

content of course «Basic of Infographics in Education» for teaching staff and trainers is proposed. The examples of practical tasks are discusses.

Keywords: infographics; visualization of information; data visualization, computer visualization; education; scientific and teaching staff training

ВІДОМОСТІ ПРО АВТОРІВ

Панченко Любов Феліксівна – доктор педагогічних наук, професор, професор кафедри фізико-технічних систем та інформатики ДЗ «Луганський національний університет імені Тараса Шевченка».

Коло наукових інтересів: інформаційно-комунікаційні технології в освіті, інформаційно-освітнє середовище університету, статистичні методи аналізу даних, комп'ютерний аналіз даних, дистанційне навчання.

Разорьонова Марина Валентинівна – кандидат педагогічних наук, доцент, докторант, ДЗ «Луганський національний університет імені Тараса Шевченка».

Коло наукових інтересів: управління освітою, інформаційно-комунікаційні технології в освіті, інноваційні методи викладання іноземних мов.

УДК 37.091.3:004.43

НАВЧАННЯ ШИФРУВАННЯ СИМВОЛЬНИХ ДАНИХ У СИСТЕМІ ПІДГОТОВКИ ПРІКЛАДНОГО ЛІНГВІСТА

Резіна Ольга

Кіровоградський державний педагогічний університет імені Володимира Винниченка

Анотація. У статті розглядаються особливості методики навчання шифрування символічних даних у процесі підготовки майбутніх фахівців з прикладної лінгвістики. Запропоновано можливий підхід до вивчення деяких шифрів підстановки та перестановки з реалізацією їх алгоритмів мовою програмування Python. Наведено програмні коди розглянутих методів шифрування та дешифрування.

Ключові слова: методика навчання, відкритий текст, шифротекст, шифрування, дешифрування, секретний ключ, шифр підстановки, шифр перестановки, мова програмування Python.

Постановка проблеми. Криптографія і шифрування використовувалися для секретного зв'язку на протязі тисячоліть. Історично необхідність секретної передачі даних у військовій галузі здійснила найбільший вплив на розвиток криптографії. В інформаційну епоху, що розпочалася у 80-х роках ХХ століття, стало актуальним питання безпеки комерційного та особистого спілкування. Передача відомостей через електронну пошту або службу World Wide Web є миттєвою, але вразливою до стороннього втручання. Використання мобільного зв'язку та електронної пошти, оплата товару у звичайному чи Інтернет-магазині через кредитну картку, здійснення банківських транзакцій в Інтернеті стає безпечним та конфіденційним за умови шифрування даних. Розуміння основ криптографії і шифрування дає змогу розробляти ефективні засоби захисту особистих та корпоративних даних.

У таких сучасних умовах професійна підготовка майбутніх фахівців з прикладної лінгвістики передбачає набуття ними компетентностей щодо розуміння принципів і методів шифрування даних, здатності до написання комп'ютерних програм, які реалізують алгоритми шифрування та дешифрування. Формування таких компетентностей можливе в процесі навчання теми «Шифрування символічних даних», включеної до програми однієї з дисциплін інформатичного циклу спеціальності «Прикладна лінгвістика».

Метою написання статті є ознайомлення з одним із методичних підходів вивчення вказаної теми.

Виклад основного матеріалу. Переходячи до розгляду питань, пов'язаних з основними ідеями шифрування відомостей, доцільно визначити базову термінологію, яка є такою:

- *відкритий текст (plaintext)* – вихідне (оригінальне) повідомлення;
- *шифротекст (ciphertext)* – зашифроване повідомлення;
- *шифрування (encipher, encrypt)* – процес перетворення відкритого тексту у шифротекст;
- *алгоритм шифрування (encryption algorithm)* – алгоритм перетворення відкритого тексту у шифротекст; вхідні дані: відкритий текст та секретний ключ;
- *дешифрування (decipher, decrypt)* – процес відтворення відкритого тексту з шифротексту;
- *алгоритм дешифрування (decryption algorithm)* – алгоритм перетворення шифротексту у відкритий текст; вхідні дані: шифротекст та секретний ключ;
- *секретний ключ (secret key)* – дані, використовувані при шифруванні, відомі тільки відправнику та отримувачу;
- *криптографія (cryptography)* – наука про принципи і методи шифрування;
- *криптоаналіз (cryptanalysis, codebreaking)* – наука про принципи і методи дешифрування без доступу до ключа;

– *криптологія (cryptology)* – наука, складовими якої є криптографія та криптоаналіз.

Розрізняють шифри симетричні та асиметричні. Якщо ключ є однаковим для шифрування та дешифрування, то шифр є симетричним. Якщо для шифрування та дешифрування використовуються різні ключі, то шифр є асиметричним. На початковому етапі навчання шифрування даних доцільно розглянути симетричні шифри, а саме:

1. класичні шифри підстановки (заміни):

- шифр Цезаря,
- шифр Гронсфельда,
- шифр Виженера,

суть яких полягає у заміні символів відкритого тексту іншими символами;

2. класичні шифри перестановки,

суть яких полягає у зміні місця розташування символів.

На рис. 1 представлена схема шифрування та дешифрування даних, яка призначена для кращого розуміння студентами сутності вказаних процесів.

