Šifrovací metody v prostředí webMathematica

Encryption methods in environment of webMathematica

Vítězslav Jaroš

Bakalářská práce
2007

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav aplikované informatiky
akademický rok: 2006/2007

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Vítězslav JAROŠ**
Studijní program: **B 3902 Inženýrská informatika**
Studijní obor: **Informační technologie**

Téma práce: **Šifrovací metody v prostředí WebMathematica**

Zásady pro vypracování:

Diplomová práce se zabývá tvorbou vybraných šifrovacích metod v prostředí WebMathematica. Předpokládá se základní znalost prostředí Mathematica, šifrovacích metod a schopnost tvorby WWW stránek.

1. Vypracujte literární rešerši na téma šifrování.
2. Navrhněte způsob realizace těchto metod s využitím prostředí WebMathematica.
3. Demonstrujte na vybraných příkladech.
4. Vypracujte závěr.

Rozsah práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

Grošek O., Porubský Š.: Šifrovanie. Algoritmy, metódy a prax. Praha, Grada 1992
Janeček J.: Odhalená tajemství šifrovacích klíčů minulosti. Praha, Naše vojsko, 1994
Prokopová Z.: Základy informatiky. Vydavatelství FT, Zlín, 1998
Zelinka I.: Aplikovaná informatika. VUT Zlín, 1999

Vedoucí bakalářské práce: **doc. Ing. Ivan Zelinka, Ph.D.**
Ústav aplikované informatiky

Datum zadání bakalářské práce: **13. února 2007**

Termín odevzdání bakalářské práce: **24. května 2007**

Ve Zlíně dne 13. února 2007

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky

prof. Ing. Vladimír Vašek, CSc.
děkan

doc. Ing. Ivan Zelinka, Ph.D.
ředitel ústavu

ABSTRAKT

Abstrakt česky

Tato práce se zabývá klasickými metodami šifrování, mezi něž mimo jiné patří Viegenérova šifra, Cardanova mřížka, Cézarova šifra a další. Vybrané metody jsou zpracovány v programu Mathematica a poté implementovány do prostředí webMathematica, což je zjednodušeně řečeno Mathematica na Internetu. Čtenář této práce by si měl po přečtení osvojit teorii každé uvedené šifry a poté si ji může vyzkoušet v praxi ve svém prohlížeči a také se dozvědět základy práce v prostředí WebMathematica.

Klíčová slova: Mathematica, webMathematica, HTML

ABSTRACT

Abstract in English

This work deal with classical methods encryption, among which among others belongs to Viegener cipher, Cardan grating, Cesar cipher and next. Choice method are processed in programme Mathematica and after it implemented to the environment webMathematica, which is simply told by Mathematica on internet. Reader those work would had after perused develop theory every mentioned cipher and after it her is able to try out practically in his browser as well as learn bases work in environment webMathematica.

Keywords: Mathematica, webMathematica, HTML

Poděkování, motto

Poděkování patří především mému vedoucímu bakalářské práce doc. Ing. Ivanu Zelinkovi, Ph.D. za rady a připomínky, které mi poskytoval během vypracovávání mojí bakalářské práce.

Prohlašuji, že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků, je-li to uvolněno na základě licenční smlouvy, budu uveden jako spoluautor.

Ve Zlíně …….

Podpis diplomanta

OBSAH

ÚVOD **8**

I TEORETICKÁ ČÁST **9**

1 KLASICKÉ METODY ŠIFROVÁNÍ **10**

1.1 TRANSPOZIČNÍ ŠIFRY 11

1.1.1 Viegenérova šifra 11
1.1.2 Jednoduchá transpozice 12
1.1.3 Dvojitá transpozice 13
1.1.4 Cardanova mřížka 14

1.2 SUBSTITUČNÍ ŠIFRY 15

1.2.1 Cézarova šifra 15
1.2.2 Jednoduchá substituce v tabulce 16
1.2.3 Autokláv 16
1.2.4 Jednoduchá záměna 17
1.2.4.1 Srovnaná abeceda 17
1.2.4.2 Reciproká abeceda 18
1.2.4.3 Rozházená abeceda 18
1.2.5 Tabulka 4x7 19
1.2.6 Tabulka 5x10 19
1.2.7 Playfair 20

2 HISTOGRAM **22**

3 WEBMATHEMATICA **23**

3.1 ÚVOD DO APLIKACE WEBMATHEMATICA 23

II PRAKTICKÁ ČÁST **24**

4 WEBMATHEMATICA **25**

4.1 PROGRAMOVANÍ STRÁNEK WEBMATHEMATICA 25

4.1.1 Základní obsah dokumentu webMathematica 25
4.1.2 Získávání dat 26
4.1.3 Zobrazení výsledku 27

4.2 WWW STRÁNKY 29

4.2.1 Grafické zobrazení 29
4.2.2 Metody programování 30
4.2.3 Struktura www stránek 30
4.2.4 Ovládání stránek 31
4.2.5 Ukázka zdrojového textu šifry a její popis 32
4.2.5.1 Zdrojový text pro aplikaci webMathematica 32
4.2.5.2 Zdrojový text pro aplikaci Mathematica 37

ZÁVĚR **40**

ZÁVĚR V ANGLIČTINĚ **41**

SEZNAM POUŽITÉ LITERATURY **42**

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK **43**

SEZNAM OBRÁZKŮ **44**
SEZNAM TABULEK **45**
SEZNAM PŘÍLOH **46**

ÚVOD

Šifrovací metody jsou předmětem výuky kurzu Aplikovaná informatika. Jelikož se jedná o velmi obsáhlé téma, které by zabralo několik semestrů, tak je tento kurz zaměřený na klasické metody šifrování. Tyto metody jsou počátkem současných šifrovacích metod. Teorie není složitá, což přispívá ke snadnému pochopení vyučované látky.

Součástí tohoto kurzu jsou i cvičení, ve kterých jsme měli možnost si odpřednášené metody šifrovaní naprogramovat v programu Mathematica a tím si prakticky ukázat funkci jednotlivých metod. Někde tady asi vznikla otázka: „Jak mám ukázat svoji skvěle naprogramovanou šifru kamarádovi, který nemá program Mathematica ve svém počítači?“ Odpovědí na tuto otázku je WebMathematica.

WebMathematica je nástroj, který přenese vaše projekty z okna programu Mathematica do internetového prohlížeče kohokoliv na celém světě.

Pokud se tedy chcete dozvědět o několika klasických metodách šifrování, způsoby jejich naprogramovaní v programu Mathematica a implementaci těchto zdrojových kódů do prostředí WebMathematica čtěte dále.

I. TEORETICKÁ ČÁST

1 KLASICKÉ METODY ŠIFROVÁNÍ

Už od starověku se lidé snaží předávat zprávy tak, aby je mohl číst pouze adresát a nikdo jiný. Je-li zpráva napsána ručně a doručována otrokem (jak tomu bylo ve starém Řecku nebo Římě) nebo poštou dnes, tak vždy existuje nebezpečí, že se dostane do ruky někomu, komu není určena. Otrok může být zajat, pošťák se může zmýlit a doručit zprávu na nesprávnou adresu. Je-li zpráva napsána srozumitelně, přirozeným jazykem bez jakékoliv snahy skrýt její obsah, tak jí může porozumět každý, komu se dostane do ruky, pokud zná jazyk, kterým je napsána. V novější době můžeme zprávy předávat telegrafem, radiovými vlnami, telefonem, faxem nebo e-mailem, ale nebezpečí, že budou odposlechnuty, je stále přítomné. Ve skutečnosti se od té doby nesmírně zvýšilo. Tak například radiové vysílání může slyšet každý, kdo je v dosahu vysílače a má přijímač naladěný na správnou frekvenci. Zpráva e-mailem může být doručena na spoustu nezamýšlených adres v důsledku překlepu nebo viru číhajícího v počítači. Jakkoliv pesimisticky to může vypadat, je dobrým pravidlem předpokládat, že každá zpráva, která má být důvěrná, se může dostat do rukou někomu, komu není určena. Je proto projevem potřebné opatrnosti učinit kroky, které zajistí, že nezamýšlený příjemce bude mít přinejmenším velké problémy zprávě porozumět, nebo ještě lépe, vůbec ji nebude schopen přečíst. Rozsah škody, kterou může nezamýšlené odhalení zprávy způsobit, do velké míry závisí na čase, který uplynul mezi odposlechnutím zprávy a jejím rozluštěním. V některých případech stačí, aby mezi odposlechnutím a rozluštěním zprávy uplynul jeden den nebo dokonce jenom pár hodin, aby k žádné škodě nedošlo. Například rozhodnutí akcionáře prodat nebo koupit okamžitě velký balík akcií, nebo rozkaz armádního velitele zaútočit za rozbřesku následujícího dne. V jiných případech má informace dlouhodobou hodnotu a musí být proto utajena co nejdéle. Například zprávy týkající se plánování rozsáhlých vojenských operací. Úsilí, které musí vynaložit soupeř, oponent nebo nepřítel k tomu, aby rozluštil zachycenou zprávu, má proto velký význam. Jestliže neautorizovaný příjemce zprávy nedokáže zprávu rozluštit za použití nejlepších známých metod a nejvýkonnějších dostupných počítačů za dobu kratší, než po kterou je utajení obsahu zprávy důležité, tak si odesílatel může být relativně jistý. Nemůže si ale být zcela jistý, protože rozluštění nějakých dříve odeslaných zpráv může protivníkovi pomoci urychlit luštění následujících zpráv. Je také možné, že protivník objevil nějakou metodu, kterou odesílatel nezná, a je proto schopen luštit zprávy rychleji, než si odesílatel dokáže představit. To se například přihodilo německé armádě se šifrovacím zařízením Enigma

