DELTA – Střední škola informatiky a ekonomie s.r.o.

Ke Kamenci 151, PARDUBICE

MATURITNÍ PROJEKT

Portál historických kryptografických algoritmů

Jméno a příjmení: Jiří Ullrich

Třída: 4. A

Studijní obor: Informační technologie

Školní rok: 2022/2023

Zadání maturitního projektu z informatických předmětů

Jméno a příjmení: Jiří Ullrich
Pro školní rok: *2022/2023*
Třída: *4.A*
Obor: *Informační technologie 18-20-M/01*

Téma práce: Portál historických kryptografických algoritmů
Vedoucí práce: Mgr. Josef Horálek, Ph.D.

Způsob zpracování, cíle práce, pokyny k obsahu a rozsahu práce:

Cílem maturitního projektu je vytvořit portál, jehož cílem je podpořit výuku úvodu do kryptografie. Autor vytvoří komplexní webové řešení pro podporu výuky ALG, které bude obsahovat vysvětlení vybraných historických kryptografických algoritmů a jejich ukázkovou realizaci. V teoretické části práce autor představí teoretické principy vybraných algoritmů a technologie pro web portálů. V praktické části autor vytvoří web portál dle uvedených kritérií za dodržení aktuálních standardů vývoje.

Stručný časový harmonogram (s daty a konkretizovanými úkoly):

09 – 10 Analýza vhodných webových technologií a návrh modelu webu
11 – 12 Zpracování historických kryptografických algoritmů
12 – 02 Implementace vybraných problémů
02 - 03 Dokončení webového řešení
03 Finalizace textového znění maturitního projektu

Prohlašuji, že jsem maturitní projekt vypracoval samostatně, výhradně s použitím uvedené literatury.

V Pardubicích 31.3.2023

Poděkování

Chtěl bych hlavně poděkovat Mgr. Josefu Horálkovi, Ph.D. za inspiraci, vedení tohoto projektu a konzultaci problémů.

Dále bych chtěl poděkovat Ing. Monice Borkovcové, Ph.D. za nápady, jak projekt vylepšit po funkční stránce.

Anotace

Maturitní projekt předkládá řešení webového portálu sloužícího jako nástroj pro podporu výuky algoritmizace vybraných kryptografických šifer. Konkrétně Caesarovy, Afinní, Transpoziční, Homofonní a Vigenèrovi šifry.

Anotation

This project shows the development of a web application serving as a tool for teaching the algorithmization of chosen cryptographic ciphers. Specifically, Caesar, Affine, Transposition, Homophonic, and Vigenère ciphers.

Klíčová slova

JavaScript, SvelteKit, Svelte, Tailwind, Substituční šifra, polyalfabetická substituční šifra, Vercel

Key words

JavaScript, SvelteKit, Svelte, Tailwind, Substitution cipher, polyalphabetic substitution cipher, Vercel

Úvod

Cílem maturitního projektu je vytvořit webový portál na objasnění a implementaci vybraných šifer. Konkrétně Caesarovy, Afinní, Transpoziční, Homofonní a Vigenèrovi šifry. Pro naplnění cílů, je zvolen postup, kdy jsou nejprve popsány principy jednotlivých šifer, zmíněn jejich historický kontext a příklady užití. Součástí každé stránky bude aplikace, kde si uživatel může funkčnost šifry vyzkoušet, zdrojový kód šifry a krokovací aplikace, ve kterém se uživatel může podívat, jak kód řádek po řádku prochází.

