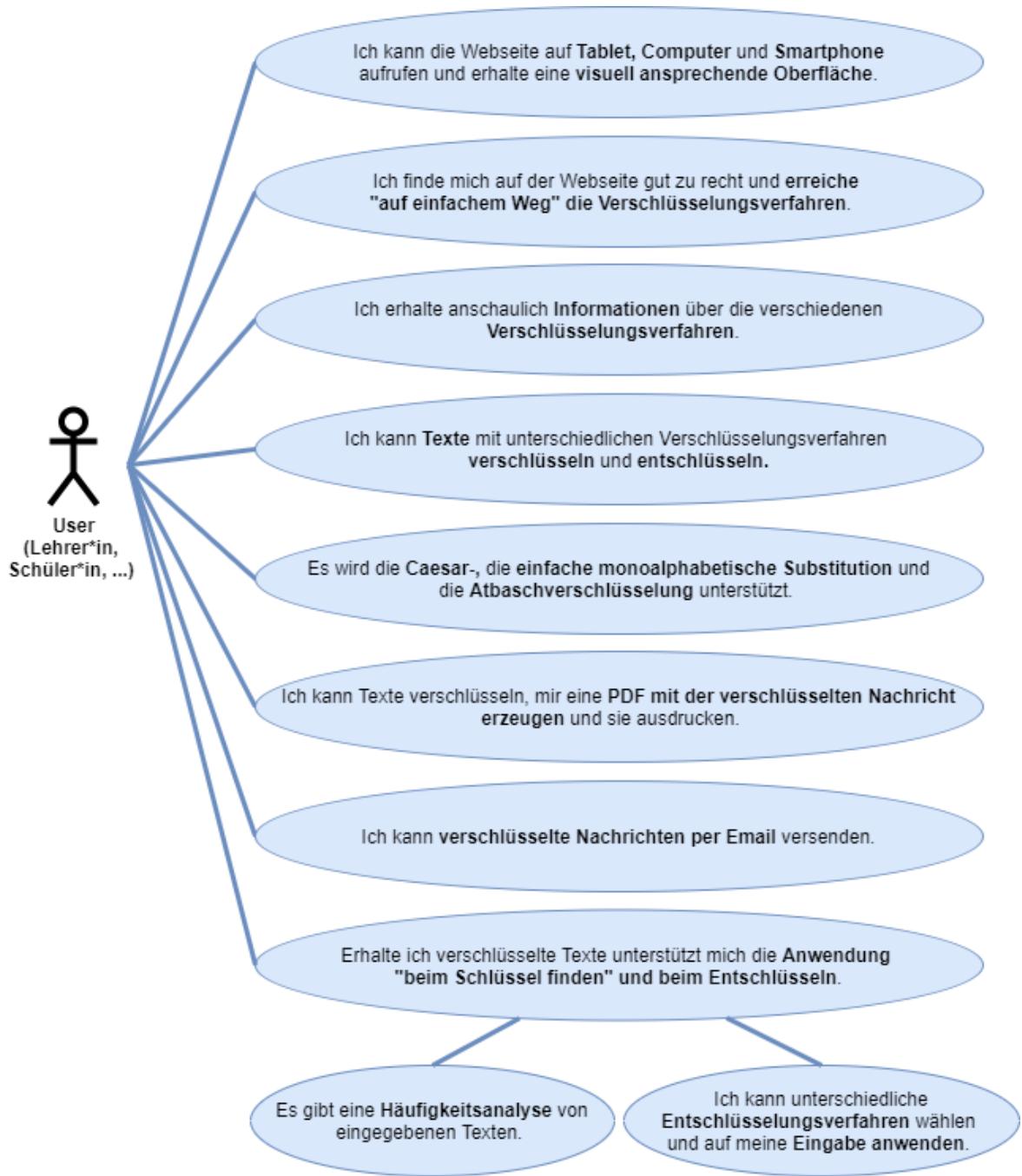


Abbildung 5: Use-Case-Diagramm

Basierend auf den beschriebenen Use-Cases wurden folgende Dienste konzipiert, die nun im Detail dargestellt werden.

Informationen über Verschlüsselungsverfahren: Durch das Lesen von Beschreibungen erfahren Nutzer*innen Informationen über das jeweilige Verschlüsselungsverfahren. Die Erklärung soll in ihrer Komplexität für 10-jährige bis 14-jährige Schüler*innen fassbar sein und ergänzend durch Beispiele erklärt werden.

Verschlüsseln/Entschlüsseln: Eine Kernkompetenz der webbasierten Anwendung liegt im Verschlüsseln und Entschlüsseln von Texten. Über Eingabefelder können User*innen beliebige Texte eingeben. Unterschiedliche Formatierungsmöglichkeiten ermöglichen eine Anpassung. Die Eingaben werden durch das gewählte Kryptografieverfahren und den festgelegten Schlüssel verschlüsselt oder können auch wieder entschlüsselt werden.

Email Service: Um eine Art „Messenger-Service“ bereitzustellen - damit z.B. Schüler*innen sich gegenseitig Nachrichten senden können - wird ein Email-Service integriert. Mittels Email-Adresse und einer persönlichen Nachricht können verschlüsselte Texte versendet werden.

PDF-Service: Die Anwendung bietet das Angebot verschlüsselte Nachrichten in Form eines anschaulichen PDF-Dokuments bereitzustellen. Die Entwicklung von individuellem Lehr-/Lernmaterial wird angestrebt: Mit den Unterlagen entschlüsseln die Schüler*innen Texte und entdecken spielerisch die Verschlüsselungstechniken.

CodeKnacker: Der CodeKnacker ist ein grafisches Unterstützungstool für Schüler*innen beim „Knacken“ von Geheimtexten mit Hilfe der Häufigkeitsanalyse. Die Anwendung hilft beim Zählen von Buchstaben, stellt Häufigkeiten in Form von Diagrammen dar und unterstützt so das Auffinden des richtigen Schlüssels.

4.2 Ausgewählte Technologien

Apache Maven *Apache Maven* ist ein Build-Management-Tool, um die Aufgaben des Testens, das Auflösen von Abhängigkeiten, das Laden von Paketen, kompilieren und endgültiges Ausführen zu vereinfachen. Automatisierte Abläufe erleichtern den Prozess der Softwareentwicklung [38, S.7f].

Java Die objektorientierte Programmiersprache *Java*, entwickelt von dem Unternehmen *Sun Microsystems Oracle*, unterstützt die Entwicklung von webbasierten Applikationen. Die plattformunabhängige Sprache ermöglicht die serverseitige Anwendung auf zahlreichen Endgeräten (Computer, Tablets oder Smartphones). Das Backend der Anwendungen wird mit Hilfe dieser Programmiersprache umgesetzt.

Spring Framework Um wesentliche Aspekte von Java zu vereinfachen, wird das Open-Source *Spring-Framework* herangezogen und bringt zahlreiche strukturelle Vorteile. Das häufig verwendete Framework bietet Möglichkeiten von vielfältigen Erweiterungen.[41, S.1ff] In diesem Projekt werden insbesondere *Spring Web Service* (Erstellung von Webanwendungen), sowie *Spring MVC* (Einhaltung des Model-View-Controller-Patterns) und *Spring Web Flow* (Abläufe auf der Webseite) herangezogen, um die Anwendung umzusetzen [24, S.5 ff.].

iText Um automatisiert PDFs zu erstellen, wird eine freie Version des Toolkits *iText* herangezogen. Diese ermöglicht die individuelle Generierung von PDF-Dokumenten unter Beachtung des globalen Standards *PDF 2.0*.[1]

Primeface Bei *Primefaces* handelt es sich um eine freie *JavaServer Face (JSF)* Komponenten-Bibliothek. Sie bietet viele Bausteine zur Gestaltung der Webseite an - wie z.B. Buttons, Eingabefelder oder Ajax-Listener - und erleichtert dadurch die Layout-Gestaltung.

Extensible Hypertext Markup Language Bei XHTML handelt es sich um eine auf XML *Extensible Markup Language* basierende Aufzeichnungssprache, welche wichtige Eigenschaften von HTML aufgreift [23, S.56 f.]. Die Sprache wird von (fast) allen Browsern unterstützt und eignet sich aufgrund der Flexibilität und Leistungsfähigkeit hervorragend zur Strukturierung von Seiten für das Frontend des Users. [11, S.7]

Cascading Style Sheets Wie der Name bereits verrät, unterstützt *Cascading Style Sheets* (CSS) die Designgestaltung von XHTML-Seiten und trennt Design von Inhalt [26, S.11 f]. Durch CSS soll im Projekt eine ansprechende Oberfläche für die Endnutzer*innen erreicht werden. Insbesondere wird durch CSS ein responsives Design ermöglicht.

Java-Script Um die Möglichkeiten von HTML und CSS zu erweitern, wird *Java-Script* ergänzend eingesetzt. Es ermöglicht dynamische Webseiten-Anpassungen sowie das Verändern von Webinhalten. Dies wird häufig für die clientseitige Anpassung der Weboberfläche verwendet. [18, S.1 ff.]

JUnit Um die Qualität der Software zu überprüfen, somit die Funktionalität zu testen, werden JUnit Tests herangezogen. Bei *JUnit* handelt es sich um ein Open-Source-Framework, das zahlreiche Testmethoden anbietet. Automatisierte Tests überprüfen, ob Funktionalitäten korrekt ablaufen oder fehlschlagen (z.B. werden Texte korrekt verschlüsselt und entschlüsselt). [3, S.10 ff.]