v průběhu druhé světové války.

1.1 Transpoziční šifry

Transpozice neboli přesmyčka spočívá ve změně pořadí znaků podle určitého pravidla. Například tak, že otevřený text je zapsán do tabulky po řádcích a šifrový text vznikne čtením sloupců téže tabulky.

1.1.1 Viegenérova šifra

Vytvoří se Viegenérova tabulka (tabulka o 26 řádcích, v každém řádku písmena od "a" do "z", v 2. řádku je na první místo posunut poslední znak předcházejícího řádku a tak se postupuje dále až v posledním řádku je převrácená abeceda). Pokud je heslo kratší než zpráva k zašifrování, tak se duplikuje tolikrát, aby v součtu mělo tolik znaků jako zpráva. Postupně vybíráme ze zprávy a hesla znaky, které si odpovídají pozicí. Znak ze zprávy určuje souřadnici sloupce a znak z hesla hledejte na prvním místě v jednotlivých řádcích Viegenérovi tabulky, tím určíte souřadnici řádku. Znak, jenž leží na určených souřadnicích, je šifra.

Jiný způsob je následující. Zjistí se pořadí znaku, získaného ze zprávy v abecedě. Označíme-li ho **n,** pak se o **n-1** pozic posune odpovídající znak v hesle.

Sepsáním těchto znaků za sebe vzniká šifrovaná zpráva.

0	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
1	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
2	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
3	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
4	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
5	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
6	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
7	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
8	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
9	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
10	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
11	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
12	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
13	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
14	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
15	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
16	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
17	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
18	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
19	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
20	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
21	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
22	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
23	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
24	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
25	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 1: Viegenérova tabulka – metoda Viegenérova šifra

Příklad:

Zpráva k zašifrování je „acryptosystem“ a použijeme heslo „michaeal“. Prvním znakem ve zprávě je písmeno **a**. V abecedě se nachází na první pozici. Posunutí bude tedy nulové (1-1=0). První znak v hesle, tedy zůstane stejný, v našem případě **m**.

zpráva:	a c r y p t o s y s t e m
klíč:	m i c h a e l m i c h a e
šifra:	m k t f p x z e g u a e q

Tabulka 1: Viegenérova šifra

1.1.2 Jednoduchá transpozice

Zpráva se poskládá do tabulky o rozměru **n** (řádků) x **m** (sloupců). Heslo pro zašifrování se skládá z číslic 1 - **m** různě zpřeházených. Jednotlivé číslice hesla se postupně přiřazují sloupcům tabulky. Zašifrovaný text se tvoří tak, že se do řádku opíšou písmena z vytvořené

tabulky, jejichž sloupec má nad sebou 1, pak se pokračuje dvojkou až po číslo m. Pokud má tabulka více buněk než je znaků v zadaném textu, pak se doplní písmenem x. Pro náš příklad jsem použil tabulku 6x4.

Příklad:

Zpráva je „zasifrovanitextuzpravy“. Heslo „123654“. Zprávu vepíšeme do tabulky 6x4. Délka zprávy je 22 znaků, zbývající místa v tabulce tedy doplníme písmenem **x**. Nad sloupce tabulky si napíšeme heslo. Zašifrovaná zpráva pak vznikne sepsáním čtveřic znaků z každého sloupce (1-6 postupně) vedle sebe. Pro náš příklad má tvar „zoefavxasatvrtpxfizxinuy“.

1	2	3	6	5	4
z	a	s	i	f	r
o	v	a	n	i	t
e	x	t	u	z	p
r	a	v	y	x	x

Tabulka 2: Jednoduchá transpozice

1.1.3 Dvojitá transpozice

Zadaný text se poskládá do tabulky o rozměru **n** (řádků) x **m** (sloupců). Heslo pro zašifrování se skládá z číslic 1-m různě zpřeházených. Jednotlivé číslice hesla se postupně přiřazují sloupcům tabulky. Poté se vytvoří tabulka **m** x **n**. Do jednotlivých řádků se opíší sloupce z první tabulky v pořadí od 1 po 6. Zašifrovaný text se tvoří tak, že se do řádku opíšou písmena z druhé tabulky, jejichž sloupec má nad sebou 1, pak se pokračuje dvojkou až po číslo **n**. Pokud má první tabulka více buněk než je znaků v zadaném textu, pak se doplní písmenem **x**.

Příklad:

Zpráva je „zasifrovanitextuzpravy“. První heslo například „123654“ a druhé heslo je „3214“. Zprávu vepíšeme do tabulky 6x4. Nad sloupce tabulky si napíšeme první heslo. Poté jednotlivé sloupce opíše ve správném pořadí do 1. -6. řádku druhé tabulky, která má rozměry 4x6. Nad sloupce v druhé tabulce doplníme druhé heslo. Zašifrovaná zpráva pak

vznikne sepsáním šestic znaků z každého sloupce (1-4 postupně) vedle sebe. Pro náš příklad má tvar „extpzuovatinzasrfifavxxy“.

1	2	3	6	5	4
z	a	s	i	f	r
o	v	a	n	i	t
e	x	t	u	z	p
r	a	v	y	x	x

Tabulka 3: Dvojitá transpozice první heslo

3	2	1	4
z	o	e	f
a	v	x	a
s	a	t	v
r	t	p	x
f	i	z	x
i	n	u	y

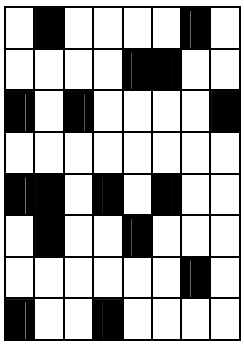
Tabulka 4: Dvojitá transpozice druhé heslo

1.1.4 Cardanova mřížka

Původně to byla mřížka vyrobená z tvrdého papíru, do níž byly vyřezány otvory, do kterých odesilatel vepsal zprávu, poté odejmul mřížku a vyplnil různými písmeny zbylá místa. V tomto případě má mřížka rozměr 8x8 (zpráva tedy může mít nejvýše 64 znaků), ve které je 16 otvorů, do kterých se vpisuje zpráva. Vězněme tedy mřížku, nastavíme do první polohy a vpisujeme do dírek písmena tak jako bychom psali zprávu, tzn. po řádcích. Až vyplníme všechna políčka, otočíme mřížku o 90 stupňů a pokračujeme v psaní zprávy opět od prvního řádku, až dojdeme opět na konec mřížky. Tento postup zopakujeme ještě 2krát a výsledkem je vyplněná tabulka 8x8, tedy zašifrovaný text. Pokud máme zprávu kratší jak 64 znaků, tak nakonec doplníme různé znaky.

Příklad:

Zprávu k zašifrování zvolíme ve tvaru „nicnenitaktezkeabysetohonemohlodosahnout“. Postupně tedy vepisujeme jednotlivé znaky do tabulky, dokud jsou prázdné políčka. Až se zaplní první tabulka. Otočíme tabulkou o 90° a pokračujeme tímto stylem, až se zaplní celá tabulka 8x8.