Obsah

1 Technologie

1.1 JavaScript

JavaScript je vysokoúrovňový dynamicky typovaný jazyk. Je široce využívaný pro tvorbu interaktivních webových stránek. Využívá se jak na frontendu, tak i na backendu. JavaScript je interpretovaný jazyk. Interpretované jazyky nekompilují celý kód najednou, ale řádek po řádku. [1]

1.2 Sveltekit

Sveltekit je framework postavený na frameworku Svelte. Svelte je frontend framework pro tvorbu uživatelského rozhraní. Je navržený pro efektivní psaní kódu, minimalizuje množství kódu, který je zapotřebí spustit v prohlížeči. Sveltekit poskytuje dodatečné nástroje pro ulehčení nasazení webových aplikací napsaných za pomocí Svelte. [2]

1.3 Tailwind

Tailwind je CSS framework, který urychluje stylování webových aplikací. Obsahuje sadu předefinovaných CSS tříd. Ty lze jednoduše aplikovat na HTML elementy k dosažení požadovaného stylování. [3]

1.4 Prism JS

Prism JS je JavaSript knihovna, která automaticky styluje ukázky kódu pro zobrazení na webu. Podporuje většinu populárních programovacích jazyků jako je HTML, CSS, JavaScript, PHP, Python a další.[4]

1.5 Vercel

Vercel je hostingová platforma s automatickým škálováním a rychlým nasazením. Podporuje řadu populárních moderních nástrojů pro tvorbu webu jako je například React, Vue.js, Next.js, Sveltekit a mnoho dalších. K nasazení aplikace stačí propojit vercel s githubem a vybrat repositář ze, kterého chcete čerpat. Vercel automaticky aplikaci aktualizuje po každém commitu do hlavní větve v repositáři. [5]

2 Teorie šifer

2.1 Caesarova šifra

2.1.1 Princip

Caesarova šifra je jednou z nejjednodušších šifer. Patří do kategorie substitučních šifer. Jedné se o šifry, u kterých při šifrování dochází k záměně jedné skupiny písmen za jinou. Šifra využívá pozice písmen v abecedě. Při šifrování dochází k posunu písmen v abecedě o počet písmen rovnu klíči. Takže pokud bude klíč 2, tak místo písmene A zapíšeme C. Pro příklad, kdybychom chtěli zašifrovat slova ahoj a klíč by byl třeba 4, tak by slovo po zakódování bylo elsn. [6]

2.1.2 Historie

Caesarova šifra je pojmenovaná podle Július Caesar, který ji používal během Galských válek. Konktrétně šifru používal s posunutím o 3. Neví se, jak moc v té době byla šifra efektivní, ale je velice pravděpodobné, že byla přijatelně bezpečná. Jenom pár Caesarových protivníků bylo gramotných natož aby se ještě zabývali kryptoanalýzou.

2.1.3 Funkcionalita kódu

Kód prochází zprávu písmeno po písmenu a každé jednotlivě zašifruje. Pokud je znak mezera, tak je šifrovací proces přeskočen a do zašifrovaní zprávy se uloží mezera. Proces šifrování písmen je následovný. Nejdříve zjistíme hodnotu pozice písmena v abecedě. Potom se k zjištěné hodnotě přičte klíč. Pokud je dosažená hodnota větší než 26 (počet písmen v anglické abecedě), tak se od hodnoty odečte 26. Následně kód vyhledá písmeno, které se nachází na pozici v abecedě odpovídající získané hodnotě a uloží ji do proměnné.[6]

```
let number = 0;
let helper = 0;
let string = "ahoj";
let vysledek = "";
let abcd = "abcdefghijklmnopqrstuvwxyz";
let ceaserCipher = (str, key) => {
  let decipher = "";
  str = str.toLowerCase();
  for (let i = 0; i < str.length; i++) {
    if (str[i] == " ") {
      decipher += " ";
    } else {
      helper = abcd.indexOf(str[i]);
      console.log(helper);
      helper += parseInt(key);
      console.log(helper);
      if (helper > 25) {
        helper = helper - 26;
      }
      decipher = decipher + abcd.charAt(helper);
    }
  }
  vysledek = decipher;
  return decipher;
};
```

Obrázek 1kód Caesarova šifra

2.1.4 Příklad šifry

Pro jednoduché znázornění funkčnosti Caesarovi šifry využijeme tabulku, v jejíž řádcích jsou vypsána všechna písmena abecedy. Do prvního řádku zapíšeme písmena standartně, jak jdou po sobě. Druhou řadu však posuneme o určitý počet řádů. V mém případě jsem písmena v druhém řádku posunul o jednu pozici. Takže a bude pod b, b pod c a tak dále až na konec abecedy. Písmeno, z které se zdánlivě do tabulky už nevejde, přesuneme zpátky na začátek pod písmeno a.