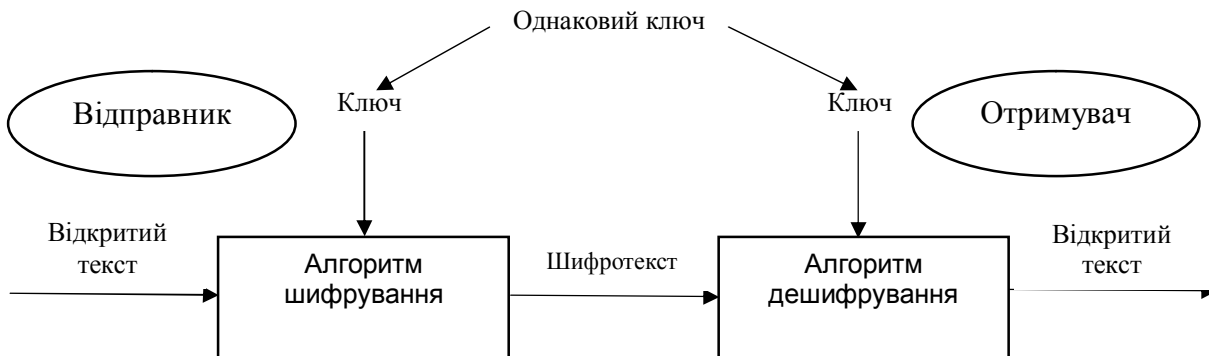


Рис. 1. Схема шифрування та дешифрування даних

При організації навчання теми однією з найважливіших проблем є проблема вибору програмного забезпечення, що надає можливість імплементації алгоритмів шифрування та дешифрування даних комп'ютерними програмами. У статті пропонується розглянути мову Python як засіб створення програм.

Python – це інтерпретована, об'єктно-орієнтована мова програмування високого рівня з динамічною семантикою. Її конструкція включає структури даних комбіновані з динамічними типами та динамічним зв'язуванням, що робить цю мову привабливою для швидкого написання програм. Python підтримує використання модулів, завдяки чому стає можливим повторне використання коду [5]. Крім того, стандартна бібліотека мови Python містить модулі, зокрема String, які надають потужні засоби опрацювання текстових даних [4].

Розпочати вивчення теми можна з розгляду шифрування методом Цезаря. Шифр Цезаря – це моноалфавітний шифр простої заміни, один з найдавніших, найпростіших і найвідоміших методів шифрування. При шифруванні кожен символ замінюється іншим, віддаленим від нього в алфавіті на фіксоване число позицій. Цей метод шифрування є найпростішим для імплементації його комп'ютерною програмою, висвітлений у багатьох навчальних ресурсах [2; 3]. Розглянемо детально наступний метод шифрування.

Відкритий текст	V	U	L		S	H	E	V	C	H	E	N	K	A
Порядковий номер букви відкритого тексту	22	21	12		19	8	5	22	3	8	5	14	11	1
Ключ	1	4	2	5	1	4	2	5	1	4	2	5	1	4
Порядковий номер букви зашифрованого тексту	23	25	14		20	12	7	27-26=1	4	12	7	19	12	5
Шифротекст	W	Y	N		T	L	G	A	D	L	G	S	L	E

Рис. 2. Схема шифрування методом Гронсфельда

Шифр Гронсфельда – це поліалфавітний шифр складної заміни, що є модифікацією шифру Цезаря числовим ключем. Принцип дії цього методу полягає в тому, що під буквами вихідного повідомлення записують цифри числового ключа. Якщо ключ коротше повідомлення, то його запис циклічно повторюється. Шифротекст отримують способом, подібним на шифр Цезаря, але відраховують за алфавітом

не фіксоване число позицій, а обирають ту букву, що зміщена за алфавітом на відповідну цифру ключа. Для кращого розуміння суті методу можна запропонувати студентам схему, представлену на рис. 2.

Далі необхідно зазначити, що комп'ютерною програмою будуть опрацьовуватися не порядкові номери букв, а їхні коди відповідно до таблиці Unicode. І тоді наведена схема шифрування набуде вигляду, як показано на рис. 3.

Відкритий текст	V	U	L	(s)	S	H	E	V	C	H	E	N	K	A
Код символів відкритого тексту	86	85	76	32	83	72	69	86	67	72	69	78	75	65
Ключ	1	4	2	5	1	4	2	5	1	4	2	5	1	4
Код символів зашифрованого тексту	87	89	78	32	84	76	71	91-26=65	68	76	71	83	76	69
Шифротекст	W	Y	N	(s)	T	L	G	A	D	L	G	S	L	E

Рис. 3. Схема шифрування методом Гронсфельда з урахуванням кодів символів

Символами (s) позначено пробіл, який має код 32.

Розглянемо один із можливих варіантів реалізації шифру Гронсфельда мовою Python. Особливістю цієї реалізації є те, що програмою розрізняються великі та малі літери, шифруються цифри, решта символів залишається без змін.