Tabulka 5: Jedna z Cardanových mřížek

b	n	d	y	o	s	i	s
d	a	e	t	c	n	d	o
e	d	n	h	d	h	n	i
o	d	o	n	u	d	t	c
t	a	e	k	d	t	m	d
c	e	d	o	z	c	h	c
d	l	c	c	d	d	k	o
e	c	d	a	d	d	c	d

Tabulka 6: Výsledná Cardanova mřížka pro náš příklad

1.2 Substituční šifry

Je druh šifry, kdy se nahrazuje každý znak otevřeného textu jiným znakem šifrovaného textu. Aby příjemce získal otevřený text, musí na zašifrovaný text použít inverzní substituci.

1.2.1 Cézarova šifra

Zpráva určená k zašifrování je rozložena na jednotlivé znaky a každý se změní na znak, který bychom našli v abecedě, pokud bychom se posunuli o **n** míst směrem dopředu. Zašifrovaný text se pak získá sepsáním zašifrovaných znaků za sebe.

Příklad:

Zpráva je „ahojkamarade“, posunutí si určíme 4. Podle výše uvedeného postupu pak vznikne zašifrovaná zpráva „elsnoeqevehi“

1.2.2 Jednoduchá substituce v tabulce

Zadaný text se rozloží na jednotlivé znaky a ty se podle zadání uživatele vymění za definované znaky, čím vznikne samotná šifra.

Příklad:

Máme zprávu „ahoj“ a definujeme si dvě výměny. Písmeno **a** zaměníme za **c** a písmeno **o** za **p**. Tím vznikne zašifrovaná zpráva „chpj“.

1.2.3 Autokláv

Postupně se vybírají **x**-té znaky zprávy a hesla, dokud nedojdou písmena ve zprávě. Používá se třeba Viegenerova tabulka (26 abeced pod sebou, **n**-tá abeceda má posunutý začátek o **n** pozic doprava a na konci vznikne inverzní abeceda). Označíme-li tuto tabulku **n** x **m**, pak pozice znaku z hesla v abecedě udává **n**-tou souřadnici řádku a pozice znaku ve zprávě udává **m**-tou souřadnici sloupce ve Viegenerově tabulce, přičemž písmeno **z** v hesle je 0tý řádek v tabulce. Znak, který v tabulce leží na této pozici, je pak hledaným šifrovaným znakem. Pokud je heslo kratší než zpráva, pak se doplňuje buď zprávou (použití abecedy otevřeného textu), nebo již zašifrovaným textem (použití abecedy šifrovaného textu).

Příklad:

Zpráva k zašifrování je „ahojkamarade“ a heslo dáme „abeceda“. Heslo je kratší než zpráva, doplníme tedy heslo na tvar „abecedaahoj“ v případě použití původní zprávy k doplnění hesla. Pokud bychom heslo doplňovali o zašifrovanou zprávu, pak by heslo mělo tvar „abecedabjtm“. Podle výše uvedeného postupu vyhledáme ve Viegenérově tabulce zašifrované znaky. Nakonec nám vyjde šifra ve tvaru „bjtmpenbzpnp“.

Pro připomenutí a přehlednost uvádím ještě jednou Viegenérovu tabulku.

0	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
1	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
2	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
3	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
4	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
5	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
6	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
7	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
8	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
9	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
10	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
11	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
12	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
13	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
14	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
15	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
16	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
17	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
18	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
19	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
20	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
21	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
22	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
23	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
24	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
25	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 2: Viegenérova tabulka – metoda Autokláv

1.2.4 Jednoduchá záměna

1.2.4.1 Srovnaná abeceda

Vytvoří se klasická abeceda **a** až **z** a srovnaná abeceda (abeceda šifrovaného textu), což je klasická abeceda, ale první prvek tedy **a** je posunut o uživatelem definovaný počet pozic směrem doprava. Při každém posunutí se poslední prvek abecedy přesouvá na místo prvního prvku a ten je posunut na bezprostředně následující pozici. Toto se provede **n**-krát, kde **n** je posunutí. Dále se určí první prvek ve zprávě, kterou chceme zašifrovat. Najde se jeho pozice v klasické abecedě a zašifrovaným znakem je pak prvek ležící na stejné pozici, ale ve srovnané abecedě. Tohle se opakuje pro celý řetězec určený k zašifrování.

Příklad:

Mějme zprávu „sifra“ a posunutí nastavíme na 4. Pak si vytvoříme abecedu šifrovaného textu (AŠT). A vytvoříme si pro přehlednost tabulku, která bude mít v prvním řádku abecedu otevřeného textu (AOT) a pod ní v druhém řádku je AŠT.

AOT	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
AŠT	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

Tabulka 7: Jednoduchá záměna – srovnaná abeceda

Zašifrovaná zpráva má pak tvar „oebnw“.

1.2.4.2 Reciproká abeceda

Vytvoří se klasická abeceda **a** až **z** a reciproká abeceda (abeceda šifrovaného textu), což je klasická abeceda v inverzním (obráceném) tvaru. Dále se určí první prvek v řetězci, který chceme zašifrovat. Najde se jeho pozice v klasické abecedě a zašifrovaným znakem je pak prvek ležící na stejné pozici, ale v reciproké abecedě. Tohle se opakuje pro celý řetězec určený k zašifrování.

Příklad:

Mějme zprávu „sifra“. Pak si vytvoříme abecedu šifrovaného textu (AŠT). A vytvoříme si pro přehlednost tabulku, která bude mít v prvním řádku abecedu otevřeného textu (AOT) a pod ní v druhém řádku je AŠT.

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

Tabulka 8: Jednoduchá záměna – reciproká abeceda

Zašifrovaná zpráva má pak tvar „hruiz“.

1.2.4.3 Rozházená abeceda

Vytvoří se klasická abeceda **a** až **z** a rozházená abeceda (abeceda šifrovaného textu), což je klasická abeceda, ale její prvky jsou náhodně rozházené do pole o délce 26 pozicí (26 znaků abecedy). Dále se určí první prvek v řetězci, který chceme zašifrovat. Najde se jeho pozice v klasické abecedě a zašifrovaným znakem je pak prvek ležící na stejné pozici, ale v rozházené abecedě. Tohle se opakuje pro celý řetězec určený k zašifrování.

Příklad:

Mějme zprávu „sifra“. Pak si vytvoříme abecedu šifrovaného textu (AŠT). A vytvoříme si pro přehlednost tabulku, která bude mít v prvním řádku abecedu otevřeného textu (AOT) a pod ní v druhém řádku je AŠT.

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

Tabulka 9: Jednoduchá záměna – rozházená abeceda

Zašifrovaná zpráva má pak tvar „inahp“.

1.2.5 Tabulka 4x7

Vytvoří se tabulka 4x7. Řádky tabulky jsou označeny čísly 7, 8, 9 a 0. První sloupec nezastupuje žádné číslo, dalších 6 sloupců je označeno čísly 1 - 6. Šifrování probíhá tak, že místo znaků zprávy se dosadí jejich souřadnice z tabulky.

Příklad:

Zpráva k zašifrování je „ahojpepo“. Nalezneme tedy jednotlivé znaky v tabulce a určíme číselnou šifru každého znaku. Nejprve se píše číslo řádku a pak sloupce.

	1	2	3	4	5	6
7	a	b	c	d	e	f	g
8	h	i	j	k	l	m	n
9	o	p	q	r	s	t	u
0	v	w	x	y	z	;	§

Tabulka 10: Tabulka 4x7

Zašifrovaná zpráva má poté tvar „789829174919“.

1.2.6 Tabulka 5x10

Vytvoří se tabulka 5x10. Řádky tabulky jsou označeny čísly 1-5. Sloupce čísly 1 - 9 a poslední označíme číslem 0. Tabulka je vyplněna znaky české abecedy. Nevyplněná políčka v tabulce doplníme různými znaky. Šifrování probíhá tak, že místo znaků zprávy se dosadí jejich souřadnice z tabulky a to v pořadí řádek-sloupec.

Příklad:

Jako zprávu si zvolíme text „kryptosystém“. Nalezneme tedy jednotlivé znaky v tabulce. Sepíšeme vedle sebe souřadnice řádků a sloupců.

	1	2	3	4	5	6	7	8	9	0
1	a	á	b	c	č	d	ď	e	é	ě
2	f	g	h	i	í	j	k	l	m	n
3	ň	o	ó	p	q	r	ř	s	š	t
4	ť	u	ú	ů	v	w	x	y	ý	z
5	ž	;	-	*	+	=	§	)	(	,

Tabulka 11: Tabulky 5x10

Tímto způsobem vznikne šifra ve tvaru „27364834303238483830102 9“.