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	S	t	u	v	w	x	y	z
z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	R	s	t	u	v	w	x	y

Tabulka 1 ceasarova šifra

Jakmile máme zhotovenou tabulku, samotné šifrování je již velice jednoduché. Vybereme si zprávu, kterou chceme zašifrovat, a každé písmeno přepíšeme dle tabulky. Takže slovo „ahoj“ by bylo „zgni“.

2.2 Afinní šifra

2.2.1 Princip

Afinní šifra patří do rodiny substitučních šifer stejně jako Caesarova šifra. Na rozdíl od ní nabízí více možností transformace, a tak se stává mnohem náročnější na prolomení. Využívá matematického vzorce k převedení každého písmena do zašifrovaného textu. Vzorec vypadá následovně $x = (a \times T + b) mod 26$ kde X je pozice šifrovaného písmena po zašifrování, a první klíč, b druhý klíč, T pozice šifrovaného písmena v abecedě. Mod je zbytek po dělení, v případě této šifry dělíme 26. K zašifrování si nejdříve vybereme dva klíče (klíče musí být celá čísla). Následně na každé písmeno zprávy použijeme vzorec. Například řekněme, že vybereme čísla a = 4 a b = 6 jako šifrovací klíče. Ještě potřebujeme zprávu k zašifrování. Pro ukázku můžeme použít třeba písmeno A jako zprávu. Vzorec by v tomto případě vypadal následovně $x = (4 \times 0 + 6) mod 26$. Výsledkem je 6, což odpovídá pozici v abecedě písmenu G. [7]

2.2.2 Historie

Afinní šifra patří k jedné z nejstarších šifer a předpokládá se, že byla využívána ve starověkém Římě a Řecku. První zmínka o afinní šifře pochází z 9. století před naším

letopočtem. Arabský matematik Al-Kindi popsal jednoduchou substituční šifru využívající afinní šifrování. V období Renesance využívala afinní šifru evropská šlechta k ochraně diplomatické a armádní komunikace. V dnešní době je afinní šifra už jen historickou a matematickou zajímavostí. K bezpečnostním účelům se již nevyužívá z důvodů jejích slabin.

2.2.3 Funkcionalita kódu

Stejně jako u Caesarovy šifry kód prochází zprávu písmeno po písmenu a jednotlivě je šifruje. Cyklus využívá délku zprávy jako počet průchodů. Pokud narazí na mezeru ve zprávě, tak nic nešifruje a jenom jí zapíše do výsledného textu. V případě, že jde o písmeno, kód využije vzorce, čímž získá novou pozici písmena v abecedě. Následně najde písmeno odpovídající pozici a uloží ho do výsledného textu.

```
let b = 0;
let a = 0;
let helper = 0;
let string = "ahoj";
let vysledek = "ahoj";
let abcd = "abcdefghijklmnopqrstuvwxyz";

let afinniCipher = (str, a, b) => {
  let decipher = "";

  for (let i = 0; i < str.length; i++) {
    if (str[i] == " ") {
      decipher += " ";
    } else {
      helper = (parseInt(a) * abcd.indexOf(str[i]) + parseInt(b))
% 26;
      decipher += abcd.charAt(helper);
    }
  }
  vysledek = decipher;
  console.log(vysledek);
  return decipher;
};
```

Obrázek 2kód Afinní šifra

2.2.4 Příklad šifry

Pro jednoduchou ukázku zašifrujeme slovo „Ahoj“. Jako klíče A = 5 a B = 8. Dalším krokem je dosazení každého písmena zprávy do vzorce afinní šifry $x = (a \times T + b) mod 26$. Prvním písmenem je „A“ a má hodnotu 1 dle tabulky níže, takže za T dosadíme 1. Vzorec bude v tomto případě vypadat následovně $x = (5 \times 0 + 8) mod 26$. Výslednou hodnotou je 8. Hodnotě osm v abecedě odpovídá písmeno „i“. Zapíšeme tedy i a pokračujeme nadále v šifrování. Tento proces následně zopakuje pro každé písmeno zprávy. Výsledek bude „irab“.