Нижче подано програмний код, який виконує шифрування повідомлення:

```

1     import string
2
3     plaintext=input("Enter the message-> ")
4
5     key=input("Enter the key -> ")
6     key=[int(v)for v in key]
7
8     rep=len(plaintext)//len(key)
9     residue=len(plaintext)%len(key)
10    key=key*rep+key[:residue]
11
12    ciphertext=""
13
14    for i, letter in enumerate(plaintext):
15
16        if letter in string.ascii_uppercase:
17            ciphercode=ord(letter)+key[i]
18            if ciphercode>90:
19                ciphercode-=26
20        elif letter in string.ascii_lowercase:
21            ciphercode=ord(letter)+key[i]
22            if ciphercode>122:
23                ciphercode-=26
24        elif letter in string.digits:
25            ciphercode=ord(letter)+key[i]
26            if ciphercode>57:
27                ciphercode-=10
28        else:
29            ciphercode=ord(letter)
30
31        cipherletter=chr(ciphercode)
32        ciphertext+=cipherletter
33
34    print("ciphertext=", ciphertext)

```

Розглянемо цей програмний код детально. Спочатку імпортується модуль `string`, що містить константи, функції та методи опрацювання рядків. За допомогою функції `input()` вводиться текст для

шифрування та секретний ключ – число будь-якої розрядності. У рядку 6 змінна рядкового типу `key`, призначена для зберігання секретного ключа, перетворюється на список цілих чисел, що надає змогу звертатися до *i*-го елемента ключа та додавати його до коду *i*-го символу. Ключ необхідно циклічно продовжити до довжини введеного тексту. Ці дії виконуються за допомогою операторів, записаних у рядках 8-10 програми: до змінної `repeat` записується кількість цілих повторень ключа; до змінної `residue` – кількість символів, до яких необхідно дописати частину ключа (остачу від ділення довжини тексту на довжину рядка); змінна `key` перетворюється таким чином, щоб ключ був підставлений під текст повністю.

Далі виконується власне шифрування тексту (рядки 12-32 програми). Зашифрований текст формується у змінній `ciphertext`, якій спочатку надається значення порожнього рядка. Усі символи відкритого тексту послідовно аналізуються за допомогою циклу `for` та функції `enumerate()`, яка індексує символи відкритого тексту (перетворює тип даних у змінній `plaintext` з рядкового на список). Для реалізації шифрування застосовується умовна конструкція `if-elif-else`, функції `ord()` (повертає код символу) та `chr()` (навпаки, повертає літеру відповідно до коду). Оператор `if-elif-else` перевіряє, чи належить символ до верхнього регістру (використовується константа `ascii_uppercase` модуля `string`), нижнього регістру (використовується константа `ascii_lowercase` модуля `string`), або є цифрою (використовується константа `ascii_digits` модуля `string`); і відповідно до результату шифрує його.

Код зашифрованого символу отримується додаванням поточної цифри ключа до коду символу, що шифрується, і заноситься до змінної `ciphercode` (рядки 17, 21, 25). Якщо символ відкритого тексту знаходиться у верхньому регістрі, то необхідно врахувати, що відповідно до таблиці Unicode, літери латиниці верхнього регістру (A-Z) займають позиції від 65 до 90. Якщо значення змінної `ciphercode` більше за 90, то необхідно повернутися на початок алфавіту (рядок 19 програми). За аналогічним принципом шифруються символи нижнього регістру (a-z, 90-122 у таблиці Unicode) та цифри (0-9, позиції 48-57). Якщо ж символ не належить до верхнього, нижнього регістрів і не є цифрою (наприклад, знаки пунктуації), то він додається в першопочатковому вигляді. Функцією `print()` зашифрований текст виводиться на екран (рядок 34).

Тепер розглянемо програмний код, який виконує дешифрування повідомлення:

```

1     import string
2
3     ciphertext=input("Enter the cifered message-> ")
4
5     key=input("Enter the key -> ")
6     key=[int(v)for v in key]
7
8     repeat=len(ciphertext)//len(key)
9     residue=len(ciphertext)%len(key)
10    key=key*repeat+key[:residue]
11
12    plaintext = ""
13
14    for i, letter in enumerate(ciphertext):
15
16        if letter in string.ascii_uppercase:
17            deciphercode=ord(letter)-key[i]
18            if deciphercode<65:
19                deciphercode+=26
20        elif letter in string.ascii_lowercase:
21            deciphercode=ord(letter)-key[i]
22            if deciphercode<97:
23                deciphercode+=26
24        elif letter in string.digits:
25            deciphercode=ord(letter)-key[i]
26            if deciphercode<48:
27                deciphercode+=10
28        else:
29            deciphercode=ord(letter)
30
31        decipherletter=chr(deciphercode)
32        plaintext+=decipherletter
33
```

```
34 print("plaintext=", plaintext)
```

Рядки 1-10 програми аналогічні відповідним рядкам програми шифрування. Дешифрований текст зберігається у змінній `plaintext`, і на початку дешифрування їй надається значення порожнього рядка (рядок 12 програми). Але тепер значення елемента ключа не додається, а віднімається (рядки 17, 21, 25), і аналізуються не верхні межі діапазону кодів групи символів, а нижні, відбувається повернення на кінець алфавіту (рядки 18-19, 22-23, 26-27).