1.2.7 Playfair

Jsou nadefinovány 4 matice 5x5 (25 písmen abecedy, znak **x** se vynechá)

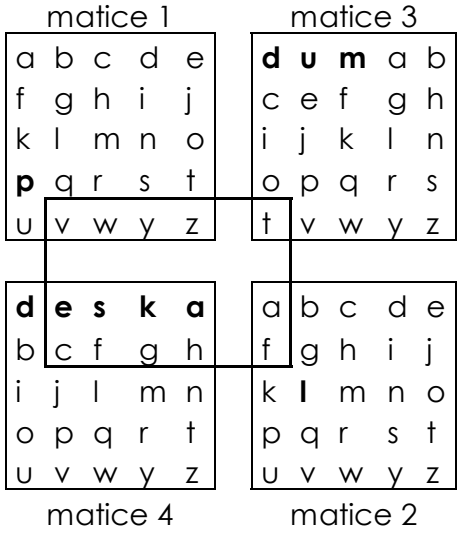
matice 1	matice 3
matice 4	matice 2

Tabulka 12: Playfair - rozložení matic

Definují se dvě hesla. První heslo obsadí první políčka matice 3, ostatní políčka obsadí písmena nevyskytující se v hesle. Druhé heslo obsadí první políčka matice 4. Heslo nesmí obsahovat dvě a více stejných písmen. Dále se definuje zpráva k zašifrování. Znaky zprávy se rozdělí postupně do dvojic. Šifrování probíhá tímto způsobem. První písmeno z každé dvojice se nalezne v matici 1. Jeho řádková souřadnice definuje řádkovou souřadnici písmene v matici 3 a jeho sloupcová souřadnice definuje sloupcovou souřadnici písmene v matici 4. Druhé písmeno z každé dvojice se nalezne v matici 2. Jeho řádková souřadnice definuje řádkovou souřadnici písmene v matici 4 a jeho sloupcová souřadnice definuje sloupcovou souřadnici písmene v matici 3. Písmena nalezená v maticích 3,4 jsou zašifrovanými písmeny dané dvojice znaků za matice 1,2. Opakováním tohoto postupu se zašifruje celý text. Tento postup se používá hlavně k programování. V příkladu si ukáže způsob, kdy máme k dispozici grafické zobrazení čtyř matic.

Příklad:

Zpráva nechť má tvar „playfair“. První heslo je „dum“ a druhé heslo „deska“. Vytvoříme si tedy čtyři matice o rozměru 5x5. Matice 1 a 2 obsahuje klasickou abecedu kromě písmene **x.** Do matice 3 a 4 nejprve vyplníme hesla a poté doplníme zbývajícími písmeny z abecedy. V matici 1 nalezneme první písmeno ze zprávy a v matici 2 druhé písmeno. Vytvoříme si pomyslný čtyřúhelník, jehož dva vrcholy se dotýkají těchto dvou písmen. Na dalších dvou vrcholech jsou jejich zašifrované protějšky. Tímto způsobem pokračujeme s dalšími dvojicemi. V ukázce jsem použil první dvojici písmen ve zprávě.

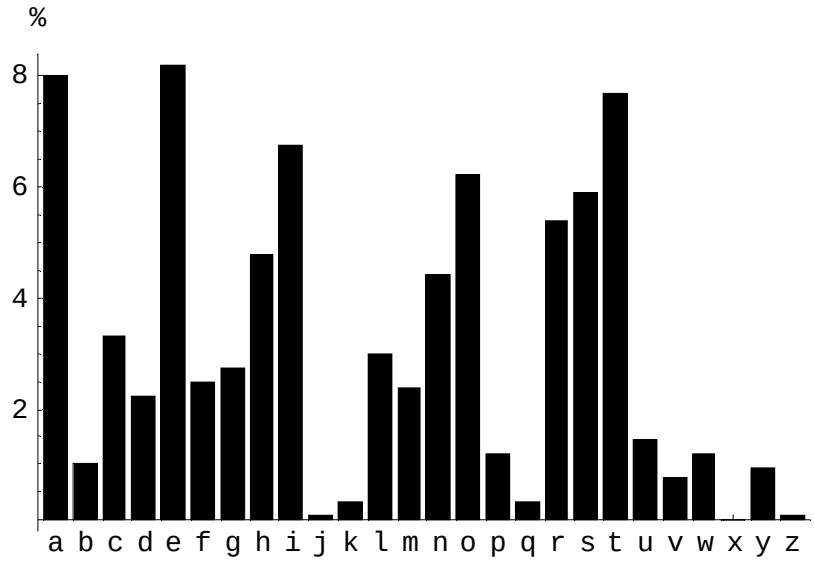


Tabulka 13: Metoda Playfair - příklad

Zašifrovaná zpráva má tvar „piaucdfr“.

2 HISTOGRAM

Histogram textu je grafické znázornění četnosti znaků v daném textu. Většinou se zobrazuje procentuální výskyt těchto znaků. Na obrázku 3 vidíme histogram textu, kde na souřadnici **y** je procentuální vyjádření četnosti znaků v textu a na ose **x** jsou vypsané znaky, pro které se četnosti počítali.



Obrázek 3. Histogram textu

3 WEBMATHEMATICA

3.1 Úvod do aplikace webMathematica

WebMathematica nám umožňuje přenést vytvořené notebooky z programu Mathematica na web. WebMathematica slouží pro lidi, co nemají Mathematicu doma, ale přes web mohou s ní pracovat. Přidává tedy interaktivní výpočty a vizualizace do vašich webových stránek díky propojení programu Mathematica s nejnovějšími serverovými technologiemi.

WebMathematica a Mathematica jsou v základu stejné, ale poskytují jiné uživatelské rozhraní a jsou zaměřeny na rozdílné skupiny uživatelů.

WebMathematica nabízí přístup ke specifickým aplikacím programu Mathematica skrz webový prohlížeč nebo jiného web klienta. Orientace v prostředí webMathematica pro uživatele nevyžaduje zvláštní znalosti programování. Ve většině případů uživatelé nemusí být obeznámeni s programem Mathematica, dokonce nemusí ani vědět, že Mathematicu používají.

Mathematica tedy poskytuje vývojové prostředí pro webMathematica web stránky. Například v programu Mathematica se vytvoří kód, který modeluje nějaký fyzický proces. Tento kód může být poté umístěn na webMathematica stránky, čímž umožníme lidem tento model spustit a řídit odkudkoliv na světě a požít ho tak pro zjištění výsledků pro své projekty.

Detaily k programování v programu webMathematica budou následovat v praktické části této bakalářské práce.

II. PRAKTICKÁ ČÁST

4 WEBMATHEMATICA

4.1 Programovaní stránek webMathematica

Když se pustíme poprvé do programování stránek, tak nás asi napadnou tři základní úkony. Zařadil bych mezi ně **základní obsah dokumentu**. Tedy základní elementy dokumentu, které by měl obsahovat, aby si s jeho překladem webMathematica poradila.

Dalším úkonem je vstup od uživatele, tedy **získávání dat**. Konkrétně v metodách šifrování je to zpráva k zašifrování a hesla (ať už ve formě textu nebo čísel).

A tím třetím by nejspíš bylo **zobrazení výsledku**. Na co by nám taky bylo šifrování, když bychom se nedozvěděli šifru či výsledek kteréhokoliv výpočtu provedeného „srdcem“ aplikace webMathematica.

Ještě je tady však jeden důležitý úkon a jím je naprogramování samotné šifry či jiné konkrétní aplikace, které by měla řešit daný technologický problém a tento kód vložit do webové stránky. Ale tohle už je za hranicemi této kapitoly. Jen připomenu, že znalost programování v prostředí Mathematica je jeden z předpokladů, ke zvládnutí programu webMathematica.