Tabulka 2 afinní šifra

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

2.3 Transpoziční šifra

2.3.1 Princip

Transpoziční šifrování je druh šifrování při, kterém dochází k přeskupení písmen původní zprávy. Pomocí předem domluveného způsobu. Aby šifru mohl příjemce rozšifrovat, musí vědět, jakým způsobem byla písmena původní zprávy přeskupena. Výsledná zpráva je permutací původní, což ji dělá náročnou k prolomení, pokud nevíte metodu transpozice, která byla využita. V transpoziční šifře jsou písmena zapsána do tabulky o předem dané velikosti sloupců. Následně jsou písmena v tabulce přeházena pomocí předem daného vzoru. Transpoziční šifry mohou mít ale mnoho podob.[8]

2.3.2 Historie

Jeden z nejstarších příkladů transpoziční šifry je skytalé. Na dřevěný válec předem domluveného průměru byl namotán pásek pergamenu či papyru. Po namotání byla na pergamen napsána zpráva. Následně byl pergamen rozmotán a zůstala jen písmena, která samostatně nedávala smysl a původní zpráva šla přečíst jen po opětovném namotání na válec o stejném průměru.

Obrázek 3 Skytalé

Skytalé používali starověký Řekové a Sparťani. Nepřímý důkaz o této metodě šifrování pochází z 7. století před naším letopočtem. Zmínil se o ní řecký básník Archilochus, ale první přímý důkaz o tomto šifrovacím nástroji byl zaznamenán ve spisech básníka Apollóniose z Rhodu.

Ve středověku bylo používáno spoustu druhů transpozičních šifer k válečným a diplomatickým komunikacím. Například Polyalfabetická šifra byla využívaná anglickou královnou Marií I.

2.3.3 Funkcionalita kódu

První část transpoziční šifry je tvorba tabulky, do které se zpráva rozepíše. V kódu se vytvoří prázdné pole. To se následně začne v cyklu plnit původní zprávou řádek po řádku. Počet písmen v jednom řádku je určen zadaným klíčem. K zjištění, kolikrát má cyklus projít, vydělíme délku původní zprávy klíčem(počtem písmen v řádku). Dalším krokem je vytvořit výslednou tabulku, do které se písmena přepíšou. Sloupce z původní tabulky budou řádky v nové. Takže první sloupec z původní tabulky bude první řádek v nové.

```
let sifra = "";
let limit = 0;
let key = 3;
let tabulka = new Array();
let vypis = "";
let decipher = new Array();

let traspozicniSifra = () => {
  if (key < 2) {
    return;
  }

  limit = 0;
  tabulka = new Array();
  vypis = "";
  decipher = new Array();
  sifra = sifra.replaceAll(" ", "");
```

Obrázek 4 kód transpoziční šifra

```
    for (let i = 0; i < Math.ceil(sifra.length / key); i++) {
      if (limit < sifra.length) {
        tabulka.push(sifra[limit]);
        limit += 1;
      }
      for (let x = 0; x < key - 1; x++) {
        if (limit < sifra.length) {
          tabulka[i] += sifra[limit];
          limit += 1;
        }
      }
    }

    for (let i = 0; i < key; i++) {
      decipher.push("");
    }
    for (let index = 0; index < key; index++) {
      for (let i = 0; i < Math.ceil(sifra.length / key); i++)
  {
        if (typeof tabulka[i][index] !== "undefined") {
          decipher[index] += tabulka[i][index];
        }
      }
    }
    for (let index = 0; index < decipher.length; index++) {
      vypis += decipher[index];
    }
  },
```