4.3 Konzeptentwurf

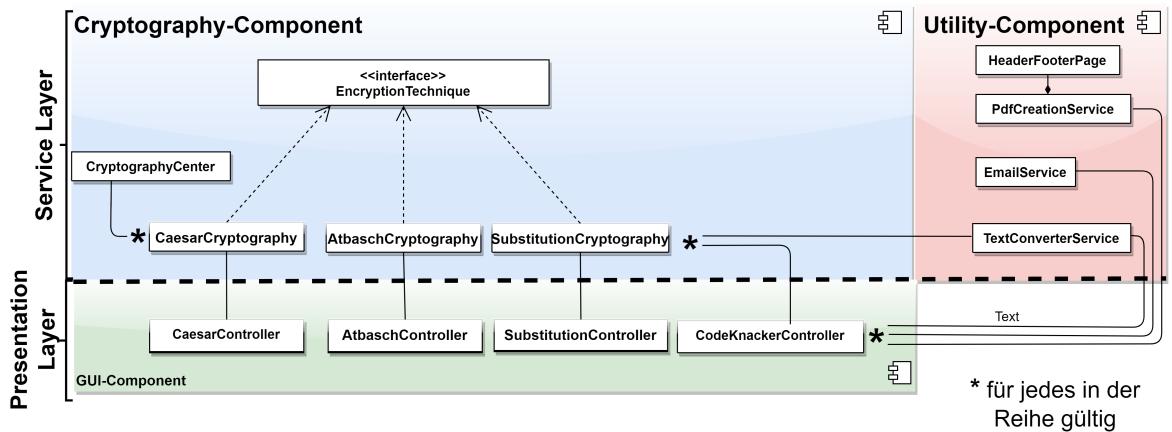


Abbildung 6: Klassendiagramm

Die Verwendung des *Spring MVC Frameworks* im Projekt strebt die Umsetzung des Model-View-Controller-Architektur (MVC) als Softwarearchitektur an [24, S.426]. Diese Drei-Schichten-Architektur (engl. Three Layered Architecture) wird im Rahmen des Projektes passend adaptiert. Sie ermöglicht eine lose Koppelung - wodurch eine Unabhängigkeit zwischen den Schichten erreicht wird. [14, S.17 ff.] Da die Speicherung persistenter Daten bei KryptoKnack nicht vorgesehen ist, wird auf die Persistenzschicht im MVC-Framework verzichtet. Aus diesem Grund besteht der Softwareentwurf nur aus der Präsentationsschicht (Presentation Layer) und Geschäftslogikschicht (Business Layer), wie in Abbildung 6 dargestellt wird.

Bemerkung

Die Erweiterung durch Speicherung persistenter Daten könnte einfach umgesetzt werden, ist jedoch in diesem Projekt nicht vorgesehen.

Präsentationsschicht Die Präsentationsschicht ist für die Darstellung der Weboberfläche verantwortlich. Unterschiedliche Benutzerschnittstellen (z.B. Buttons, Bilder oder Eingabefelder) werden durch Controller verwaltet. Im Projekt sind insbesondere Controller für die eingehenden Anfragen der jeweiligen Kryptografieverfahren verantwortlich. Diese Schicht umfasst die Komponente *GUI Component*.

Geschäftslogikschicht Um die Anfragen (z.B. Drücken eines Buttons) der einzelnen Controller zu verwalten, stehen unterschiedliche Services zur Verfügung. Diese bearbeiten die Anfragen nach bestimmten Verarbeitungsmechanismen. [28, S.260] Solch ein Fall könnte zum Beispiel das Verschlüsseln oder Entschlüsseln eines Textes sein. Insbesondere umfasst diese Schicht die *Cryptography Component*, sowie *Utility Component*.

Diese beschriebene Schichtarchitektur umfasst wie erwähnt drei Komponenten: *Cryptography Component*, *Utility Component* und *GUI Component*. Im Nachfolgenden werden die einzelnen Komponenten näher erläutert und auf deren Funktionalitäten eingegangen.

4.3.1 GUI Component

Die *GUI Component* im Projekt ist für die Realisierung des Benutzerfrontends verantwortlich. Eine mittels XHTML, Javascript und CSS aufgebaute Webseite ermöglicht die Bedienung (durch Buttons, Eingabefelder und Schieberegler) der einzelnen Funktionalitäten. [28, S.40] Über diese Benutzeroberfläche erhalten Nutzer*innen Zugriff auf die im Abschnitt Systemüberblick und Use-Cases beschriebenen Möglichkeiten (z.B. Verschlüsseln, Informationen aufrufen oder CodeKnacker bedienen). Aktionen, die von Nutzer*innen ausgelöst werden (z.B. durch Drücken eines Knopfes), werden durch MVC-Controller verwaltet: Anfragen werden entgegengenommen, dabei Geschäftslogikprozesse der dazugehörigen Services ausgelöst und die passende Repräsentation der GUI gewählt. [24, S.28 und S.426]

4.3.2 Cryptography Component

Die verschiedenen Verschlüsselungsvorgänge, die von der GUI-Komponente benötigt werden, sind in der *Cryptography Component* zusammengefasst. Sie bieten die benötigten Schnittstellen an. Um die Erweiterbarkeit der Software zu sichern, wurde zur konzeptionellen Umsetzung das Entwurfsmuster *Strategy Pattern* gewählt. Durch die Vorteile der Modifizierbarkeit und Wiederverwendbarkeit dieses Patterns kann sichergestellt werden, dass neue Verschlüsselungsverfahren einfach ergänzt werden können. [39, S.144 ff.]

Jedes Verschlüsselungsverfahren repräsentiert eine eigene Klasse, welche nach Implementierung des Interfaces „EncryptionTechnique“ Funktionen zum Ver- und Entschlüsseln einzelner Buchstaben bereitstellt. Die Klasse *CryptographyCenter* enthält eine Referenz auf das gewünschte Verschlüsselungsverfahren, um auf die Funktionen des jeweiligen Verfahrens zuzugreifen. Dann übernimmt *CryptographyCenter* die Aufgabe Eingaben in passende Formate zu konvertieren und den gesamten Verschlüsselungsprozess (siehe *4.4 Beschreibung des Konvertierungsprozesses beim Verschlüsseln*, S. 32) zu bewerkstelligen. Dabei greift die *Cryptography Component* über Schnittstellen auf die Services der *Utility Component* zu.

4.3.3 Utility Component

Die *Utility Component* ist Teil der Geschäftslogikschicht und stellt wichtige Funktionalitäten für die *GUI Component* und *Cryptography Component* bereit. Diese Komponente umfasst folgende Services

TextConverterService Diese Klasse stellt einen Dienst zur Verfügung, um Texte in unterschiedliche Formate zu konvertieren. Einerseits wird diese Klasse verwendet, um die Eingaben der User in gewünschte Formate (z.B. UpperCase) zu formatieren. Andererseits wird dieses Service von den Verschlüsselungs-Services verwendet, um Strings in geeignete Formate zu konvertieren, welche anschließend verschlüsselt und entschlüsselt werden können.

EmailService Das *EmailService* umfasst alle Funktionen, welche zum Versenden von Mailnachrichten benötigt werden.

PDFService Die Anwendung umfasst die Möglichkeit verschlüsselte Nachrichten in Form eines anschaulichen PDF-Dokuments bereitzustellen. Eine automatische Konvertierung in ein geeignetes PDF-Format wird erzielt.

4.4 Beschreibung des Konvertierungsprozesses beim Verschlüsseln

Da sich das Projekt an die Sekundarstufe I richtet, wird Wert darauf gelegt, dass ein verständliches und sicheres Verfahren die User-Eingabe erleichtert. Unter sicher ist gemeint, dass Sonderzeichen, Großbuchstaben, Kleinbuchstaben und Umlaute als Eingabe akzeptiert werden. Der nachfolgend beschriebene Prozess berücksichtigt diese unterschiedlichen Zeichen und beschreibt den allgemeingültigen Ablauf von der Eingabe bis hin zur verschlüsselten Nachricht. Die Schritte werden am Beispiel Caesars in vereinfachter Form in der nachfolgenden Grafik dargestellt.

4.4 Beschreibung des Konvertierungsprozesses beim Verschlüsseln

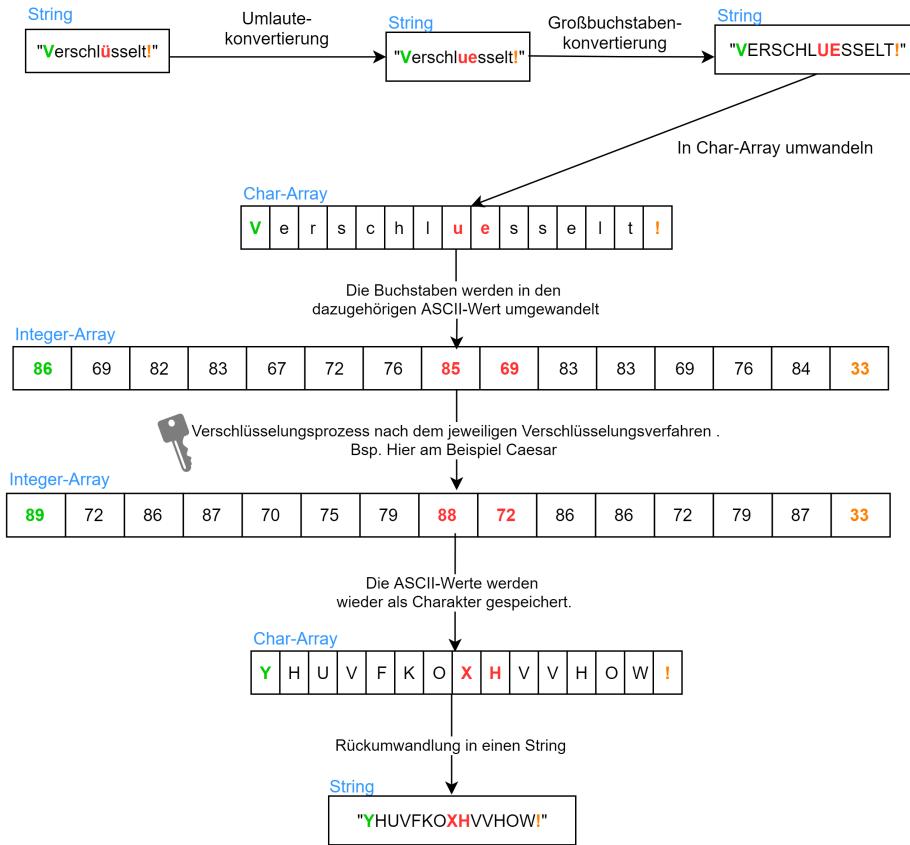


Abbildung 7: Konvertierungsprozess nach Eingabe eines Users

Nach Eingabe eines zu verschlüsselnden Textes findet als erster Schritt im Algorithmus eine Übersetzung aus dem deutschen ins lateinische Alphabet statt. Das heißt eine Ersetzung der Buchstaben ä, ö und ü durch ae, oe und ue wird durchgeführt. Außerdem werden die Kleinbuchstaben durch Großbuchstaben ersetzt. Im Anschluss daran wird der Umlaute-freie String in ein Char-Array umgewandelt, da zwischen diesen beiden Datenstrukturen (im Vergleich z.B. zu C/C++) in Java unterschieden wird. Dies ermöglicht nun Funktionen auf einzelne Zeichen anzuwenden [27].