4.1.1 Základní obsah dokumentu webMathematica

Základní vzhled kódu, který webMathematica umí zpracovat je následující.

```
1  <%@ taglib uri="/webMathematica-taglib" prefix="msp" %>
2  <%@ page contentType="text/html; charset=windows-1250"
   language="java" import="java.sql.*" errorPage="" %>
3  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
4  <html xmlns="http://www.w3.org/1999/xhtml">
5  <head>
6  <meta http-equiv="Content-Type" content="text/html; charset=windows-
   1250" />
7  <title>titulek stránky</title>
8  </head>
9
10 <body>
11 <msp:allocateKernel>
12 <form action="url_stranky.jsp" method="post">
```

```
13 <msp:evaluate>
14     Needs[ "AGO`Algorithms`"];
15 </msp:evaluate>
16 <msp:evaluate>
17
18 </msp:evaluate>
19 </form>
20
21 </msp:allocateKernel>
22 </body>
23 </html>
```

Vysvětlení jednotlivých řádků (číslování odpovídá jednotlivým řádkům v kódu):

1. Standardní jsp hlavička, která nám říká, že v kódu je používána syntaxe, jenž zpracovává weMathematica a prvky této syntaxe mají předponu **msp.**
2. Nastavení typu obsahu a kódovaní javascriptu.
3. Typ dokumentu HTML.
4. - 23. Obsah HTML dokumentu.
5. - 8. Hlavička HTML dokumentu.

10. – 22. Tělo HTML dokumentu.

11. Přiřadí stránce kernel programu Mathematica, umístěný např. na serveru, k výpočtům použitým v kódu stránky. Počáteční tag.
12. - 19. Obsah formuláře, který se po potvrzení odesílá na URL stránky uvedené v parametru *action* metodou `post`.
13. - 15. Obsah těchto tagů je odeslán kernelu Mathematica ke zpracování.
14. Vyžádání si vnější knihovny pro výpočty.

21. Uvolnění kernelu Mathematica.

4.1.2 Získávání dat

Většina komunikace s uživatelem a získávání dat od něj probíhá přes formulář a v něm umístěné prvky <`input> </input>`.

Ukázka formuláře:

```
1 ..
2 <form action="url_stranky.jsp" method="post">
3 <input type="text" name="retezec"  /><br />
4 <input type="submit" value="Odeslat" name="odeslat" />
5 <msp:evaluate>
6 promena=$$retezec;
7 </msp:evaluate>
8 </form>
9 ..
```

Tag na 3. řádku zobrazí ve formuláři prázdné okýnko, do kterého uživatel zadává jakýkoliv text. Důležitým parametrem tohoto tagu je `name.` Pomocí tohoto názvu se po odeslání formuláře získávají uživatelem sdělená data (text) do proměnných, které dále zpracovává jádro programu Mathematica. Toto „přenesení“ dat z políčka formuláře do proměnné se provádí v příkazu na 6. řádku výše uvedeného kódu. Obecná syntaxe je následující.

```
nazev_promenne=$$nazev_prvku_input_z_formulare;
```

Z uvedeného označení tedy vyplývá, že ať už bereme data z formuláře pomocí prvku `input` nebo třeba `textarea`, vždy se před jeho název dají dva znaky dolaru - $$. Následné zpracování proměnných už probíhá tak jak jej známe z programování v programu Mathematica. Pro doplnění uvedu, že prvek `input` na 4. řádku slouží k potvrzení odeslání formuláře ke zpracování.

4.1.3 Zobrazení výsledku

V předchozí kapitole jsme si ukázali jak od uživatele získat data ke zpracování. Data prošla algoritmem některé z našich šifer. A nyní bych se rád pokusil vysvětlit zobrazení výsledků výpočtu na níže uvedených výřezech ze zdrojového kódu.

V první ukázce jsem použil jednoduché uložení hodnoty do proměnné (4. řádek) a následný výpis obsahu dané proměnné. Na stránce se zobrazí číslo „**5**“.

```
1 ...
2 <msp:allocateKernel>
3 <msp:evaluate>
4  x=5; x
5 </msp:evaluate>
6 </msp:allocateKernel>
7 ...
```

Následující ukázka obsahuje příkaz **Print[]**. Ten určitě znají nejen programátoři Mathematica. Uvnitř hranatých závorek je většinou nějaký text nebo proměnná určená k výpisu na obrazovku. Při použití tohoto příkazu v programu Mathematica je obsah závorek okamžitě vypsán na obrazovku. V prostředí webMathematica to však takto nepracuje. Obsah jednotlivých Print[] se ukládá postupně do schránky a tam sečkají na příkaz, který je naráz vypíše na obrazovku. Tímto příkazem je `ColumnForm[MSPGetPrintOutput[]].`

Jeho nevýhodou je, že nedokáže zvlášť vypsat jednotlivé výstupy. V naší ukázce jsem schválně uvedl dvakrát tento příkaz a vložil jsem mezi ně další text pro výpis na obrazovku (14. řádek). Co tedy udělá níže uvedený kód? Příkaz na 11. řádku zobrazí zprávu uvedenou na 5. řádku, tedy „Výsledek výpočtu je: 4“. Na tom by nebylo nic divného. Avšak na řadu přichází 17. řádek, který tuto zprávu vypíše znovu a přidá k ní další, která je uvedený na 14. řádku našeho kódu.

```
 1 ...
 2 <msp:allocateKernel>
 3 <msp:evaluate>
 4 x=2;
 5 Print[ "Výsledek výpočtu je: ", x^2];
 6  </msp:evaluate>
 7 ...
 8     nějaký kód
 9 ...
10 <msp:evaluate>
11     ColumnForm[ MSPGetPrintOutput[]]
12 </msp:evaluate>
13 <msp:evaluate>
14 Print[ "Další text k zobrazení"];
15  </msp:evaluate>
```

```
16 <msp:evaluate>
17     ColumnForm[ MSPGetPrintOutput[]]
18 </msp:evaluate>
19 </msp:allocateKernel>
20 ...
```

Poslední způsob, který bych rád uvedl, jsem používal nejčastěji. Používá se příkaz **StringJoin[]**, který už známe z programu Mathematica. Níže uvedený příklad vypíše „Proměnná x obsahuje hodnotu: 2“. Jak už z názvu vyplívá, jde o spojení dvou a více řetězců. Přičemž řetězcem rozumějme text v uvozovkách, nebo proměnná, které byl text někde v předchozích řádcích kódu přidělen.

Pozn. Číselný obsah proměnné lze převést na řetězec příkazem **ToString[promenna]**.

```
21 ...
22 <msp:allocateKernel>
23 <msp:evaluate>
24 x=2;
25 </msp:evaluate>
26 ...
27     nějaký kód
28 ...
29 <msp:evaluate>
30     StringJoin[“Proměnná x obsahuje hodnotu: ”,ToString[x]]
31 </msp:evaluate>
32 </msp:allocateKernel>
33 ...
```

4.2 WWW stránky

Jsou praktickou ukázkou programování aplikací pro webMathematicu.

4.2.1 Grafické zobrazení

Jsou graficky navrženy tak, aby byli rychlé i pro majitele pomalejších připojení k Internetu. Tudíž nejsou tvořeny z obrázků umístěných v tabulce, která tvoří strukturu stránky, ale

99% grafického zobrazení je vytvořeno CSS styly (to jedno procento je pro obrázek v levém horním rohu).

4.2.2 Metody programování

K naprogramování obsahu byl použit JavaScript (menu stránky), XHTML a samozřejmě Mathematica (veškeré algoritmy uvedených šifer).