Obrázek 5 kód transpoziční šifra

2.3.4 Příklad šifry

K použití transpoziční šifry budeme potřebovat klíč, který bude udávat počet sloupců původní tabulky. Na příklad použijme klíč 4 k zašifrování zprávy „ahoj jak se máš“. Tajnou zprávu zapíšeme do tabulky o 4 sloupcích řádek po řádku. Následně první tabulku přepíšeme tak, že první řádek původní tabulky bude první sloupec nové tabulky, druhý řádek původní tabulky bude druhým sloupcem nové tabulky. Tímto způsobem přepíšeme celou tabulku. Zašifrovanou zprávu dostaneme přečtením druhé tabulky řádek po řádku „ajehamokájsš“.

a	h	o	j
j	a	k	s
e	m	á	š

Tabulka 3 transpoziční šifra

a	j	e
h	a	m
o	k	á
j	s	š

Tabulka 4 transpoziční šifra

2.4 Homofonní šifra

2.4.1 Princip

Homofonní šifra patří mezi pokročilejší substituční šifry. Využívá nahrazování jednotlivých písmen abecedy za několik různých symbolů nebo kombinací symbolů. Tyto nahrazování jsou obvykle vytvořeny tak, aby každé písmeno mělo mnoho možných kombinací, a tím pádem je těžké zjistit, jaké písmeno nebo kombinace písmen daný symbol představuje. Homofonní šifry mohou být vytvářeny různými způsoby. Některé šifry nahrazují jednotlivá písmena za jediný symbol nebo kombinaci symbolů, zatímco jiné šifry používají různé symboly a kombinace pro každou jednotlivou pozici v textu. Tyto různé přístupy mají rozdílné úrovně zabezpečení a složitosti. Pro vytvoření homofonní šifry je obvykle nutné mít klíč, který umožní převést šifrovaný text zpět do původního textu. Tento klíč obsahuje

informace o tom, které symboly odpovídají jednotlivým písmenům abecedy. Bez klíče je dešifrování velmi obtížné a v některých případech nemožné.[6]

2.4.2 Historie

Přesný původ homofonní šifry není zcela jasný. Několik osobností a skupin využívalo variantu této šifry. První náznaky homofonní šifry můžeme najít již v 9. století, když arabští matematici vyvinuly šifru, která nahrazuje každé písmeno několika různými znaky. Tato metoda byla ve středověku využívána evropskými kryptografy. Mimo jiné byla homofonní šifra využívána například za první světové války. Obě strany využívaly šifry, které kombinovaly substituci a nahrazování písmen několika znaky.

2.4.3 Funkcionalita kódu

Kód vygeneruje dvoj rozměrné pole „homophones“, které se využije jako klíč k šifrování. V každém poli tohoto pole budou 3 čísla od 1 do 99. Každé číslo bude v celém dvojrozměrném poli použito jen jednou. V kódu se tento proces vykonává následovně. Pole „cisla“ obsahuje čísla od 1 do 99. Z této tabulky se vybere náhodné číslo a je zapsáno do dvojrozměrného pole „homophones“. Vybrané číslo se z pole „cisla“ odebere. Tento proces se opakuje, dokud není pole „homophones“ naplněno 26 poli, kde každé pole obsahuje 3 čísla. Každé pole v poli „homophones“ představuje jedno písmeno abecedy. Teprve po kompletním vygenerování klíče dochází k šifrování samotné zprávy. Pro každé písmeno zprávy kód náhodně vybere jedno ze tří čísel odpovídající pozici písmena v abecedě z tabulky „homophones“ a zapíše ho do proměnné „encrypted“.

```
let homophones = new Array();
let limit = 99;
let rng;
let abc = "abcdefghijklmnopqrstuvwxyz";
let message = "Ahoj";
let encrypted = "";
for (let a = 0; a < 26; a++) {
  homophones.push([]);
  for (let b = 0; b < 3; b++) {
    rng = Math.floor(Math.random() * limit);
    homophones[a].push(cisla[rng]);
    cisla.splice(rng, 1);
    limit = limit - 1;
  }
}

function encription() {
  encrypted = "";
  for (let i = 0; i < message.length; i++) {
    const letter = message[i].toLowerCase();
    if (letter !== " ") {
      const homophoneIndex = Math.floor(
        Math.random() *
homophones[abc.indexOf(letter)].length
      );
      encrypted +=
homophones[abc.indexOf(letter)][homophoneIndex];
    } else {
      encrypted += message[i];
    }
  }
}
```