Im nächsten Schritt des Algorithmus werden die Zeichen auf ihre dazugehörigen ASCII-Werte (American Standard Code for Information Interchange) abgebildet z.B. 'A' → 65. Die bijektive Abbildung wird als a bezeichnet.

Im weiteren Verlauf sei $G = \{65, \dots, 90\}$ die Menge der Großbuchstaben und $S = \{0, \dots, 127\} \setminus G$ die Menge der Sonderzeichen. Im Konvertierungsprozess wird die Unterteilung in die beiden Mengen benötigt, um den Verschlüsselungsprozess zu beschreiben. Infolgedessen kann für ein Zeichen bestimmt werden, ob es ein Großbuchstabe oder ein Sonderzeichen ist.

Durch den bisher beschriebenen Algorithmus können wir Zeichen durch ASCII-Werte ausdrücken, dadurch in \mathbb{N} operieren und zwischen den Mengen Großbuchstaben und Sonderzeichen unterscheiden. Nun wird mit der eigentlichen Verschlüsselung begonnen. Das gewählte Verschlüsselsverfahren wird durch die Funktion $c : \{0, 1, \dots, 25\} \mapsto \{0, 1, \dots, 25\}, x \rightarrow c(x)$ beschrieben. Dabei steht 0 für 'A', 1 für 'B', ..., 25 für 'Z'. Wie die Verschlüsselungsfunktion definiert ist, ist je nach Verfahren unterschiedlich. Für Caesar, Atbasch und die monoalphabetische Substitution sind die Funktionen c im Abschnitt *2.1 Datensicherheit durch Verschlüsselung* ([KRYPTOKNACK: Verschlüsselung](#)), S. 9 angegeben.

Für eine Zeichen mit ASCII-Wert x , sei \tilde{x} der verschlüsselte ASCII-Wert des Zeichens.

$$\tilde{x} = \begin{cases} x, & \text{wenn } x \in S \\ 65 + c(x - 65), & \text{wenn } x \in G \end{cases} \quad (4.1)$$

(4.2)

Mit Hilfe von a^{-1} , kann der verschlüsselte Wert wieder in ein ASCII-Zeichen umgewandelt werden. Im Anschluss werden die verschlüsselten Zeichen wieder zusammengesetzt und als String ausgegeben.

Bemerkung

Der Entschlüsselungs-Vorgang erfolgt ähnlich.

5 Funktionstests

Zum Testen der Anwendung wird das Konzept *A new Approach to Software Verification* von David J. Panzel verwendet. Der Softwareentwickler zeigt eine dreistufige Testpyramide auf (siehe Abbildung 8), die beschreibt auf welcher Softwareebene (Level) die Anwendung getestet werden sollte:

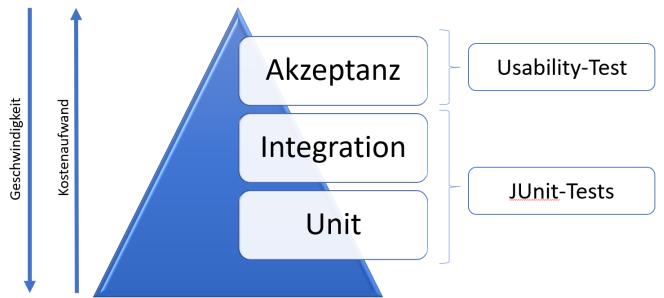


Abbildung 8: Testpyramide

- 1. Unit Testing:** Einzelne Units (Methoden bzw. Klassen) werden auf eine korrekte Funktionalität getestet.
- 2. Integration Testing:** Das **Zusammenspiel** verschiedener Units wird untersucht.
- 3. Acceptance Testing:** Es wird überprüft, ob die funktionalen Erwartungen vom System für den User im Gebrauch erfüllt werden.

Durch levelbasiertes Testen soll die Qualität der Software gesichert werden [21, S.5 f.]. Im Rahmen des Projekts werden mit Hilfe von *JUnit* die Levels *Unit Testing*, als auch *Integration Testing* abgedeckt. Der Fokus liegt dabei auf automatisierten Tests, welche das Backend testen. Anschließend wird zum Testen der Benutzeroberfläche ein *Akzeptanztest* mit der Zielgruppe der Anwendung durchgeführt, wobei hier die Usability eine wichtige Rolle spielt.

5.1 Integrations- und Systemtest

Das Testframework *JUnit* dient als Testgrundlage, um sowohl einzelne Klassen und Methoden auf das zu erwartende Verhalten zu testen, als auch die Interaktion zwischen den Units [15, S.175]. Basierend auf dem Entwurfskonzept *Layered Architecture* werden einerseits Tests auf Ebene der *Services* durchgeführt, andererseits einzelne *Controller*, somit auch die *Beans*, untersucht. Insgesamt stellt die Software 53 Testmethoden bereit.

Bei dem am 01.01.2020 durchgeführten Tests (mittels *IntelliJ IDEA*) liefen 53 von 53 Tests erfolgreich ab. Die dabei erzielte CodeCoverage in den einzelnen Testbereichen wird in Tabelle 8 genau illustriert. Für eine detaillierte Auflistung der Testergebnisse sei auf Unterabschnitt 8.2.1 verwiesen.

Bereich	Klasse	Methode	Zeile
Services	100% (10/10)	96% (48/50)	92% (210/227)
Controllers	100% (6/6)	92% (12/13)	93% (98/105)
Beans	100% (3/3)	95% (47/49)	96% (91/94)

Tabelle 8: CodeCoverage der Methoden, Klassen und Zeilen

Der Autor Jeff Langr verweist in *Pragmatic Unit Testing in Java 8 with JUnit*, dass ein Wert über 90 % bei Test-Driven-Design für eine umfassende Abdeckung spricht. Sowohl in Klassen-, Methoden- und Zeilenabdeckung wird dieser Wert überschritten. Der Vergleich mit dieser Referenz zeigt, dass das Testen der Units und das Integration Testing als ausreichend angesehen werden kann [25, S.194f].

5.2 Akzeptanztest

5.2.1 Untersuchungsdesign

Im Bereich der Oberflächen-Tests (Frontend) sieht das entwickelte Testkonzept die Durchführung eines Betatests vor. Bei diesem wird die vorläufige Betaversion des Softwareproduktes unter realen Bedingungen getestet [20, S.314]. Deshalb wird *KRYPTOKNACK* an einer Sekundarstufe I erprobt. Es wird untersucht, ob die angestrebten Benutzer*innenanforderungen im Gebrauch erfüllt sind. Darüber hinaus werden Erfahrungen im Bereich der Usability beim Akzeptanztest erhoben [33, S.139].

In einer einstündigen Unterrichtseinheit können die Schüler*innen die Anwendung ausprobieren. In Form eines *Black-Box-Prinzips*, d.h. es gibt kein festgelegtes Testskript, können sie die Software frei im Rahmen des Unterrichtsentwurfes nützen. Die vorbereitete Unterrichtsstunde sieht vor, dass sowohl die Bereiche *Verschlüsseeln*, als auch der *CodeKnacker* thematisch behandelt werden. Aus diesem Grund bearbeiten Schüler*innen Aufgaben aus dem inkludierten Aufgabenkatalog (siehe 8.1 *Arbeitsmaterial*, S. 51). Außerdem können die Schüler*innen die Anwendung frei testen.

Nach Abschluss der Einheit findet eine schriftliche Befragung über die Anwendung statt. Der bereitgestellte Fragebogen (siehe 8.2.2 *Fragebogen*, S. 64) umfasst 13 konkrete Aussagen in den Bereichen *Gesamteindruck* (6), *Verschlüsseln* (3) und *CodeKnacker* (4). Anhand einer fünfstufigen Skala bewerten die Schüler*innen die Aussagen. Dadurch soll festgestellt werden, ob die Verwendung der Software in der Praxis mit der derzeitigen entwickelten Version möglich ist [31, S.62]. Des Weiteren umfasst der Fragebogen vier offene Fragestellungen, um weiteres Feedback von den Anwender*innen zu erhalten. Diese offenen Fragen beziehen sich insbesondere darauf, warum Schüler*innen die Arbeit mit der Anwendung gefällt oder missfällt und was sie durch die Nutzung gelernt haben.

5.2.2 Durchführung der Untersuchung

Die durchgeführte Untersuchung fand am 10.10.2019 am BG/BRG Sillgasse im Modul Informatik statt. Die Probanden*innen waren zwischen 13 und 14 Jahren alt. Insgesamt nahmen 5 Schülerinnen und 10 Schüler teil. Die Unterrichtseinheit wurde von mir geleitet. Nach einer 5-minütigen Initialphase, die keine Informationen über die Bedienung der Anwendung beinhaltete, begannen die Schüler*innen mit einer 30-minütigen Bearbeitungsphase der Aufgaben. Da ein Teil der Klasse frühzeitig fertig wurde, beschäftigten sie sich frei mit der Software. Im Anschluss hatten die Schüler*innen 15 Minuten Zeit den Fragebogen auszufüllen.

5.2.3 Auswertung der Ergebnisse

Bereich Verschlüsseln

Beim Verschlüsseln von Texten zeigten sich wenige Schwierigkeiten. In zwei Fällen wurden Performance-Probleme (Leistung) der Applikation festgestellt. Einmal dauerte das serverseitige Verschlüsseln etwas länger, ein anderes Mal brauchte es einige Zeit um Mails zu versenden. Bei der Analyse der Ursache konnte festgestellt werden, dass dies nicht auf das Programm zurückzuführen war, sondern auf die Hardware (RaspberryPi). Auf einem leistungsfähigeren Server ist davon auszugehen, dass das Endprodukt keine Probleme bei der Performance aufweist.