Для повідомлення *Address: 1 vul. Shevchenka, Kropyvnytskyi, Ukraine, 25006* та ключа *1425* результати роботи програм будуть такими, як показано на рис. 4:

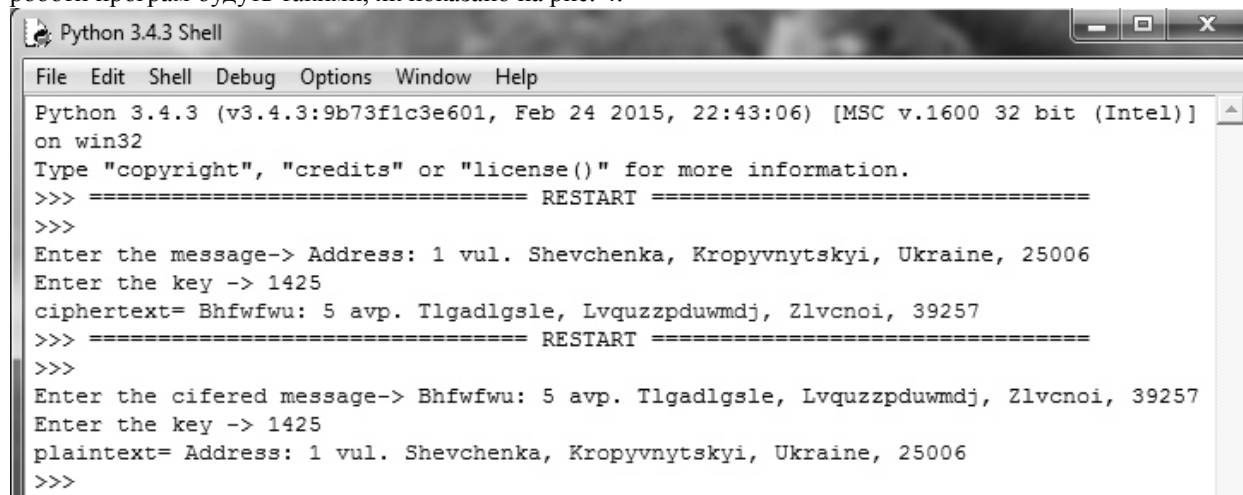


Рис. 4. Результати роботи програм шифрування та дешифрування методом Гронсфєльда

Наступним кроком вивчення теми є розгляд шифрування методом Віженера. Шифр Віженера – це поліалфавітний шифр підстановки, призначений для шифрування буквеного тексту. На відміну від шифрів Цезаря та Гронсфєльда, у шифрі Віженера використовується не числовий ключ, а буквений, наприклад, деяке слово. Це ключове слово розбивається на окремі літери. Якщо використати в якості ключа слово «CLIP», то першим підключем є літера С, другим – літера L, третім – I і четвертим – P. Перший підключ використовується для шифрування першої букви відкритого тексту, другий – для другої і так далі. При шифруванні п'ятої букви відкритого тексту необхідно повернутися до використання першого підключа. Для демонстрації цього процесу доцільно використати схему шифрування, представлену на рис. 5.

Відкритий текст	V	U	L	(s)	S	Н	Е	V	С	Н	Е	N	К	A
Код букви відкритого тексту	86	85	76	32	83	72	69	86	67	72	69	78	75	65
Ключ	C (67)	L (76)	I (73)	P (80)	C (67)	L (76)	I (73)	P (80)	C (67)	L (76)	I (73)	P (80)	C (67)	L (76)
Код букви зашифрованого тексту	88	70	84	32	85	83	77	75	69	83	77	67	77	76
Шифротекст	X	F	T	(s)	U	S	M	К	Е	S	M	С	M	L

Рис. 5. Схема шифрування методом Віженера з урахуванням кодів символів

Код букви зашифрованого тексту отримується за формулою: код букви відкритого тексту + код ключа - 64 (код першої літери A=65). Якщо отримане значення більше 90, то від нього треба відняти число 26.

Ключове слово необов'язково повинно бути словниковим. Ключем може бути довільна послідовність символів, наприклад, «DWNC». І це є кращий варіант, оскільки використання такого ключа утруднює злам шифру.

Програмний код одного з можливих варіантів реалізації шифру Віженера мовою Python наведено нижче:

```
1 import string
2
```

```

3     plaintext=input("Enter the message -> ")
4     plaintext = plaintext.upper()
5
6     key=input("Enter the key_word -> ")
7     key=key.upper()
8
9     repeat=len(plaintext)//len(key)
10    residue=len(plaintext)%len(key)
11    key=key*repeat+key[:residue]
12
13    ciphertext=""
14
15    for i, letter in enumerate(plaintext):
16        if letter in string.ascii_uppercase:
17            ciphercode=ord(letter)+(ord(key[i])-64)
18            if ciphercode>90:
19                ciphercode-=26
20        else:
21            ciphercode=ord(letter)
22
23        cipherletter=chr(ciphercode)
24        ciphertext+=cipherletter
25
26    print("ciphertext=",ciphertext)

```

Шифрування методом Віженера відбувається у верхньому регістрі, тому відкритий текст і слово-ключ автоматично переводяться у верхній регістр за допомогою методу `upper()` (рядки 4 та 7 програми). Потім відбувається реалізація шифрування (рядки 13-24) за тим самим принципом, що і розглянутий вище метод Гронсфельда. Але тепер програма шифрує лише символи верхнього регістру, а всі інші символи ігнорує. У рядку 26 відбувається виведення зашифрованого тексту на екран.