Obrázek 6 kód homofonní šifra

2.4.4 Příklad šifry

K homofonní šifrování využijeme tabulku, ve které každé písmeno abecedy má několik čísel jakými lze reprezentovat. K příkladu budu vyžívat tabulky, kterou můžete najít níže. Při šifrování jednotlivých písmen vybereme jednu z hodnot písmena. Šifrování slova „ahoj" bude vypadat takto. Pro písmeno „a" vybereme jednu ze 3 možností například 42. Stejný proces využijeme pro ostatní písmena „h" 28, „o" 52, a pro písmeno „j" 04. Konečná zpráva bude 42285204.

A	5	42	47
B	62	46	93
C	72	71	58
D	29	86	39
E	33	78	54
F	24	1	77
G	7	8	16
H	28	38	92
I	98	53	70
J	15	4	40
K	11	26	23
L	91	89	12
M	81	35	34
N	6	63	31
O	52	43	83
P	65	67	76
Q	9	45	99
R	2	25	82
S	74	94	90
T	49	14	85
U	37	3	80
V	69	56	79
W	57	48	50
X	61	19	55
Y	44	66	17
Z	36	96	97

Obrázek 7 homofonní šifra klíč

2.5 Vigenèrova šifra

2.5.1 Princip

Vigenérova šifra je polyalfabetická substituční šifra, která k šifrování a dešifrování využívá tabula recta. Tabula recta obsahuje 26 pod sebou napsaných abeced. Na každém řádku začíná abeceda jiným písmenem. Na prvním řádku je abeceda napsaná standartně od a do z, druhý řádek je posunutý, písmena jsou zapsána od b do z následované písmenem a. Celý proces se opakuje pro každé písmeno. Klíčem pro Vigenèrovu šifru může být libovolné slovo. Při šifrování písmen se hledá ve sloupci každého písmena zprávy odpovídající písmeno ve sloupcích písmen klíče. Pro prvním písmeno zprávy se odpovídající písmeno hledá v řádku prvního písmena klíče. Tento proces se opakuje pro každé písmeno zprávy. [6]

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

Obrázek 8 Tabula Recta

2.5.2 Historie

Vigenèrova šifra se poprvé objevila roku 1585 v knize Traicté des Chiffres, kterou napsal francouzský diplomat Blaise de Vigenère. Přestože byla šifra popsána již v 16. století, nebyla moc známá. Až skoro o 200 let později ji prolomili matematik Charles Babbage a německý důstojník Friedrich Kasiski.

2.5.3 Funkcionalita kódu

Nejprve kód vytvoří dvojrozměrné pole, do kterého se uloží tabula recta. Pole bude obsahovat počet polí odpovídající počtu písmen v abecedě tedy 26. V druhém cyklu kódu dochází k samotnému šifrování. Písmena zprávy se postupně zašifrují. K šifrování se využívá pozice písmen z klíče a pozice šifrovaného písmena. Zašifrované písmeno se uloží do proměnné „zakod“.

```
let abc = "abcdefghijklmnopqrstuvwxyz";
let vysledek = new Array();
let kod = "kod";
let pocitadlo = 0;
let zprava = "ahooj jak se mas";
let zakod = "";
let pozice = 0;
let zacatek = 0;
for (let i = 0; i < 26; i++) {
  vysledek.push([]);
  pozice = zacatek;
  for (let b = 0; b < 26; b++) {
    vysledek[i] += abc[pozice];
    pozice += 1;
    if (pozice > 25) {
      pozice = 0;
    }
  }
  zacatek += 1;
}
pocitadlo = 0;
zakod = "";
for (let p = 0; p < zprava.length; p++) {
  if (zprava[p]!== " ") {
    zakod +=
vysledek[abc.indexOf(kod[pocitadlo])][abc.indexOf(zprava[p])
];
```