Verschlüsselung	Aussage							
		stimme voll zu	stimme eher zu	unentschieden	stimme eher nicht zu	stimme gar nicht zu	keine Aussage	ungültig
	Ich konnte alle gewünschten Texte eingeben.	13	2	0	0	0	0	0
	Ich konnte alle gewünschten Texte verschlüsseln.	14	1	0	0	0	0	0
	Es traten beim Entschlüsseln keine Fehler auf.	12	2	0	1	0	0	0

Abbildung 9: Auswertung Bereich Verschlüsselung

Bereich CodeKnacker

Bei Auswertung des Fragebogens ist bei der Aussage *Die grafische Darstellung (Diagramm) unterstützt mich beim Finden des Schlüssels* eine negative Tendenz im Vergleich zu den anderen formulierten Aussagen bemerkbar. Da für 11 von 15 Teilnehmer*innen der Codeknacker übersichtlich gestaltet ist, kann vermutet werden, dass dies nicht auf die grafische Repräsentation zurückzuführen ist. Als mögliche Ursache könnte eine fehlende Erklärung seitens der Anwendung oder seitens der Lehrperson angeführt werden. Eine Anpassung in der Applikation wurde deshalb vorgenommen.

	stimme voll zu	stimme eher zu	unentschieden	stimme eher nicht zu	stimme gar nicht zu	keine Aussage	ungültig
Aussage							
Die grafische Darstellung (Diagramm) unterstützt mich beim Finden des Schlüssels.	5	3	3	2	1	0	0
Ich konnte schrittweise den passenden Schlüssel erarbeiten.	11	3	1	0	0	0	0
Der CodeKnacker ist übersichtlich gestaltet, ich finde mich gut zurecht.	11	3	1	0	0	0	0
Der CodeKnacker funktioniert ohne Probleme, es treten keine Fehler auf (visuell und funktionell).	9	4	2	0	0	0	0

Abbildung 10: Auswertung Bereich CodeKnacker

Bereich Gesamteindruck

Im Bereich Gesamteindruck ist der Großteil der Probanden*innen zufrieden mit dem Design der Anwendung. Auch die farbliche Abstimmung findet keine geschlechtsspezifische Tendenz bei der Zu- bzw. Abneigung. Obwohl bei der Einführung auf die Bedienung der Anwendung verzichtet wurde, gaben 10 von 15 Schüler*innen an, dass sie sich rasch im Umgang mit der Software zuretfanden, die restlichen 5 Probanden*innen stimmten eher zu. Deshalb kann vermutet werden, dass die im Konzeptentwurf festgehaltenen Punkte (Use-Cases und Beschreibung) zum Großteil erfüllt sind.

Gesamteindruck	Aussage							
		stimme voll zu	stimme eher zu	unentschieden	stimme eher nicht zu	stimme gar nicht zu	keine Aussage	ungültig
	Mir gefallen die Farben und das Layout.	9	5	1	0	0	0	0
	Die Bilder finde ich passend gewählt.	10	5	0	0	0	0	0
	Die Startseite (erste Seite beim Laden) ist übersichtlich.	11	3	1	0	0	0	0
	Ich finde schnell die Bereiche, die ich suche.	10	5	0	0	0	0	0
	Ich verstehe die Texte gut.	12	2	1	0	0	0	0
	Mir gefällt die gesamte Anwendung.	9	3	2	0	0	0	1

Abbildung 11: Auswertung Gesamteindruck

Bereich offene Fragen

Das freie Feedback in Form von Fragen spiegelt einen positiven Eindruck gegenüber der Web-Anwendung wieder. Bemerkungen wie „mir hat das Entschlüsseln des Codes viel Spaß bereitet“ oder „dass es sehr übersichtlich ist und man gut zurecht kommt“ bestätigen die Eindrücke aus den Bereichen Gesamteindruck, Verschlüsseln und CodeKnacker. Bei der Frage „Was hast du heute gelernt?“ zeigte sich ein weites Spektrum an Antworten. Die gesamte Auflistung ist in *8.2.2 Antworten auf offenen Fragen, S. 65* aufgelistet zu finden. Zusammengefasst beziehen sich die Aussagen der Schüler*innen insbesondere auf

- Wissen über einzelne Verfahren,
- wie man Texte verschlüsselt bzw. entschlüsselt,
- wie und wo Verschlüsselung verwendet wird und
- wie man Geheimtexte knackt.

6 Reflexion und Nachwort

Am Beginn meines Bachelorprojektes startete ich recht zügig mit der Programmierung, um einzelne Komponenten der Anwendung zu erstellen. Diese Herangehensweise ohne konkretes Konzept scheiterte. Die Schnittstellen zwischen den entwickelten Softwarekomponenten waren mangelhaft. Das „unstrukturierte Chaos“ funktionierte miteinander nicht. Ich habe etliche Arbeitsstunden verschenkt.

Aus diesem Grund beschloss ich einen Neuanfang zu starten. Ich erstellte ein neues umfassendes Softwarekonzept und legte Schnittstellen zwischen den Komponenten fest. Ich merkte, dass diese Vorgehensweise der richtige Ansatz war. Durch den gewonnenen Überblick gelang es mir deutlich effizienter zu entwickeln. In Zukunft lege ich mehr Wert auf eine gute Planung am Beginn.

Die Motivation zum Programmieren verringerte sich, als ich mich an die Frontend-Entwicklung (User-Interface) setzte. Bugs über Bugs, neue Konzepte (XHTML, JavaScript,...), sowie meine geringe Erfahrung im Frontend-Bereich stellten mich vor einer großen Herausforderung. Es benötigte einige zeitlichen Aufwand, um mich mit der Thematik vertraut zu machen, jedoch habe ich nun das Wissen und Können aus diesem Entwicklungsbereich, um dies für weitere Projekte nutzen zu können.

Bei der didaktischen Umsetzung bereitete mir großes Kopfzerbrechen, wie ich es schaffen könnte, mathematische Themen in *KRYPTOKNACK* umzusetzen. Dieser Meilenstein war jedoch zu meinem Erstaunen recht einfach erreichbar. Das Buch von Thomas Borys: *Codierung und Kryptologie* war dabei eine sehr große Unterstützung. Die „Vorarbeit“, die der Autor leistete, half mir mathematische Themenschwerpunkte für meine Anwendung zu finden. Es wurde deutlich, wie wichtig eine umfassende

wissenschaftliche Recherche ist. Nur dadurch konnte ich auf bereits bestehende wissenschaftlich erarbeitete Inhalte aufbauen und in meinem Projekt umsetzen.

Besonders positive Erfahrungen sammelte ich beim Testen der Software, trotz des Mehraufwandes im Projekt. Durch mein verwendetes Testkonzept, entdeckte ich sowohl Softwarefehler, als auch Usability-Schwierigkeiten im Umgang mit **KRYPTOKNACK**. Dementsprechend konnte ich durch das Testfeedback passende Adaptionen vornehmen, um so die Qualität meiner Anwendung zu erhöhen.

Im Großen und Ganzen bin ich sehr zufrieden damit, wie mein Bachelorprojekt verlaufen ist. Ich bin stolz auf meine Arbeit, die ich geleistet habe und was aus dieser Projektarbeit am Ende herauskam. Die Einladung beim *ARGE-Informatik-Vernetzungstreffen* Informatiklehrer*innen aus ganz Tirol **KRYPTOKNACK** vorzustellen, war eine krönender Abschluss meines Projektes. Das positive Feedback, dass Lehrer*innen gerne die Anwendung im Unterricht verwenden möchten und sie Interesse hätten, die Anwendung auch auf die Sekundarstufe II auszuweiten, freute mich sehr.

7 Danksagung

Am Ende möchte ich jenen danken, die zum Gelingen meines Projekts beigetragen haben. Besonderen Dank gilt meiner Betreuerin Karin Schnass und meinem Betreuer Tobias Hell, die mir bei so manchem Problem rasch und hilfreich zu Seite standen. Auch möchte ich mich bei Harald Pietersteiner für die Möglichkeit bedanken in seiner Klasse meine Anwendung zu testen.

Für die seelische Unterstützung, den aufmunternden Worten, wenn ich zweifelte, die Sorge um mein körperliches Wohl und die regelmäßige Versorgung mit Gummibärchen, dafür möchte ich meinen Freunden und insbesondere meiner Familie danken.

Literaturverzeichnis

- [1] *Harness the power of PDF*. <https://itextpdf.com/en>. Version: 2019
- [2] ALESSANDRO, Nassiri: *Enigma: Museo nazionale della scienza e della tecnologia Leonardo da Vinci*. [https://de.wikipedia.org/wiki/Datei:Enigma_\(crittografia\)_-_Museo_scienza_e_tecnologia_Milano.jpg](https://de.wikipedia.org/wiki/Datei:Enigma_(crittografia)_-_Museo_scienza_e_tecnologia_Milano.jpg). Version: 2012
- [3] BECK, Kent: *JUnit Pocket Guide*. Sebastopol : O'Reilly Media, 2004. – ISBN 0-596-00743-4
- [4] BIERMANN, Kai ; MONDIAL, Sebastian: Sony Pictures Entertainment: Warum der Sony-Hack zum Gau wurde. In: *Zeit* (2014). <https://www.zeit.de/digital/datenschutz/2014-12/sony-spe-hack-daten>
- [5] BISCHOFF, Jürgen: Operation Enigma - der stille Kampf der Codebrecher: Zweiter Weltkrieg. <https://www.geo.de/magazine/geo-epochen-kollektion/20905-rtkl-zweiter-weltkrieg-operation-enigma-der-stille-kampf-der>
- [6] BORYS, Thomas: *Codierung und Kryptologie: Facetten einer anwendungsorientierten Mathematik im Bildungsprozess*. Wiesbaden : Vieweg+Teubner Verlag / Springer Fachmedien Wiesbaden GmbH Wiesbaden, 2011. <http://dx.doi.org/10.1007/978-3-8348-8252-3>. – ISBN 978-3-8348-1706-8
- [7] BREU, Ruth: *Objektorientierter Softwareentwurf: Integration mit UML*. Berlin and Heidelberg : Springer, 2001. <http://dx.doi.org/10.1007/978-3-642-56600-4>. – ISBN 3-540-41286-7 <http://dx.doi.org/10.1007/978-3-642-56600-4>.