4.2.3 Struktura www stránek

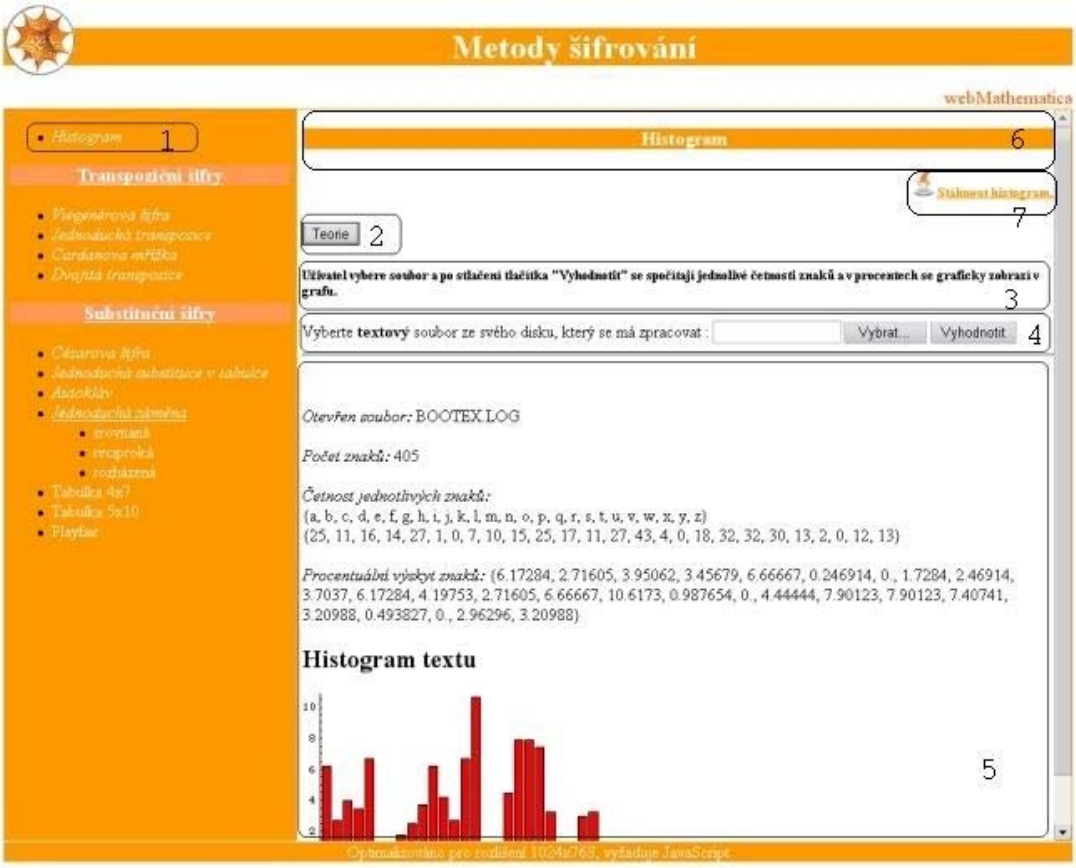
Stránka je tvořena neměnným záhlavím (s nápisem Metody šifrování) a zápatím. Dynamickou částí stránky je menu (1) obsahující názvy a zároveň odkazy na praktické ukázky některých klasických šifer zpracovaných v Mathematice a implementovaných do prostředí webMathematica. Odkazy z menu jsou nasměrovány do prostorově největší části stránky (2). Po prvním načtení stránky je na tomto místě stránka s krátkým úvodem do problematiky.

Obrázek 4: Vzhled www stránky

4.2.4 Ovládání stránek

Kliknutím na odkaz (1) v menu se v hlavní části stránek zobrazí daná problematika. V horní části je uveden název problematiky (6). Dále je zde tlačítko „Teorie“ (2), po jejímž stlačení se bezprostředně pod ním zobrazí teorie daného příkladu šifry, nebo třeba histogramu jak je na následujícím obrázku. V následující oblasti (4) se nachází „okýnka“ pro vkládání dat uživatelem a potvrzovací tlačítka pro odeslání těchto dat. Po odeslání dat se výsledek zobrazí v oblasti (5). Poslední na co bych chtěl na každém příkladu šifer nebo histogramu upozornit je oblast (7), kde se nachází odkaz pro stáhnutí dané šifry v souboru s příponou *.nb, se kterou si poradí Mathematica.

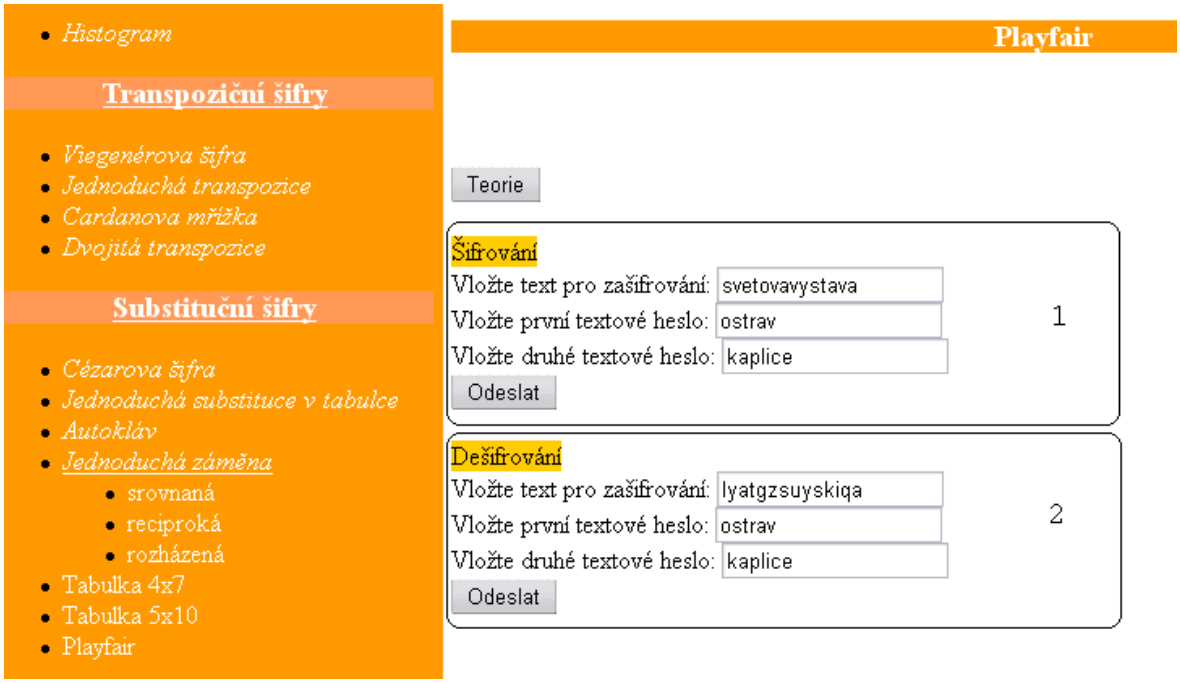
Každá šifra nebo histogram mají jiné požadavky na vstupní data, tudíž se mění i obsah prvků v oblasti (4).



Obrázek 5:Ovládání stránek

Dalším typickým vzhledem oblasti (4) z obrázku 5 může být i následující výřez ze stránky s metodou Playfair zobrazený na obrázku 6.

Oblast je rozdělená na dvě části a to na část pro data k zašifrování (1) a pro data k dešifrování (2).



Obrázek 6: Výřez oblasti pro zadávání dat uživatelem

4.2.5 Ukázka zdrojového textu šifry a její popis

Vybral jsem šifru jednoduchá záměna s použitím rozházené abecedy. Nejprve uvedu a vysvětlím zdrojový text z www stránek a poté pro porovnání i zdrojový text z Mathematici. Zdrojové texty ostatních šifer mají podobnou strukturu, jen se liší v některých použitých funkcích.