Obrázek 9 kód Vigenèrova šifra

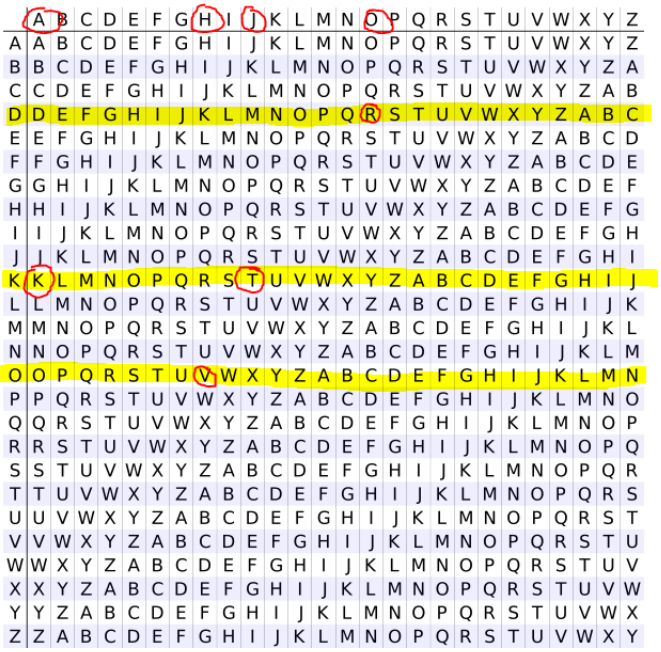
```
        pocitadlo += 1;


          if (pocitadlo >= kod.length) {
             //podmínka zajišťující
            pocitadlo = 0;
          }
        } else {
          zakod += " ";
        }
      }
```

Obrázek 10 kód Vigenèrova šifra

2.5.4 Příklad šifry

Pro šifrování je využívána tabula recta. Jako klíč použijeme slovo „kod“ budeme šifrovat zprávu „ahoj“. Pro každé písmeno budeme hledat reprezentaci tohoto písmena na řádku písmena z klíče. Pro první písmeno zprávy použijeme první písmeno klíče, pro druhé písmeno zprávy druhé písmeno klíče a tak dále. Zpráva má ale více písmen než klíč, jakmile vyčerpáme písmena klíče, tak použijeme znova první písmeno klíče. Pokud je zpráva delší než „ahoj“, tak se celý proces opakuje, dokud není každé písmeno zašifrované. Výsledek šifrování je kvrt.