- [8] BUNDESMINISTERIUM FÜR BILDUNG, WISSENSCHAFT UND FORSCHUNG: *Lehrplan Mathematik der AHS-Unterstufe*. https://bildung.bmbwf.gv.at/schulen/unterricht/lp/ahs14_789.pdf?61ebzm
- [9] BUNDESMINISTERIUM FÜR BILDUNG, WISSENSCHAFT UND FORSCHUNG: *Digitale Grundbildung*. <https://www.bmbwf.gv.at/Themen/schule/zrp/dibi/dgb.html>. Version: 13.03.2019
- [10] BUNDESMINISTERIUM FÜR BILDUNG, WISSENSCHAFT UND FORSCHUNG: *Änderung der Verordnung über die Lehrpläne der Neuen Mittelschulen sowie der Verordnung über die Lehrpläne der allgemeinbildenden höheren Schulen*. https://www.ris.bka.gv.at/Dokumente/BgblAuth/BGBLA_2018_II_71/BGBLA_2018_II_71.pdfsig. Version: 2018-04-19
- [11] CASTRO, Elizabeth: *HTML, XHTML & CSS: HTML, XHTML & CSS - Der Meisterkurs*. München : Markt & Technik Verl., 2009. – ISBN 978-3827244666
- [12] CURZON, Paul ; McOWAN, Peter W.: *Computational Thinking: Die Welt des algorithmischen Denkens - in Spielen, Zaubertricks und Rätseln*. Berlin : Springer, 2018. – ISBN 978-3-662-56773-9
- [13] DROBNA, Daniela ; PARVANOVA, Nona ; SCHUBERT, Maximilian ; ISPA – INTERNET SERVICE PROVIDERS (Hrsg.): *Internet sicher nutzen: Ein Leitfaden im Rahmen der Saferinternet.at-Initiative*. https://www.saferinternet.at/fileadmin/categorized/Materialien/ISPA_Internet_sicher_nutzen.pdf. Version: 2015
- [14] DUNKEL, J. ; HOLITSCHKE, A.: *Softwarearchitektur für die Praxis*. Springer Berlin Heidelberg, 2013. – ISBN 9783642555527
- [15] EDELSON, Justin ; LIU, Henry: *JRuby cookbook: Ruby meets Java*. Sebastopol, Calif. : O'Reilly, 2009. – ISBN 978-0-596-51980-3
- [16] ESSLINGER, Bernhard: *Kryptographie lernen und anwenden mit CrypTool und SageMath*. 12. Heilbronn/Siegen, 2018 <https://www.cryptool.org/images/ctp/documents/CT-Book-de.pdf>

- [17] FILLER, A.: *Einführung in die Mathematikdidaktik, Teil 2: Einige lerntheoretische Grundlagen und daraus resultierende Prinzipien für den Mathematikunterricht.* Berlin, 2015/2016 <http://didaktik.mathematik.hu-berlin.de/files/einfmadid2lernpsych.pdf>
- [18] FLANAGAN, David: *JavaScript: Das umfassende Referenzwerk.* 3. Aufl. Köln : O'Reilly Verlag, 2007 (Safari Books online). – ISBN 9783897214910
- [19] FREIERMUTH, Karin ; HROMKOVIČ, Juraj ; KELLER, Lucia ; STEFFEN, Björn: *Einführung in die Kryptologie: Lehrbuch für Unterricht und Selbststudium.* 1. Wiesbaden : Vieweg + Teubner, 2010. <http://dx.doi.org/10.1007/978-3-8348-9671-1>
<http://dx.doi.org/10.1007/978-3-8348-9671-1>
- [20] GRECHENIG, Thomas: *Softwaretechnik: Mit Fallbeispielen aus realen Entwicklungsprojekten.* München : Pearson Studium, 2010. – ISBN 978-3-86894-007-7
- [21] GULATI, Shekhar ; SHARMA, Rahul: *Java Unit Testing with JUnit 5: Test Driven Development with JUnit 5.* Berkeley, CA : Apress L. P, 2017. – ISBN 978-1-4842-3014-5
- [22] HAFTENDORN, Dörte: *Mathematik sehen und verstehen: Werkzeug des Denkens und Schlüssel zur Welt.* 3. Berlin, Heidelberg : Springer Berlin Heidelberg, 2019. – ISBN 978-3-662-58136-0
- [23] JENDRYSCHIK, Michael: *Einführung in XHTML, CSS und Webdesign.* 2. München : Addison-Wesley Verlag, 2009 (Safari Books). – ISBN 978-3-8273-2739-0
- [24] JOHNSON, Rod: *Professional Java development with the Spring Framework.* Indianapolis, IN : Wiley Pub, 2010. – ISBN 978-0-7645-7483-2
- [25] LANGR, Jeff: *Pragmatic unit testing in Java 8 with JUnit.* Raleigh, NC : The Pragmatic Bookshelf, 2015 (The pragmatic programmers). – ISBN 978-1-94122-259-1
- [26] LECHNER, Bettina K. ; STOCKMANN, Bernhard: *CSS pur! Ultimative Weblösungen mit Stil.* München : Addison-Wesley, 2010
- [27] MANOJ, Dannana: *Difference between String and Character Array in Java.* <https://www.geeksforgeeks.org/difference-between-string-and-character-array-in-java/>

- [28] MATEVSKA, Jasminka: *Rekonfiguration komponentenbasierter Softwaresysteme zur Laufzeit*. 1. Vieweg + Teubner, 2010
- [29] MEINEL, Christoph ; SACK, Harald: *WWW: Kommunikation, Internetworking, Web-Technologien*. Berlin : Springer, 2013 (Xpert.press). – ISBN 978-3-642-62384-4
- [30] PETER, Ira-Katharina ; PETERMANN, Franz: *Klinische Kinderpsychologie*. Bd. 15: *Cybermobbing im Kindes- und Jugendalter*. 1. Göttingen : Hogrefe, 2018. – ISBN 978-3-8409-2915-1
- [31] PUSULURI, N. R.: *Software Testing Concepts And Tools*. Dreamtech Press, 2006. – ISBN 9788177227123
- [32] SCHÄFER, Georg E.: *Digitalisierung braucht Vertrauen, das auf Verständnis gründet: Technische, juristische und weltanschauliche Fragen der Quantencomputer, neuronalen Netze, Algorithmen und Künstlichen Intelligenz*. Norderstedt : Books on Demand, 2018. – ISBN 3746093546
- [33] SCHÄUFFELE, Jörg: *Automotive Software Engineering*. Wiesbaden : Springer Fachmedien, 2010 (ATZ / MTZ-Fachbuch). – ISBN 978-3-8348-0364-1
- [34] SCHMEH, Klaus: *Kryptografie: Verfahren, Protokolle, Infrastrukturen*. Heidelberg : dpunkt.verlag, 2016 (iX Edition). – ISBN 978-3-86490-356-4
- [35] SCHULZ, André: *Wie britische Schachspieler halfen, den Krieg zu gewinnen: Enigma-Entschlüsselung*. <https://www.spiegel.de/einestages/enigma-entschlüsselung-wie-schachspieler-halfen-den-krieg-zu-gewinnen-a-1239487.html>. Version: 24.11.2018
- [36] SINGH, Simon: *Dtv. Bd. 33071: Geheime Botschaften: Die Kunst der Verschlüsselung von der Antike bis in die Zeiten des Internet*. 13. München : dtv, 2016
- [37] SPITZ, Stephan ; PRAMATEFTAKIS, Michael ; SWOBODA, Joachim: *Kryptografie und IT-Sicherheit: Grundlagen und Anwendungen*. 2. Wiesbaden : Vieweg+Teubner Verlag / Springer Fachmedien Wiesbaden GmbH Wiesbaden, 2011. – ISBN 978-3-8348-1487-6

Literaturverzeichnis

- [38] SRIRANGAN: *Apache Maven 3 cookbook*. Birmingham : Packt Open Source, 2011. – ISBN 978-1-849512-44-2
- [39] STELING, Stephen ; MAASEN, Olav: *Applied Java Patterns*. Englewood Cliffs, NJ : Prentice Hall, 2002 (Safari Books online). – ISBN 0-13-093538-7
- [40] SUETONIUS TRANQUILLUS, Gaius ; WITTSTOCK, Otto: *Schriften und Quellen der Alten Welt*. Bd. 39: *Kaiserbiographien*. Berlin : Akademie-Verlag, 1993. – ISBN 3050018445
- [41] WOLFF, Eberhard: *Spring 3: Framework für die Java-Entwicklung*. 3. Heidelberg : dpunkt.verlag, 2011

Abbildungsverzeichnis

1	Logo	5
2	Übermitteln und Verschlüsseln eines Datenfragmentes	8
3	Exemplar einer <i>Enigma</i> [2]	15
4	Beispiele zu den Darstellungsformen von Funktionen in <i>KRYPTO-KNACK</i> : Tabellen, Texte und Bilder	21
5	Use-Case-Diagramm	25
6	Klassendiagramm	29
7	Konvertierungsprozess nach Eingabe eines Users	33
8	Testpyramide	35
9	Auswertung Bereich Verschlüsselung	38
10	Auswertung Bereich CodeKnacker	39
11	Auswertung Gesamteindruck	40
12	Gesamte Auswertung der Fragebogen	65

Tabellenverzeichnis

2	Beispiel eines Klartext- und Geheimtextalphabets einer freien Buchstabensubstitution	10
4	Klartext- und Geheimtextalphabet beim Atbasch-Verfahren	11
6	Klartext- und Geheimtextalphabet bei der Caesar-Chiffre	12
7	Relative Häufigkeit und Stabilisierungscharakter	22
8	CodeCoverage der Methoden, Klassen und Zeilen	36