4.2.5.1 Zdrojový text pro aplikaci webMathematica

Název souboru je JednZam_ROA.jsp.

```
1 <%@ taglib uri="/webMathematica-taglib" prefix="msp" %>
2 <%@ page contentType="text/html; charset=windows-1250"
  language="java" import="java.sql.*" errorPage="" %>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
4 <html xmlns="http://www.w3.org/1999/xhtml">
5
6 <head>
```

```
 7 <meta http-equiv="Content-Type" content="text/html; charset=windows-
   1250" />
 8 <link href="grafika.css" type="text/css" rel="stylesheet" />
 9 <script language="javascript" src="grafika.js"></script>
10 <title>Jednoduchá záměna-Rozházená abeceda</title>
11 </head>
12 <body>
13 <h3>Jednoduchá záměna-Rozházená abeceda</h3>
14 <a href="./nb/Jednoducha_zam_ROA.nb" ><h5 class="download"><img
   src="images/download-icon.bmp" height="25" width="25" alt="download"
   />Stáhnout Jednoducha_zam_ROA.nb</h5></a>
15 <input type="button" class="ukaz" onclick="Skryj_Zobraz(Div1)"
   value="Teorie" />
16 <h5><div id="Div1" class="skryte">Vytvoří se klasická abeceda
   <strong>a</strong>..<strong>z</strong> a rozházená abeceda, což je
   klasická abeceda, ale její prvky jsou náhodně rozházené do pole o
   délce 26 pozicích(26 znaků abecedy). Dále se určí první prvek v
   řetězci, který chceme zašifrovat. Najde se jeho pozice v klasické
   abecede a zašifrovaným znakem je pak prvek ležící na stejné pozici,
   ale v rozházené abecede. Tohle se opakuje pro celý řetězec určený k
   zašifrování.</div></h5>
17 <form action="JednZam_ROA.jsp" method="post">
18 <msp:allocateKernel>
19 <msp:evaluate>
20     Needs["AGO`Algorithms`"];
21 </msp:evaluate>
22
23 <span class="sifrovani">Šifrování</span><br />
24 Zadejte řetězec k zašifrování:
25 <input type="text" name="retezec" value="ahojpepino" maxlength="26"
   />
26 <input type="submit" name="odeslat" value="Zašifrovat" />
27 <msp:evaluate>
28 If[ValueQ[$$odeslat],
29 Clear["in","j",j,"i",i,"x",x];
30 pomocna = Characters[$$retezec];
31 delka = Length[pomocna];
32 abeceda = CharacterRange["a", "z"];
33 rozabeceda = {p, q, m, z, l, a, o, x, n, w, s, k, c, j, e, u, d, h,
   i, r, y, v, b, f, g, t};
34 sifrovany ="";pocitadlo=0;
35 For[pocitadlo = 0, pocitadlo < delka, ++pocitadlo;
36   pomoc2 = Position[abeceda, pomocna[[pocitadlo]]];
37   sifrovany = StringJoin[sifrovany, ToString[rozabeceda[[pomoc2[[1,
   1]]]]]];
```

```
38   ]
39 ]
40 </msp:evaluate>
41 <br />
42 <msp:evaluate>
43 vypis="";
44 If[ValueQ[$$odeslat],
45 StringJoin["Zašifrovaný text: ",sifrovany,"<br />Klasická abeceda:
   ",abeceda,"<br />Rozházená abeceda: ",ToString[rozabeceda]]
46 ]
47 </msp:evaluate>
48 <hr />
49 <span class="desifrovani">Dešifrování</span><br />
50 Zadejte řetězec k dešifrování:
51 <input type="text" name="retezec2" value="pxewulunje" maxlength="26"
   />
52 <input type="submit" name="odeslat2" value="Dešifrovat" />
53 <msp:evaluate>
54 If[ValueQ[$$odeslat2],
55 pomocna2 = Characters[$$retezec2];
56 delka2 = Length[pomocna2];
57 abeceda = CharacterRange["a", "z"];
58 pomoc2 = {p, q, m, z, l, a, o, x, n, w, s, k, c, j, e, u, d, h, i,
   r, y, v, b, f, g, t};
59 desifrovany = "";
60 For[pocitadlo = 0, pocitadlo < delka2, ++pocitadlo;
61   pozice2 = Position[pomoc2, ToExpression[pomocna2[[pocitadlo]]]];
62   desifrovany = StringJoin[desifrovany,
   ToString[abeceda[[pozice2[[1,1]]]]]]
63   ]
64 ]
65 </msp:evaluate>
66 <br />
67 <msp:evaluate>
68 vypis="";
69 If[ValueQ[$$odeslat2],
70 StringJoin["Dešifrovaný text: ",desifrovany,"<br />Klasická abeceda:
   ",abeceda,"<br />Rozházená abeceda: ",ToString[pomoc2]]
71 ]
72 </msp:evaluate>
73 </form>
```

```
74 </msp:allocateKernel>
75 </body>
76 </html>
```

Popis zdrojového textu:

Řádky 1-12 přeskočím, protože jejich význam jsme si už osvětlili v sekci 4.1.1. Dále budu psát čísla řádků a k nim popis.

13. Hlavička s názvem šifry, jejíž styl je v souboru grafika.css .

14. Hypertextovy odkaz na soubor Jednoducha_zam_ROA.nb, který lze kliknutím na tento odkaz na stránkách stáhnout na Váš disk a otevřít v Mathematice.

15. Tlačítko s nápisem „Teorie“ po jejímž stlačení se volá funkce JavaScriptu s parametrem Div1 (což je id prvku na řádku 16) „Skryj_Zobraz(Div1)", která je zapsána v souboru grafika.js.

16. Teorie k dané šifře.

17. Začátek formuláře s cílovou stránkou viegener.jsp a data se posílají metodou post.

18. Alokace výpočetního jádra na serveru webMathematici pro následující výpočty.

19. Záčatek buňky s příkazy pro Kernel (dále již nebudu zmiňovat).

20. Vyvolání knihovny pro výpočty.

21. Konec buňky s příkazy pro Kernel (dále již nebudu zmiňovat).

23. Prvek slouží pro řádkou úpravu stylu textu. Má nastavenou třídu „sifrovani“, jejíž styl je popsán v souboru grafika.css. A tento prvek definuje nadpis „Šifrování“.

24. Textové sdělení vypsané přímo na obrazovku.

25. Prvek <input> s parametrem typu použití nastaveným na „text“ a názvem „retezec“. Pokud je potřeba získat data vložená do tohoto textového pole, použije se právě tento název „retezec“. Maximální délka řetezce nastavena na 26 znaků. Přednastavenou hodnotou pro testování je řetězec „ahojpepino“.

26. Tento prvek má nastavený typ na „submit“ a ten z něj udělá tlačítko pro odeslání. Parametr value nastaví popisek na tlačítku a parametrem name je název pro využití stavu tlačítka v další části formuláře.

28. Zde se testuje, zda bylo stlačeno tlačítko s názvem „odeslat“ (jeho zdrojový text se nachází na 26. řádku). ValueQ má dvě hodnoty. Pokud vrátí hodnotu „1“, pak to znamená, že tlačítko „odeslat“ bylo zmáčknuto. Hodnota „0“ značí, že uživatel ještě nestlačil toto tlačítko, tudíž se následující příkazy (zahrnuté v hranatých závorkách If) nebudou provadět.

29. Uvolnění proměnných uvnitř závorek.

30. Vytvoří se proměnná „pomocna“ typu pole, jehož prvky jsou jednotlivé znaky řetězce, který uživatel zadal do prvku <input> s názvem “retezec” ve formuláři.

31. Vznikne proměnná „delka“ do níž se uloží délka předcházející proměnné.

32. Do proměnné „abeceda“ se uloží abeceda „a“ – „z“.

33. Vytvoříme proměnnou „rozabeceda“, což je klasická abeceda, jejíž prvky jsou náhodně přemístěny. Později se použije k zašifrování textu.

34. Inicializace proměnných „sifrovany“ a „pocitadlo“ na prázdnou resp. nulovou hodnotu.

35. – 39. Cyklus for, jenž probíhá tak dlouho, dokud nedojde na konec uživatelem zadaného řetězce. Do proměnné „pomoc2“ se uloží pozice právě zpracovávaného znaku (pomocna[[pocitadlo]]) v abecedě, neboli v proměnné „abeceda“. Následně se najde prvek, který leží na právě získané pozici, avšak v proměnné „rozabeceda“ (ToString[rozabeceda[[pomoc2[[1, 1]]]]], tady probíhá šifrování) a obdržený znak se poté zapíše na konec řetězce uloženého v proměnné „siforovany“.

43.

44. Znovu testování stlačení tlačítka „odeslat“.

45. – 46. Výpis výsledku šifrování a tvaru rozházené abecedy.

48. Horizontální čára.

49. – 71. Zde probíhá opačný proces, tedy dešifrování. Postup je velmi podobný, takže zmíním pouze rozdíly. Řetězec k dešifrování má název „retezec2“. Tlačítko pro odeslání textu k dešifrování má tentokrát název „odeslat2“, čímž se rozliší úmysl uživatele. Jestli se jedná o šifrování, obdrží hodnotu „1“ pouze od tlačítka „odeslat“, v případě dešifrování bude hodnotu jedna obsahovat „odeslat2“. Dešifrovaný text je pak uložen v proměnné „desifrovany“ a ten se pak použije při výpisu dešifrovaného textu.

73. Zde končí obsah formuláře.

74. Uvolnění kernelu pro jiné výpočty.

75. Konec těla dokumentu.

76. Konec dokumentu HTML.

4.2.5.2 Zdrojový text pro aplikaci Mathematica

Název souboru je Jednoducha_zam_ROA.nb.

zdrojový text pro šifrování textu:

```
1  retezec="ahojpepino";
2  pomocna=Characters[retezec];
3  delka=Length[pomocna];
4  abeceda=CharacterRange["a","z"];
5  pomoc={p, q, m, z, l, a, o, x, n, w, s, k, c, j,i, e, u, d, h, r, y,
   v, b, f, g, t};
6  sifrovany="";
7  For[pocitadlo=0,pocitadlo<delka,++pocitadlo;
8    pozice=Position[abeceda,pomocna[[pocitadlo]]];
9    sifrovany=StringJoin[sifrovany,ToString[pomoc[[pozice[[1,1]]]]] ]
10   ]
11 sifrovany
```

Zašifrovaný text: pxiwelenji

Popis zdrojového textu šifrování:

1. Do proměnné „retezec“ se uloží řetězec uvedený v uvozovkách za znakem rovnítka.
2. Proměnná s názvem „pomocna“ je tímto definována jako pole, které je postupně naplněno jednotlivými znaky z řetězce uloženého v proměnné „retezec“.
3. Zjištění délky pole „pomocna“ a uložení této hodnoty do proměnné „delka“.
4. Vytvoření pole „abeceda“ a jeho naplnění znaky z rozsahu „a“ až „z“.
5. Pole „pomoc“ s obsahem uvedeným v závorkách.
6. Inicializace proměnné „sifrovany“.
7. – 11. Cyklus For, ve kterém se nejprve na 9. řádku zjišťuje pozice jednotlivých znaků získaných z proměnné „pomocna“ v poli s názvem „abeceda“ a uložení

získané hodnoty do „pozice“. Poté na 10. řádku následuje zjištění prvku, který leží na právě získané pozici, avšak v poli „pomoc“. Tento znak se poté přidá na konec řetězce, jenž je uložen v proměnné „sifrovany“. Cyklus proběhne tolikrát, jak dlouhý je řetězec určený k šifrování (proměnná „delka“).