Відповідна програма дешифрування повідомлення є такою:

```

1     import string
2
3     ciphertext=input("Enter the cifered message -> ")
4     ciphertext = ciphertext.upper()
5
6     key=input("Enter the key_word -> ")
7     key=key.upper()
8
9     repeat=len(ciphertext)//len(key)
10    residue=len(ciphertext)%len(key)
11    key=key*repeat+key[:residue]
12
13    plaintext=""
14
15    for i, letter in enumerate(ciphertext):
16        if letter in string.ascii_uppercase:
17            deciphercode=ord(letter)-(ord(key[i])-64)
18            if deciphercode<65:
19                deciphercode+=26
20        else:
21            deciphercode=ord(letter)
22
23        decipherletter=chr(deciphercode)
24        plaintext+=decipherletter
25
26    print("plaintext=", plaintext)

```

Для повідомлення *Kirovohrad Volodymyr Vynnychenko State Pedagogical University* та ключа *DWNC* результати роботи програм будуть такими, як показано на рис. 6:

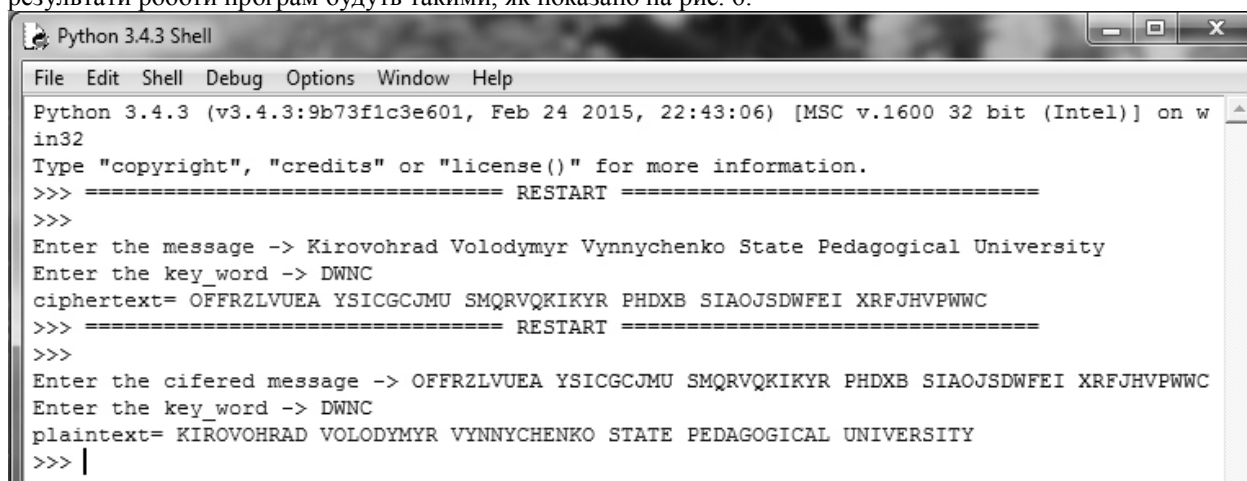


Рис. 6. Результати роботи програм шифрування та дешифрування методом Віженера

Окрім шифрів підстановки (заміни), які систематично замінюють літери або групи літер іншими літерами або групами літер, основними типами класичних шифрів також є шифри перестановки, які змінюють порядок літер у повідомленні. Принцип шифрів перестановки полягає в тому, що окремі символи або певні групи символів переставляються місцями у відкритому тексті за певними правилами.

Під час проведення мотиваційного етапу вивчення шифрів перестановки необхідно зазначити, що шифри підстановки не є безпечними. Наприклад, для зламу шифру Цезаря не потрібно багато часу та ресурсів комп'ютера, щоб перебрати всі двадцять шість можливих ключів. Тобто шифри такого типу можуть бути відкриті без надмірних зусиль, «в лоб». На відміну від шифрів підстановки, шифри перестановки мають більше можливих ключів та є складнішими для зламу.

Розглянемо методику вивчення принципів шифрування методом перестановки. Для прикладу використаємо повідомлення «Realization of a transposition cipher». Це повідомлення має 38 символів, враховуючи пробіли та пунктуацію. У якості ключа виберемо число 6. Спочатку створюється рядок з такою кількістю клітинок, що еквівалентна ключу, та у кожен клітинку вписується один символ повідомлення.

R	e	a	l	i	z
---	---	---	---	---	---

Так як загальна кількість символів у повідомлення 38, а клітинок 6, то до першого рядка потрібно додати таку кількість рядків, щоб вписати ціле повідомлення (рис. 7).

Клітинки, що залишились пустими, у процесі шифрування ігноруються. Шифротекст зчитується, починаючи з першого стовпця, рухаючись вниз. Коли перший стовпець закінчується, відбувається перехід на другий, потім на третій і т.д. [3]. У результаті виходить нечитабельний шифротекст «Raors rtfaic.aitloasipin pohz tone».