8 Anhang

8.1 Arbeitsmaterial

8.1.1 Einleitung und Erklärung

1. Einleitung und Erklärung

Protagonisten



Name: KryptoWoman GLORIA

Alter: 13 Jahre

Hobbies: UNO spielen, klettern, Computer spielen

Lieblingsfarbe: orange

Lieblingsyoutuber_in:
DagiBee

Fähigkeit: Knobel-Genie



Name: KryptoMan MAX

Alter: 12 Jahre

Hobbies: Fußball, zeichnen, Rätsel lösen

Lieblingsfarbe: rot

Lieblingsyoutuber_in: Rezo

Fähigkeit: Schnelle Auffassungsgabe

KryptoWoman GLORIA und **KryptoMan MAX** sind begeistert von Kryptografie – dem Verschlüsseln und Entschlüsseln von Texten. Gemeinsam lösen sie Rätsel, knacken Geheimbotschaften und entwickeln neue Geheimschriften. Da Verschlüsselung nicht nur auf Papier stattfindet, möchten die beiden ihre Nachrichten auch am Handy für andere geheim halten. Glorias großer Bruder Benedikt ist ihnen beim Verschlüsseln behilflich und hat dafür eine App programmiert. Dieses Tool nützt Gloria und Max, um ihre Botschaften zu verschlüsseln und entschlüsseln, ein Geheimtext kann damit geknackt werden. Versucht nun nachfolgende Aufgaben zu lösen. Viel Spaß und Erfolg wünschen euch KryptoWoman Gloria und KryptoMan Max!

Bereiche der Anwendung

Information



Hier erhältst du weitere Informationen zum Verschlüsselungs-Verfahren, sowie Beispiele und Erklärungen.

Verschlüsselung

[Verschlüsselung](#)

Hier kann zu den jeweiligen Verschlüsselungsverfahren gewechselt werden. Das Verschlüsseln von Texten, die Erstellung von PDF-Dokumenten und das Versenden von Geheimtexten wird unterstützt.

[!\[\]\(1f8c92b8e1c45c0c7161a9bdf0663761_img.jpg\) Substitution](#)

CodeKnacker



Der *CodeKnacker* ist ein Lern-Unterstützungs-Tool, um Geheimtexte zu analysieren und passende Schlüssel zu finden.

8.1.2 Atbasch Verfahren

2. Atbasch-Verfahren

2.1. Aufgabe



Abb. 1: Gelehrter aus Palästina

Am Abend sitzt Max mit seiner Mutter und seinem Vater im Wohnzimmer. Sein Vater liest in einem Buch über die Gelehrten von Palästina aus der Zeit 600 v.Chr.. Max fragt ihn: „Warum interessiert dich dieses alte Buch?“. Der Vater antwortet ihm, dass das Buch über die „Geheimschrift“ Atbasch handle. Max ist baff, dass es vor 2600 Jahren Geheimschriften gab.

Der Vater schmunzelt und gibt Max ein Rätsel auf, das dieser mithilfe des **Atbasch Verfahrens** lösen solle. Als Tipp gibt der Vater: Der erste ist der letzte Buchstabe, der zweite der vorletzte Buchstabe usw.

Geheimtext: „DVMM WZH QVNZMW ORVHG, DFIWV WZH ZGYZHS-EVIUZSIVM TVPMZXPG. EVIWZNNG!“

Auftrag: Entschlüssle den Geheimtext von Max Vater, indem du jeden Buchstaben durch einen passenden Buchstaben ersetzt.

2.2. Aufgabe

Am nächsten Tag schickt Max Gloria eine Nachricht, welche er mit Hilfe des Atbasch-Verfahrens verschlüsselt. Folgende Nachricht möchte Max verschlüsseln:

„Gloria, ich habe eine 2600 Jahre alte Nachricht geknackt“

Klartextalphabet	A	B	C	D	E	F	...	X	Y	Z
Schlüsseltextalphabet	Z	Y	X	W	V	U	...	C	B	A

Auftrag:

- 1) Verschlüssle die Klartext-Nachricht mit Hilfe des Atbasch-Verfahrens.
- 2) Verwende zur Kontrolle die Kryptografie-Anwendung und verschlüssle dazu den Text.

8.1.3 Caesar Verfahren

3. Caesar-Verfahren

3.1. Aufgabe (PAIR)

Während eines Schulbesuchs in einem Museum finden Max und Gloria eine jahrhundertealte Caesar-Scheibe.

Mit solch einer Scheibe können Klartexte mit Hilfe des Caesar-Verfahrens verschlüsselt werden. Neben der Caesar-Scheibe steht ein Computer. Dort kann man einen Text eingeben. Dieser Text wird mit dem Caesar-Verfahren verschlüsselt und kann per Email versendet werden.

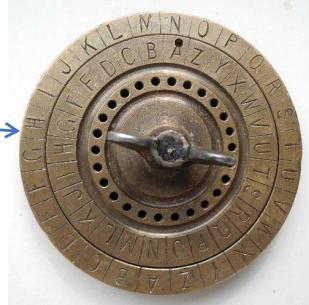


Abb.: Caesar-Scheibe

Auftrag:

1. E-Mail versenden:

- 1.1. Öffne die Kryptografie-Anwendung, wechsle auf den Tab *Caesar (Verschlüsselung)*
- 1.2. Gib einen Text ein, welchen du an deinen Teampartner verschicken möchtest. Der Text sollte mindestens 50 Zeichen lang sein.
- 1.3. Wähle einen Schlüssel zwischen 1 und 26.
- 1.4. Versende die Nachricht per Email.

2. Code knacken:

- 2.1. Jeder im Team sollte die Nachricht mit der Geheimbotschaft der _ des anderen erhalten haben. Nun versucht die Geheimbotschaft zu knacken. Findet dazu mit Hilfe des *CodeKnackers* den passenden Schlüssel.

3.2. Aufgabe

Max und Gloria finden im Museum auf einer Informationstafel einen Geheimtext und den dazugehörigen Klartext. Es wurde das Caesar-Verfahren verwendet. Sie fragen sich welcher Schlüssel verwendet wurde.

Geheimtext FQQJ BJLJ KZJMWJS SFHM WTR.

Klartext: ALLE WEGE FÜHREN NACH ROM.

Auftrag: Finde einen passenden Schlüssel $x \in \{0, \dots, 25\}$.

8.1.4 Substitutions-Verfahren

4. Substitutions-Verfahren

4.1. Aufgabe

In einem Feriencamp nehmen Gloria und Max an einem Wettbewerb teil, sie müssen einen Geheimtext knacken. Wenn sie x Minuten benötigen, erhalten sie

- 3 Punkte falls $x \leq 20$ min
- 2 Punkte falls $20 < x < 30$ min
- 1 Punkt sonst

Aufgabe: Jeder Buchstabe wurde durch einen anderen Buchstaben verschlüsselt. Entschlüssle nachfolgenden Text:

„YZBY YZBKJYNGJBO ZB XZY MZHRYBHNUKI LCA LYGHWNTJYHHYTB LYGVYGOYB JBX LYGNYZATZW NYB CNBY UTTY OYNYZABZHRGUYAGYZ UVYG BZWNM CNBY NZBITGTZHIZOYB HWNUTR XUGOYHIYT TI SJA BJISYB JBX YGOCYISYB XYH UTTOYAZBYB DJVTZRJAH“¹

Auftrag: Auch du nimmst am Wettbewerb teil, knacke den Geheimtext und verwende dazu den *Codeknacker*.

¹ Quelle: URL: <http://www.mathe.tu-freiberg.de/~hebisch/cafe/kryptographie/decodierauf.pdf> (Zugriff am 23.07.2019)

8.1.5 Themenschwerpunkt: Stabilisierungscharakter bei der Häufigkeitsanalyse

5.1. Stabilisierungscharakter bei der Häufigkeitsanalyse

Max und Gloria erhalten drei verschiedene Texte von ihrem Lehrer, welche alle mit Caesar und dem gleichen Schlüssel chiffriert wurden.

- **Text 1:** 5 Buchstaben (datei1.txt)
- **Text 2:** 30 Buchstaben (datei2.txt)
- **Text 3:** 1 000 Buchstaben (datei3.txt)
- **Text 4:** 10 000 Buchstaben (datei4.txt)

Beide versuchen den passenden Schlüssel durch Häufigkeitsanalyse bei den unterschiedlichen Textlängen herauszufinden. Am Ende kommt Max zum Schluss:

„Bei Text1 und Text2 war es schwieriger den Schlüssel zu finden, während bei Text3 bzw. Text4 der Schlüssel relativ schnell klar war“

Auftrag:

- Wie kommt Max zu dieser Aussage?
- Vergleiche die relative Häufigkeit des am häufigsten vorkommenden Buchstabens - mit der zu erwartenden Häufigkeit des Buchstabens e: 17.4%. Was fällt dir auf?

Bsp. Der am häufigsten vorkommende Buchstabe d kommt 5-mal im Text vor, der Text hat insgesamt 30 Buchstaben, d.h. der Buchstabe d hat eine relative Häufigkeit von

$$\frac{\text{Anzahl des Buchstabens } d}{\text{Anzahl der Buchstaben im Text}} = \frac{5}{30} = 0.166666 \dots \approx 16.6\%$$

Anschließend vergleicht man den ermittelten Wert $\frac{5}{30}$ mit dem zu erwarteten Wert 17.4%.

- Fülle nachfolgende Liste aus

Text	Text 1	Text 2	Text 3	Text 4
der(die) am häufigsten vorkommende(n) Buchstabe(n)				
Anzahl				
Relative Häufigkeit (in Prozent)				

- Warum fällt es ihm leichter bei Text3 bzw. Text4 einen Schlüssel zu finden als bei Text1 bzw. Text2?

8.1.6 Themenschwerpunkt: Sicherer Datenaustausch

5.2. Sicherer Datenaustausch

Gloria und Max quatschen gerne während der Unterrichtsstunde. Der Deutschlehrer Herr Mayer ist davon nicht begeistert, deshalb setzt er die beiden voneinander weg. Nun schmieden Gloria und Max einen Plan, wie sie trotz räumlicher Entfernung miteinander kommunizieren können.