12. Výpis proměnné „sifrovany“. Vzniklá šifra ukázkového textu (2. řádek zdrojového textu) je „pixwelei“, uvedl jsem ho do ukázky pro dešifrování, abychom si ukázali, že šifra funguje správně.

zdrojový text pro dešifrování textu:

```
1  retezec2="pxiwelei";
2  pomocna2=Characters[retezec2];
3  delka2=Length[pomocna2];
4  abeceda=CharacterRange["a","z"];
5  pomoc2={p, q, m, z, l, a, o, x, n, w, s, k, c, j,i, e, u, d, h, r,
   y, v, b, f, g, t};
6  desifrovany="";
7  For[pocitadlo=0,pocitadlo<delka2,++pocitadlo;
8    pozice2=Position[pomoc2,ToExpression[pomocna2[[pocitadlo]]]];
9
   desifrovany=StringJoin[desifrovany,ToString[abeceda[[pozice2[[1,1]]]
   ]]]
10   ]
11 desifrovany
```

Dešifrovaný text: ahojpepo

Popis zdrojového textu šifrování:

1. – 5. Jsou téměř identické jako u šifrování, jen jsem přidal index „2“. V proměnné „retezec2“ je vložený zašifrovaný řetězec z předcházejícího příkladu šifrování.
6. Inicializace proměnné „desifrovany“ na prázdnou hodnotu.
7. Začátek cyklu For.
8. – 9. Tělo cyklu provádí stejný proces jako na řádcích 7. – 11. z předchozího příkladu s tím rozdílem, že se prohodily proměnné „pomoc2“ a „abeceda“, takže se znaky převádí z abecedy šifrovaného textu do abecedy otevřeného textu (klasický abeceda „a“ – „z“).
10. Konec cyklu.

11. Výpis proměnné „desifrovany“ na výstupu. Podle dešifrovaného textu uvedeného následně po zdrojovém textu je vidět, že program pracuje správně, což si ostatně můžete vyzkoušet i v přiloženém souboru, jehož název je uvedený výše.

ZÁVĚR

Hlavním cílem této práce bylo naznačit potenciál aplikace webMathematica na několika příkladech metod šifrování klasické kryptografie. Když zkouším jednotlivé šifry na www stránkách, tak získávám stejné výsledky, jako kdybych prováděl šifrování na papíře, tudíž se mi tahle část práce povedla. Kromě šifrování jsem u většiny šifer uvedl i možnost dešifrování pro ověření správnosti výsledku, které získáme pomocí šifrování.

Možnosti rozšíření mých www stránek bych viděl v přeložení do anglického jazyka, aby jim rozumělo širší spektrum uživatelů.

Vývoj aplikací v prostředí webMathematica mě osobně hodně zaujal a myslím, že propojení programu Mathematica s webovými stránkami byl velmi dobrý nápad a může směřovat k velmi zajímavým projektům.

ZÁVĚR V ANGLIČTINĚ

Main aim those work was indicate potential of application webMathematica on several instances methods encryption classical cryptography. When I examine individual cipher on www pages, so extract same record, like if do encryption on paper, hence me traction volume of work answer. Except encryption near most cipher introduced also possibility decipherment for check rightness result that we obtain by the help of encryption.

Possibilities enlargement mine www pages I would saw in transfer to the English language, to state intelligibility wider spectrum users.

I am personally with application development in environment webMathematica much impressed and I think that the connection program Mathematica with web pages was very good idea and is able to aim at intriguing projects.

SEZNAM POUŽITÉ LITERATURY

Grošek O., Porubský Š.: Šifrovanie. Algoritmy, metódy a prax. Praha, Grada 1992

Janeček J.: Odhalená tajemství šifrovacích klíčů minulosti. Praha, Naše vojsko, 1994

Prokopová Z.: Základy informatiky. Vydavatelství FT, Zlín, 1998

Zelinka I.: Aplikovaná informatika. VUT Zlín, 1999

http://www.zivel.cz/toISO-8859-2.en/article.php?id=304

http://cs.wikipedia.org/

http://zelinka-mathematica.utb.cz:8080/webMathematica/student/

http://www.wolfram.com/products/webmathematica/whatis.html

http://www.karlin.mff.cuni.cz/~tuma/nciphers/nciphers1.pdf

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

AOT abeceda otevřeného textu.

AŠT abeceda šifrovaného textu.

SEZNAM OBRÁZKŮ

Obrázek 1: Viegenérova tabulka – metoda Viegenérova šifra 12
Obrázek 2: Viegenérova tabulka – metoda Autokláv 17
Obrázek 3. Histogram textu 22
Obrázek 4: Vzhled www stránky 30
Obrázek 5:Ovládání stránek 31
Obrázek 6: Výřez oblasti pro zadávání dat uživatelem 32

SEZNAM TABULEK

Tabulka 1: Viegenérova šifra 12
Tabulka 2: Jednoduchá transpozice 13
Tabulka 3: Dvojitá transpozice první heslo 14
Tabulka 4: Dvojitá transpozice druhé heslo 14
Tabulka 5: Jedna z Cardanových mřížek 15
Tabulka 6: Výsledná Cardanova mřížka pro náš příklad 15
Tabulka 7: Jednoduchá záměna – srovnaná abeceda 18
Tabulka 8: Jednoduchá záměna – reciproká abeceda 18
Tabulka 9: Jednoduchá záměna – rozházená abeceda 19
Tabulka 10: Tabulka 4x7 19
Tabulka 11: Tabulky 5x10 20
Tabulka 12: Playfair - rozložení matic 20
Tabulka 13: Metoda Playfair - příklad 21

SEZNAM PŘÍLOH

CD S OBSAHEM SPUSTITELNÝM NA POČÍTAČI S NAINSTALOVANÝ WEBMATHEMATICA SERVEREM.

Obsah CD:

4x7.jsp (3662 bytů)

5x10.jsp (3439 bytů)

Cardan.jsp (18586 bytů)

JednZam_RA.jsp (3109 bytů)

JednZam_ROA.jsp (3401 bytů)

JednZam_SA.jsp (3748 bytů)

autoklav.jsp (5391 bytů)

cesar.jsp (2516 bytů)

cesar2.jsp (2734 bytů)

grafika.css (554 bytů)

grafika.js (157 bytů)

histogram2.html (1148 bytů)

histogram2.jsp (2960 bytů)

index.html (1471 bytů)

obsah.html (2707 bytů)

playfair.jsp (5921 bytů)

transpozice.jsp (3696 bytů)

transpozice2.jsp (4401 bytů)

viegener.jsp (3361 bytů)

vymena.jsp (2312 bytů)

- **nb**

4x7.nb (7706 bytů)

5x10.nb (6401 bytů)

Dvojita transpozice.nb (11575 bytů)

Jednoducha_zam_RA.nb (5195 bytů)

Jednoducha_zam_ROA.nb (5139 bytů)

Jednoducha_zam_SA.nb (4833 bytů)

Viegener.nb (13781 bytů)

autoklav_moje.nb (18394 bytů)

caesar.nb (5757 bytů)

histogram.nb (159127 bytů)

jednoduchaTranspozice.nb (6008 bytů)

playfair.nb (13575 bytů)

sifra_autoklav.nb (8675 bytů)

vymena.nb (3161 bytů)

- **images**

download-icon.bmp (1954 bytů)

spikey.gif (1918 bytů)

- **dokumenty**

Bakalarska prace.pdf

Dokumentace k programum.pdf