У програмі вихідний текст міститься у змінній рядкового типу, а це означає, що кожен символ у повідомленні має індекс (індексація починається з 0). Зі схеми на рис. 7 видно, що елементи першого стовпчика мають індекси 0,6,12,18,24,30,36, другого – 1,7,13,19,25,31,37 і т.д. Тобто, елементи n-го стовпчика матимуть значення 0+n, 6+n, 12+n, 18+n, 24+n, 30+n, 36+n. Використаємо це при написанні програми шифрування [3], код якої наведено нижче.

```

1     def main():
2         myMessage = input("Enter the message -> ")
3         myKey = int(input("Enter the key -> "))
4
5         ciphertext = encryptMessage(myKey, myMessage)
6
7         ciphertext = ciphertext + '|'
8         print(ciphertext)
9
10    def encryptMessage(key, plaintext):
11        ciphertext = [''] * key
12
13        for col in range(key):
14            pointer = col
15

```

```

16         while pointer < len(plaintext):
17             ciphertext[col] += plaintext[pointer]
18             pointer += key
19
20         return ''.join(ciphertext)
21
22     if __name__ == '__main__':
23         main()
    
```

На цьому етапі навчання можна пропонувати студентам створювати власні функції, мотивуючи це тим, що програму, розділену на змістові частини простіше розробляти, розуміти та вдосконалювати. Наведена програма містить основну функцію `main()`, яка є початковою точкою виконання програми. Операторами функції `main()` є: ініціалізація відкритого тексту, ініціалізація ключа, виклик функції `encryptMessage()`, додавання до зашифрованого тексту знака | («pipe»), який є маркером пробілів наприкінці зашифрованого тексту), виведення зашифрованого тексту на екран (рядки 2-8 програми). У рядку 5 функцію `main()` передає управління функції `encryptMessage()`, призначенням якої є шифрування тексту. Ключ і відкритий текст передаються як параметри. Фактичним параметрам `myKey` та `myMessage` ставляться у відповідність формальні параметри `key` та `plaintext`.

	1	2	3	4	5	6
R	e	a	l	i	z	
0	1	2	3	4	5	
a	t	i	o	n	(s)	
6	7	8	9	10	11	
o	f	(s)	a	(s)	t	
12	13	14	15	16	17	
r	a	n	s	p	o	
18	19	20	21	22	23	
s	i	t	i	o	n	
24	25	26	27	28	29	
(s)	c	i	p	h	e	
30	31	32	33	34	35	
r	.					
36	37					

Рис. 7. Схема шифрування методом перестановки

У рядку 11 використовується такий тип даних як список. Список, елементами якого є 6 рядків, що містять символи кожного стовпця нашого прикладу, матиме вигляд:

```
['Raors r', 'etfaic.', 'ai nti', 'loasip', 'in poh', 'z tone']
```

Оператором рядка 11 програми змінній `ciphertext` надається значення списку, елементами якого є порожні рядки, кількість цих рядків дорівнює значенню ключа. Кожний рядок списку репрезентує одну колонку. Таким чином, `ciphertext[0]` представляє першу колонку, `ciphertext[1]` – другу і т.д.

Наступний крок – додавання тексту до кожного рядка шифротексту. Цикл `for` (рядок 13) перебирає значення кожного стовпця, змінна `col` має цілочисловий тип і використовується як індекс рядка шифротекста. А змінна `pointer` використовується як індекс символу рядка шифротекста. Поки значення `pointer` менше, ніж довжина повідомлення, на кінець рядка шифротексту додається символ `plaintext[pointer]`, і значення змінної `pointer` збільшується на величину ключа (рядки 16-18 програми). Під час першої ітерації це буде `plaintext[0]`, другої – `plaintext[6]`, третьої – `plaintext[12]`, і т.д., сьомої – `plaintext[36]`. Кожна з цих змінних додається у кінець змінної `ciphertext[col]`.

Кожний стовпець є сформованим в окремий елемент списку: `['Raors r', 'etfaic.', 'ai nti', 'loasip', 'in poh', 'z tone']`. Для їх об'єднання потрібно використати метод `join()`. Оператор `return` повертає зашифрований текст функції `main()` (рядок 20).

Умова у рядку 22 дає змогу Python визначити, яку функцію запускати в тому випадку, якщо файл запущений як основний, а не як імпортований в інший скрипт.

Вивчення принципів дешифрування доцільно почати з припущення, що отримувачу надійшов шифротекст «Raors retfaic.ai ntiloasipin pohz tone» і значення ключа – 6. Спочатку необхідно визначити розмір дешифрувальної таблиці. Для визначення кількості стовпчиків потрібно розділити загальну кількість символів повідомлення (38) на ключ (6). Отримане число округлити до найближчого більшого цілого. Тобто, кількість стовпчиків дешифрувальної таблиці – 7. Кількість рядків дорівнює значенню ключа. У результаті отримується таблиця розміром 7x6. Також є необхідним визначити, яку кількість клітинок потрібно ігнорувати у процесі дешифрування. Для цього від загальної кількості клітинок таблиці (42) необхідно відняти кількість символів у повідомленні (38). У результаті отримується 4 – кількість клітинок у стовпці справа, які ігноруватимуться. Таблиця заповнюється рядками справа наліво. Відкритий текст зчитується стовпцями зліва направо і отримується повідомлення «Realization of a transposition cipher.» (рис. 8) [3].