- **Gloria:** Hey Max, wir könnten geschriebene Nachrichten miteinander austauschen. Unsere Mitschüler*innen geben die Briefe an uns weiter.
- **Max:** Das ist eine gute Idee! Ich sehe nur ein Problem: Die anderen sind neugierig und werden unsere Nachrichten lesen wollen.
- **Gloria:** Oje, daran habe ich nicht gedacht.
- **Max:** Kein Problem. Wie verschlüsseln unsere Nachricht mit *Atbasch*, dann kann niemand die Nachricht lesen, auch unser Lehrer nicht!

In einer Deutschstunde probieren sie ihren Plan aus:

Max schreibt	Ist Lisa in Jonas verliebt?
Max verschlüsselt	
Gloria antwortet	QZ, ORHZ SZG VH NRI VIAZVSOG. ZYVI WF WZIUHG MRXSGH DVRGVIHZTVM!
Max entschlüsselt Glorias Nachricht	

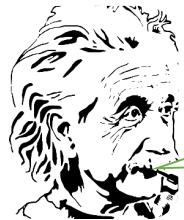
Auftrag:

1. Verschlüsse die Nachricht von Jonas und entschlüssle Glorias Nachricht (CodeKnacker).
2. Warum könnte es wichtig sein, dass auch digitale Nachrichten/Daten (z.B. WhatsApp-Nachrichten, Online-Banking) verschlüsselt werden?
3. Interpretiere nachfolgendes Bild?



8.1.7 Themenschwerpunkt: Sicherheit ist relativ

5.3. Sicherheit ist relativ



Wenn die Entwicklung der Quantencomputer weiter voranschreitet, sind unsere heutigen Verschlüsselungsverfahren (z.B. bei Whatsapp ...) (bald) nicht mehr sicher.

Max und Gloria grübeln über diese Aussage und fragen Max Vater nach Rat. Dieser meint:

„Lange glaubte auch Caesar sein Verfahren sei nicht „knackbar“. Das stellte sich als Irrtum heraus. Schon ein paar Jahre nach dem Caesar das Verfahren entwickelte, gelang es Gelehrten die Caesar-Verschlüsselung zu knacken. Ich habe eine Aufgabe für euch.“

Er übergibt ihnen einen Zettel:

TRIGV UZVD! ELKQV UVÉ KRX!

Auftrag:

1. Knacke das Verfahren. Finde den passenden Schlüssel.
2. Wie lautet die entschlüsselte Nachricht?
3. Stimmen folgende Aussagen? Kreuze die **richtigen** Aussagen an:
 - Unsere Daten sind immer sicher, das heißt niemand kann sie jemals „knacken“.
 - Ich achte darauf welche Daten ich öffentlich zugänglich mache. Es besteht immer die Gefahr, dass meine Daten geknackt werden.
 - Es gibt kein 100% sicheres Verfahren.
 - Quantentechnologie könnte in Zukunft ein Problem für die derzeit verwendeten Verfahren werden.

8.1.8 Lösungen

6. Lösungen

- 2.1. WENN DAS JEMAND LIEST, WURDE DAS ATBASCH-VERFAHREN GEKNACKT. VERDAMMT!
2.2. TOLIRZ, RXS SZYV VRMV 2600 QZSIV ZOGV MZXSIRXSG TVPMZXPG

3.2. Schlüssel 5

4.1. EINE EINFUEHRUNG IN DIE WISSENSCHAFT VOM VERSCHLUESSELN VERBERGEN UND
VERHEIMLICHEN OHNE ALLE GEHEIMNISKRAEMEREI ABER NICHT OHNE HINTERLISTIGEN SCHALK
DARSTELLT ZUM NUTZEN UND ERGOETZEN DES ALLGEMEINEN PUBLIKUMS.

A	B	C	D	E	F	G	H	I	J	K	L	M
M	N	O	P	-	-	R	S	T	U	F	V	W
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
H	G	-	-	K	Z	L	A	B	C	D	E	I
NICHT verwendet: - - - - - J - - - - - Q - - - - - X Y -												
Verwendet: A B C D E F G H I - K L M N O P - R S T U V W - - Z												

- 5.1. Mit zunehmender Anzahl an Buchstaben stabilisiert sich die relative Häufigkeit. Den Schlüssel zu finden wird einfacher.

	Text	Text 1	Text 2	Text 3	Text 4
Gesamtzahl an Buchstaben	5	30	1 000	10 000	
der(die) am häufigsten vorkommende(n) Buchstabe(n)	O,F,P, T,G,	I,N,R,S	J	J	
Anzahl der(die) am häufigsten vorkommende(n) Buchstabe(n)	1	3	166	1690	
Relative Häufigkeit (in Prozent)	20 %	10%	16,6%	16,9	

- 5.2. Max verschlüsselt: RHG ORHZ RM QLMZH EVIORVYG?

Max entschlüsselt Glorias Nachricht: JA, LISA HAT ES MIR ERZAEHLT. ABER DU DARFST NICHTS WEITERSAGEN!

5.1.

1. Schlüssel: 17
2. CARPE DIEM! NUTZE DEN TAG!
3. Richtig
4. Falsch – richtig – richtig – richtig

8.2 Funktionstest

8.2.1 Testergebnisse des Integrations- und Systemtests (01.01.2020)

An dieser Stelle werden genauere Informationen über die Test-Durchführung am 01.01.2020 um 13:26 angeführt

- * Test-Programm: IntelliJ IDEA coverage runner
- * 53 von 53 Test liefen erfolgreich ab
- * Java: jdk 8.0 * Einige Klassen, werden von der Coverage ausgeschlossen, da diese nicht getestet werden können:
 - Main.java (Ausführung)
 - ServletInitializer.java und CustomServletInitializer (Verbindungsaufbau)
 - ServletInitializer.java bzw. CustomServletInitializer (Verbindungsaufbau)
 - ChartBean.java (Erzeugen von Diagrammen)

Code Coverage Service

Element	Klasse%	Methode%	Zeile%
AtbaschCryptography	100% (1/1)	100% (5/5)	100% (8/8)
CaesarCryptography	100% (1/1)	100% (8/8)	100% (18/18)
CryptographyCenter	100% (1/1)	100% (4/4)	88% (22/25)
SubstitutionCryptography	100% (1/1)	100% (8/8)	100% (14/14)
UmlauteRuntimeException	100% (1/1)	0% (0/1)	50% (1/2)
CodeKnackerService	100% (1/1)	100% (1/1)	100% (10/10)
EmailService	100% (1/1)	100% (1/1)	86% (13/15)
HeaderFooterPage	100% (1/1)	100% (2/2)	100% (11/11)
PdfCreationService	100% (1/1)	85% (6/7)	85% (65/76)
TextConverterService	100% (1/1)	100% (13/13)	100% (48/48)

Code Coverage Controller

Element	Klasse%	Methode%	Line%
AtbaschController	100% (1/1)	100% (1/1)	100% (13/13)
CaesarController	100% (1/1)	100% (1/1)	100% (13/13)
CodeKnackerControllerTEST	100% (1/1)	85% (6/7)	88% (40/45)
EmailController	100% (1/1)	100% (1/1)	100% (4/4)
PdfController	100% (1/1)	100% (1/1)	66% (4/6)
SubstitutionController	100% (1/1)	100% (2/2)	100% (24/24)

Code Coverage Beans

Element	Klasse%	Methode%	Line%
CodeKnackerBean	100% (1/1)	92% (24/26)	96% (52/54)
EmailBean	100% (1/1)	100% (6/6)	100% (10/10)
InputHandleBean	100% (1/1)	100% (17/17)	96% (29/30)

Durchgeflaufen Test aufgelistet

ms	Bereich	Ergebnis
47	AtbaschCryptographyTest.encrypt	passed
5	AtbaschCryptographyTest.decryptString	passed
5	AtbaschCryptographyTest.encryptString	passed
4	AtbaschCryptographyTest.decrypt	passed
4	CaesarCryptographyTest.encrypt	passed
6	CaesarCryptographyTest.decryptString	passed
5	CaesarCryptographyTest.encryptString	passed
5	CaesarCryptographyTest.getAndSetKey	passed
4	CaesarCryptographyTest.decrypt	passed
4	SubstitutionCryptographyTest.encrypt	passed
4	SubstitutionCryptographyTest.decryptString	passed

ms	Bereich	Ergebnis
5	SubstitutionCryptographyTest.encryptString	passed
5	SubstitutionCryptographyTest.getAndSetCiphertextAlphabet	passed
3	SubstitutionCryptographyTest.decrypt	passed
4	UmlauteRuntimeExceptionTest.getErrormessage	passed
606	PdfCreationServiceTest.pdfCreationhardText	passed
273	PdfCreationServiceTest.pdfCreationSi	passed
8	TextConverterServiceTest.firstCharToArray	passed
3	TextConverterServiceTest.re	passed
4	TextConverterServiceTest.checkIfU	passed
0	TextConverterServiceTest.asciiArrayToString	passed
8	TextConverterServiceTest.stringToUpperCase	passed
4	TextConverterServiceTest.charArrayToString	passed
0	TextConverterServiceTest.integrateU	passed
9	TextConverterServiceTest.checkIfLetterInAlphabet	passed
0	TextConverterServiceTest.replaceU	passed
0	TextConverterServiceTest.stringToCharArray	passed
10	TextConverterServiceTest.deleteSpecialCharacters	passed
0	TextConverterServiceTest.stringToAsciiArray	passed
8	TextConverterServiceTest.counterLettersInString	passed
18	AtbaschControllerTest.handleEventEncrypt	passed
0	AtbaschControllerTest.handleEventPara	passed
8	AtbaschControllerTest.handleEventWithSpecialCharacters	passed
0	AtbaschControllerTest.handleEventWithWhitespace	passed
12	AtbaschControllerTest.handleEventDecrypt	passed
10	CaesarControllerTest.handleEventEncrypt	passed
0	CaesarControllerTest.handleEventPara	passed
10	CaesarControllerTest.handleEventWithSpecialCharacters	passed
0	CaesarControllerTest.handleEventWithWhitespace	passed
10	CaesarControllerTest.handleEventDecrypt	passed
10	CodeKnackerControllerTesting.TestControllerCaesarAndSi	passed
0	CodeKnackerControllerTesting.TestControllerCaesarAndHARD	passed

ms	Bereich	Ergebnis
10	CodeKnackerControllerTesting.TestSubstitutionAndReset	passed
0	CodeKnackerControllerTesting.TestControllerAtbasch	passed
1033	MailControllerTest.sendWrongE	passed
1600	MailControllerTest.sendE	passed
215	PdfControllerTest	passed
215	PdfControllerTest.createPDF	passed
5	SubstitutionControllerTest.handleEventEncrypt	passed
5	SubstitutionControllerTest.handleEventPara	passed
4	SubstitutionControllerTest.handleEventWithSpecialCharacters	passed
5	SubstitutionControllerTest.handleEventWithWhitespace	passed
6	SubstitutionControllerTest.rando	passed
6	SubstitutionControllerTest.handleEventDecrypt	passed