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

Obrázek 11 příklad tabula recta

3 Funkce webu

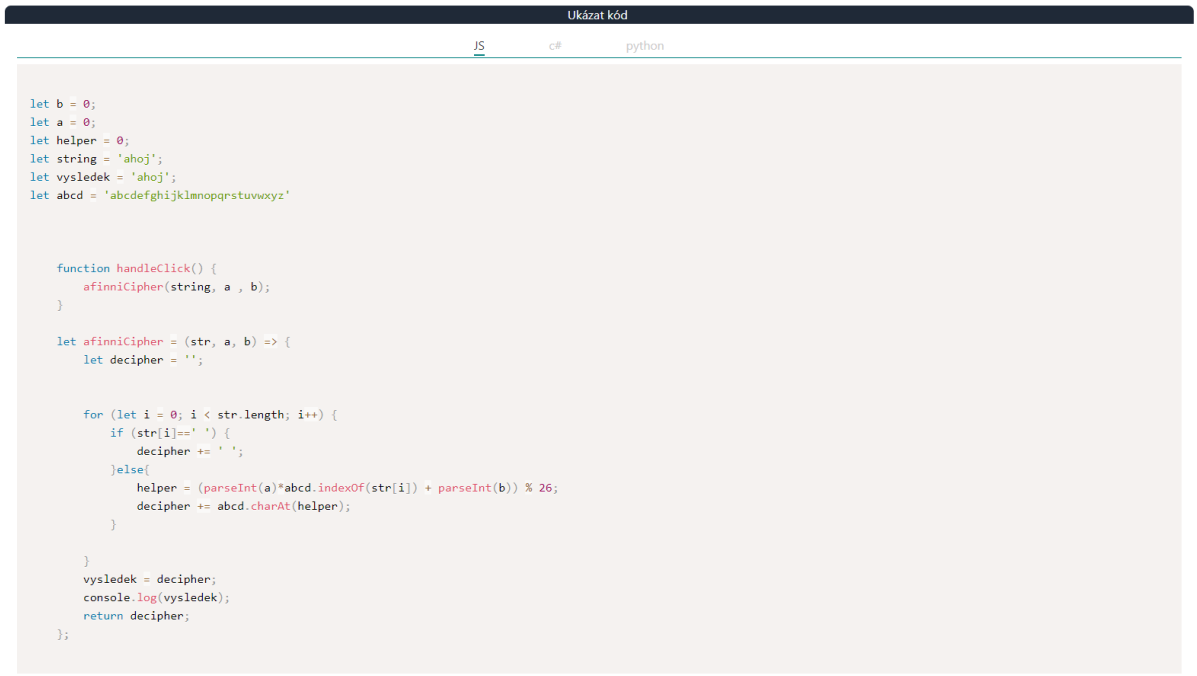
3.1 Šifrování

Stránka každé šifry nabízí možnost vyzkoušení dané šifry. Pro zadání potřebných informací slouží input boxy. Například afinní šifra k šifrování potřebuje dva klíče a samotný text k zašifrování. Po kliknutí na tlačítko šifrovat se zobrazí zašifrovaná zpráva.

Obrázek 12 ukázka šifrování

3.2 Zdrojové kódy

Stránka každé šifry dále nabízí možnost náhledu do zdrojového kódu šifer. Po kliknutí na tlačítko ukázat více, se rozvine blok s kódem. Kódy jsou stylované za použití prism js. Kódy je možno zobrazit ve více programovacích jazycích.



Obrázek 13ukázka zobrazení zdrojového kódu

3.3 Krokovací aplikace

Po úspěšném zašifrování dojde k zobrazení tlačítka pro krokování. Tato funkce umožňuje procházení kódu krok po kroku klikáním na tlačítko pokračovat. Při šifrování dochází k zaznamenání průchodu kódu. Krokovací aplikace prochází záznamem postupu kódu a umožňuje bližší náhled na šifrovací proces. Při průchodu aplikací dochází k postupnému zobrazení zašifrované zprávy odpovídající ukládání zprávy v aplikaci.

Obrázek 14 ukázka krokovací aplikace

4 Závěr

V rámci projektu byl úspěšně vytvořen portál pro podporu výuky historických kryptografických algoritmů. Pro každou z vybraných šifer byla vytvořená stránka s jejím popisem, aplikací umožňující vyzkoušet si funkcionalitu šifry, stylovaný zdrojový kód a krokovací aplikaci ve, si uživatel může podívat jakým způsobem kód probíhá. V první kapitole jsou vysvětleny použité technologie. Druhá kapitula se věnuje teorii použitých šifer. Třetí kapitola popisuje funkcionalitu finální webové aplikace.

5 Zdroje

[1] What is JavaScript. *Https://developer.mozilla.org/* [online]. [cit. 2023-03-31]. Dostupné z: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript

[2] *Svelte* [online]. [cit. 2023-03-31]. Dostupné z: https://svelte.dev/

[3] *Tailwindcss* [online]. [cit. 2023-03-31]. Dostupné z: https://tailwindcss.com/

[4] *Prismjs* [online]. [cit. 2023-03-31]. Dostupné z: https://prismjs.com/

[5] Dokumentace. *Vercel* [online]. [cit. 2023-03-31]. Dostupné z: https://vercel.com/docs

[6] PIPER, F. C. a Sean MURPHY. *Cryptography: a very short introduction*. New York: Oxford University Press, 2002. ISBN 0192803158.

[7] Afinní šifra. *Algoritmy.net* [online]. [cit. 2023-03-31]. Dostupné z: https://www.algoritmy.net/article/49/Afinni-sifra

[8] Transpoziční šifra. *Youtube* [online]. [cit. 2023-03-31]. Dostupné z: https://www.youtube.com/watch?v=sHsnH1u03e4&t=40s

6 Přílohy