R	a	o	r	s	(s)	r
e	t	f	a	<u>i</u>	c	.
a	<u>i</u>	(s)	n	t	<u>i</u>	
l	o	a	s	<u>i</u>	p	
<u>i</u>	n	(s)	p	o	h	
z	(s)	t	o	n	e	

Рис. 8. Схема дешифрування методом перестановки

Програмний код, який виконує процес дешифрування [3], є таким:

```

1     import math
2
3     def main():
4         myMessage = input("Enter the ciphered message -> ")
5         myKey = int(input("Enter the key -> "))
6
7         plaintext = decryptMessage(myKey, myMessage)
8
9         print(plaintext + '|')
10
11        def decryptMessage(key, message):
12            numColumns = math.ceil(len(message) / key)
13            numRows = key
14            numShadedBoxes = (numColumns * numRows) - len(message)
15
16            plaintext = [''] * numColumns
17
18            col = 0
19            row = 0
20
21            for symbol in message:
22                plaintext[col] += symbol
23                col += 1
24
25                if (col == numColumns) or (col == numColumns - 1 and row
>= numRows - numShadedBoxes):
26                    col = 0
27                    row += 1
28
29            return ''.join(plaintext)
30
31
32        if __name__ == '__main__':
33            main()

```

У програмний код дешифрування імпортується модуль `math`, який дає змогу використовувати математичні функції. Безпосередньо дешифрування виконує функція `decryptMessage()`, розглянемо її оператори. У рядках 12-14 програми визначається кількість стовпців і рядків таблиці дешифрування, а також кількість клітинок, що будуть ігноруватися. Функція `math.ceil()` модуля `math` округлює число до найближчого більшого цілого. Змінній `plaintext` надається значення списку рядків, кожен з яких репрезентує стовпець у дешифрувальній таблиці (рядок 16). Значення змінних `col` і `row` визначають, в якій клітинці таблиці знаходиться символ, що буде опрацьований. Вони модифікуються в тілі оператора циклу `for`, в якому перебираються символи шифротексту і формується дешифрований текст. Оператор розгалуження трактується таким чином: якщо опрацьовані всі стовпці або досягнуті клітинки, що ігноруються, то перейти до першого стовпця і наступного рядка (рядки 25-27 програми). Результатом роботи циклу є список рядків:

[‘Realiz’, ‘ation ‘, ‘of a t’, ‘ranspo’, ‘sition’, ‘ ciphe’, ‘r.’]. Остаточний результат формується об’єднанням елементів списку за допомогою методу `join()` і передається в основну програму.

Результати роботи програм показані на рис. 9:

```

Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)] on w
in32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
Enter the message -> Realization of a transposition cipher.
Enter the key -> 6
ciphertext= Raors retfaic.ai ntiloasipin pohz tone|
>>> ===== RESTART =====
>>>
Enter the ciphered message -> Raors retfaic.ai ntiloasipin pohz tone
Enter the key -> 6
plaintext= Realization of a transposition cipher.|
>>> |
    
```

Рис. 9. Результати роботи програм шифрування та дешифрування методом перестановки

Висновки. Запропонований у статті підхід до навчання студентів шифрування символічних даних використовується під час вивчення дисципліни «Програмування та обчислення» на факультеті іноземних мов (спеціальність «Прикладна лінгвістика») КДПУ імені Володимира Винниченка. Ця тема є світоглядною і водночас практичнозначущою. Її вивчення дає змогу викладачеві ознайомити студентів з важливими принципами шифрування та дешифрування даних, сформувані знання про алгоритми реалізації цих методів, удосконалити навички написання комп’ютерних програм та використання різних типів даних.

Перспективою подальших досліджень може бути розробка методики навчання інших типів шифрів, наприклад, мультиплікативних та афінних і використання текстових файлів у процесі передачі зашифрованих даних комп’ютерною мережею.

БІБЛЮГРАФІЯ

1. McDonald, N. Past, Present, and Future Methods of Cryptography and Data Encryption. A Research Review [Electronic resource] / McDonald, N. – Mode of access: <http://www.eng.utah.edu/~nmcdonal/Tutorials/EncryptionResearchReview.pdf>
2. Stallings, W. Cryptography and Network Security Principles and Practices / Stallings, W. – 4th Edition. – Prentice Hall, 2005. – 592 p. – Mode of access: http://www.inf.ufsc.br/~bosco.sobral/ensino/ine5680/material-cripto-seg/2014-1/Stallings/Stallings_Cryptography_and_Network_Security.pdf
3. Sweigart, A. Hacking Secret Ciphers with Python [Electronic resource] / Sweigart, A. – Mode of access: <https://inventwithpython.com/hackingciphers.pdf>
4. Text Processing Services [Electronic resource] / The Python Standard Library – Mode of access: <https://docs.python.org/3.4/library/text.html>
5. What is Python? Executive Summary [Electronic resource] / Python’s standard documentation. – Mode of access: <https://www.python.org/doc/essays/blurb/>

ОБУЧЕНИЕ ШИФРОВАНИЮ СИМВОЛЬНЫХ ДАННЫХ В СИСТЕМЕ ПОДГОТОВКИ ПРИКЛАДНОГО ЛИНГВИСТА

Резина Ольга

В статье рассматриваются особенности методики обучения шифрованию символьных данных в процессе подготовки будущих специалистов по прикладной лингвистике. Предложен возможный подход к изучению

некоторых шифров подстановки и перестановки с реализацией их алгоритмов на языке программирования Python. Приведены программные коды рассмотренных методов шифрования и дешифрования.