8.2.2 Akzeptanztest

Fragebogen

<p>Alter: _____ Jahre</p> <p>Geschlecht: <input type="radio"/> weiblich <input type="radio"/> männlich <input type="radio"/> divers</p> <p>Ort, Zeit: _____</p>	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td></td> <td>stimme voll zu</td> <td></td> <td>stimme eher nicht zu</td> </tr> <tr> <td></td> <td>stimme eher zu</td> <td></td> <td>stimme gar nicht zu</td> </tr> <tr> <td></td> <td>unentschieden</td> <td>X</td> <td>keine Aussage</td> </tr> </table>		stimme voll zu		stimme eher nicht zu		stimme eher zu		stimme gar nicht zu		unentschieden	X	keine Aussage																								
	stimme voll zu		stimme eher nicht zu																																		
	stimme eher zu		stimme gar nicht zu																																		
	unentschieden	X	keine Aussage																																		
<p>Gesamteindruck (Design, Bedienbarkeit)</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>Mir gefallen die Farben und das Layout.</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> <tr> <td>Die Bilder finde ich passend gewählt.</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> <tr> <td>Die Startseite (erste Seite beim Laden) ist übersichtlich.</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> <tr> <td>Ich finde schnell die Bereiche, die ich suche.</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> <tr> <td>Ich verstehe die Texte gut.</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> <tr> <td>Mir gefällt die gesamte Anwendung.</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> </table>		Mir gefallen die Farben und das Layout.					X	Die Bilder finde ich passend gewählt.					X	Die Startseite (erste Seite beim Laden) ist übersichtlich.					X	Ich finde schnell die Bereiche, die ich suche.					X	Ich verstehe die Texte gut.					X	Mir gefällt die gesamte Anwendung.					X
Mir gefallen die Farben und das Layout.					X																																
Die Bilder finde ich passend gewählt.					X																																
Die Startseite (erste Seite beim Laden) ist übersichtlich.					X																																
Ich finde schnell die Bereiche, die ich suche.					X																																
Ich verstehe die Texte gut.					X																																
Mir gefällt die gesamte Anwendung.					X																																
<p>Bereich Verschlüsseln</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>Ich konnte alle gewünschten Texte eingeben.</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> <tr> <td>Ich konnte alle gewünschten Texte verschlüsseln.</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> <tr> <td>Es traten beim Entschlüsseln keine Fehler auf.</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> </table>		Ich konnte alle gewünschten Texte eingeben.					X	Ich konnte alle gewünschten Texte verschlüsseln.					X	Es traten beim Entschlüsseln keine Fehler auf.					X																		
Ich konnte alle gewünschten Texte eingeben.					X																																
Ich konnte alle gewünschten Texte verschlüsseln.					X																																
Es traten beim Entschlüsseln keine Fehler auf.					X																																
<p>Bereich CodeKnacker</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>Die grafische Darstellung (Diagramm) unterstützt mich beim Finden des Schlüssels.</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> <tr> <td>Ich konnte schrittweise den passenden Schlüssel erarbeiten.</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> <tr> <td>Der CodeKnacker ist übersichtlich gestaltet, ich finde mich gut zurecht.</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> <tr> <td>Der CodeKnacker funktioniert ohne Probleme, es treten keine Fehler auf (visuell und funktionell).</td> <td></td> <td></td> <td></td> <td></td> <td>X</td> </tr> </table>		Die grafische Darstellung (Diagramm) unterstützt mich beim Finden des Schlüssels.					X	Ich konnte schrittweise den passenden Schlüssel erarbeiten.					X	Der CodeKnacker ist übersichtlich gestaltet, ich finde mich gut zurecht.					X	Der CodeKnacker funktioniert ohne Probleme, es treten keine Fehler auf (visuell und funktionell).					X												
Die grafische Darstellung (Diagramm) unterstützt mich beim Finden des Schlüssels.					X																																
Ich konnte schrittweise den passenden Schlüssel erarbeiten.					X																																
Der CodeKnacker ist übersichtlich gestaltet, ich finde mich gut zurecht.					X																																
Der CodeKnacker funktioniert ohne Probleme, es treten keine Fehler auf (visuell und funktionell).					X																																
<p>Persönliches Feedback (offene Fragen)</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th>Was hat dir gefallen?</th> <th>Was hat dir NICHT gefallen?</th> <th>Was hast du heute gelernt?</th> <th>Was wünschst du dir noch von der Anwendung?</th> </tr> <tr> <td colspan="4" style="height: 150px;"></td> </tr> </table>		Was hat dir gefallen?	Was hat dir NICHT gefallen?	Was hast du heute gelernt?	Was wünschst du dir noch von der Anwendung?																																
Was hat dir gefallen?	Was hat dir NICHT gefallen?	Was hast du heute gelernt?	Was wünschst du dir noch von der Anwendung?																																		

Auswertung des Fragebogens

		stimme voll zu	stimme eher zu	unentschieden	stimme eher nicht zu	stimme gar nicht zu	keine Aussage	ungültig
	Aussage							
Gesamteindruck	Mir gefallen die Farben und das Layout.	9	5	1	0	0	0	0
	Die Bilder finde ich passend gewählt.	10	5	0	0	0	0	0
	Die Startseite (erste Seite beim Laden) ist übersichtlich.	11	3	1	0	0	0	0
	Ich finde schnell die Bereiche, die ich suche.	10	5	0	0	0	0	0
	Ich verstehe die Texte gut.	12	2	1	0	0	0	0
Verschlüsselung	Mir gefällt die gesamte Anwendung.	9	3	2	0	0	0	1
	Ich konnte alle gewünschten Texte eingeben.	13	2	0	0	0	0	0
	Ich konnte alle gewünschten Texte verschlüsseln.	14	1	0	0	0	0	0
CodeKnacker	Es traten beim Entschlüsseln keine Fehler auf.	12	2	0	1	0	0	0
	Die grafische Darstellung (Diagramm) unterstützt mich beim Finden des Schlüssels.	5	3	3	2	1	0	0
	Ich konnte schrittweise den passenden Schlüssel erarbeiten.	11	3	1	0	0	0	0
	Der <i>CodeKnacker</i> ist übersichtlich gestaltet, ich finde mich gut zurecht.	11	3	1	0	0	0	0
	Der <i>CodeKnacker</i> funktioniert ohne Probleme, es treten keine Fehler auf (visuell und funktionell).	9	4	2	0	0	0	0

Abbildung 12: Gesamte Auswertung der Fragebogen

Antworten auf offenen Fragen

- Was hat dir gefallen?
 - Atbasch
 - Mir hat das Entschlüsseln der Code viel Spaß bereitet.
 - Benedikt war sehr freundlich
 - die Fragen
 - der Fragezettel
 - Das Programm ist im gesamten sehr übersichtlich!

- sehr tolle Spielerei, sehr interessant
 - Das Verschlüsseln der Nachrichten
 - Das Entcoden von den Nachrichten
 - Caesar -> Verschlüsselung & CodeKnacker
 - alles
 - Das es sehr übersichtlich ist und man gut zurecht kommt
 - Das wir die Buchstaben auch ändern können [SIC] wie bei der Frage von Caesar
- Was hat dir nicht gefallen?
 - Caesar
 - Das senden von Emails hat nicht geklappt.
 - Die Verzögerung beim Verschlüsseln
 - Nichts!!:)
 - nichts!
 - es hängt ein bisschen
 - Was hast du heute gelernt?
 - Codes zu entschlüsseln
 - Ich habe etwas über die wichtigsten Codes gelernt.
 - Wie, man Codes verwendet und verschlüsselt.
 - Wie man Codes verschlüsselt und entschlüsselt.
 - Praktische Seite gefunden [Mündliche Bemerkung bei Abgabe: „jetzt kann ich geheime Nachrichten meiner Freundin versenden“]
 - Das Whatsapp-Nachrichten verschlüsselt werden.
 - Das es Code Programme gibt zum entschlüsseln von Texten.
 - Wie Verschlüsselung aufgebaut ist.
 - Dass es viele Codes zum Verschlüsseln gibt.
 - Das Entcoden von Nachrichten, dass es schon in der Antike so etwas gab.
 - Wie ich mit meiner Freundin geheim schreiben kann.
 - Geheimschlüssel „knacken“
 - Wie man Nachrichten verschlüsselt und wozu man Nachrichten verschlüsseln muss.

- ein Code um mit meiner Freundin während des Unterricht zu reden oder etwas zu fragen. [Bem. Bezogen auf Aufgabe]
- Was wünscht du dir noch von der Anwendung
 - Nichts
 - Dass man seinen eigenen Code nicht nur durch Zufall generieren kann.
 - Schwere Wörter zum Knacken
 - Dass man [SIC] Email gut versenden kann
 - Mehr Codes, eigene Codes erstellen, Hintergrundinformationen zu den Codes (Geschichte)
 - Dass man dort keine IP-Adresse in die Suchzeile eingeben muss (z.B. www.codeKnacker.at)
 - auch nichts!
 - Bin damit schon zufrieden