Ключевые слова: методика обучения, открытый текст, шифротекст, шифрование, дешифрование, секретный ключ, шифр подстановки, шифр перестановки, язык программирования Python.

TEACHING CHARACTER DATA ENCRPTION IN THE SYSTEM OF AN APPLIED LINGUIST TRAINING

Riezina Olga

The article highlights the peculiarities of teaching methods to encrypt character data in the process of training future specialists in Applied Linguistics. The topicality of the theme has been defined in consideration of the importance of the commercial and personal communication security question. The basic terminology has been outlined. A possible approach to studying some substitution and transposition ciphers and applying their algorithms to a computer program has been suggested. The Python programming language has been defined as an effective means of processing character data. Program codes of the discussed methods to encrypt and decrypt data have been revealed.

Keywords: teaching methods, plaintext, ciphertext, encrypt, decrypt, secret key, substitution cipher, transposition cipher, Python programming language.

ВІДОМОСТІ ПРО АВТОРА

Резіна Ольга Василівна – кандидат педагогічних наук, доцент, доцент кафедри інформатики Кіровоградського державного педагогічного університету імені Володимира Винниченка.

Коло наукових інтересів: дослідження можливостей використання ресурсів мережі Інтернет у науковій, дослідницькій та навчальній діяльності; технології опрацювання текстових даних; технології дистанційного навчання.

УДК 53(07) 535

ПРОБЛЕМИ НАВЧАННЯ НЕРІВНОВАЖНИХ ПРОЦЕСІВ

Садовий Микола

Кіровоградський державний педагогічний університет імені Володимира Винниченка

***Анотація.** Стаття присвячена важливій проблемі дослідження розвитку нерівноважних, відкритих, неізольованих систем. Проаналізовано рівноважні процеси, чотири начала термодинаміки. Визначені межі їх застосування. У статті встановлено, що упродовж кількох останніх десятиліть фізики, хіміки і біологи зуміли наблизитись до розуміння процесів формування структур у відкритих системах, тобто системах, які обмінюються речовиною й енергією з оточуючим середовищем. Відповідь на питання про причини та загальні закономірності самоорганізації міститься в термодинаміці незворотних процесів, або ж, як її прийнято називати, нерівноважній термодинаміці. Величезною заслугою нерівноважної термодинаміки є усвідомлення того факту, що нерівноважність може бути причиною порядку. У статті приведено порівняння рівноважних та нерівноважних систем, механізм утворення локальних лінійних нерівноважних систем та перспективи їх розвитку.*

***Ключові слова:** нерівноважний, рівноважний, термодинаміка, відкрита система, історія розвитку досліджень.*

Постановка проблеми. Із великих наукових досягнень двох минулих століть своєю особливою специфічною направленістю виділяється еволюційна теорія Ч. Дарвіна і феноменологічна термодинаміка. Перша обґрунтувала розвиток живої матерії від нижчих форм до вищих, тобто ускладнення організації в процесі еволюції. Це передбачає дезорганізацію або руйнування наперед заданої структури в ізольованій системі під час її еволюції до стану рівноваги. Тобто еволюційна ідея ХІХ ст. виникла у вигляді двох протилежних форм – у вигляді теорії «створення структури» Ч. Дарвіна і теорії «руйнування структур», якою є класична термодинаміка. Як наслідок виникла проблема: як доповнити класичну термодинаміку теорією «створення структури». Така проблема належить до числа фундаментальних у природничих науках, а пояснення виникнення структур є однією з найважливіших цілей наукового пізнання.

Упродовж кількох останніх десятиліть фізики, хіміки і біологи зуміли наблизитись до розуміння процесів формування структур у відкритих системах, тобто системах, які обмінюються речовиною й енергією з оточуючим середовищем. Відповідь на питання про причини та загальні закономірності самоорганізації міститься в термодинаміці незворотних процесів, або ж, як її прийнято називати, нерівноважній термодинаміці. Величезною заслугою нерівноважної термодинаміки є усвідомлення того факту, що нерівноважність може бути причиною порядку. Таким чином, уперше вдалось «перекинути» місток і встановити зв'язок між протилежними властивостями. Виявилось, що незворотні процеси у відкритих системах можуть приводити до виникнення нового типу динамічних станів матерії – дисипативних (нерівноважних) систем, які самоорганізуються. При цьому джерелом нововведень у системі, тобто джерелом структурної еволюції, є флуктуації, які «каталізують» механізм нестійкості, що у свою чергу, приводить до формування нової просторо-часової структури.