МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ»
КАФЕДРА ПРИКЛАДНОЇ МАТЕМАТИКИ ТА ІНФОРМАТИКИ

МЕТОДИЧНІ ВКАЗІВКИ

ДО ВИКОНАННЯ ПРАКТИЧНИХ І РОЗРАХУНКОВИХ РОБІТ
З ДИСЦИПЛІНИ «ТЕХНОЛОГІЇ ЗАХИСТУ ІНФОРМАЦІЇ»

для студентів спеціальності
122 Комп’ютерні науки
усіх форм навчання

Луцьк - 2022

УДК 004.056.5(072)
М 54

Методичні вказівки до виконання практичних і розрахункових робіт з дисципліни «Технології захисту інформації» для студентів спеціальності 122 Комп'ютерні науки всіх форм навчання [Електронний ресурс] / укл. Т. О. Черняк, Н. О. Маслова, О.М. Любименко. – Луцьк : ДонНТУ, 2022. – 41 с.

Наведено завдання до виконання практичних робіт з дисципліни «Технології захисту інформації», у стиснутій формі надано теоретичні відомості. Сформульовано завдання й описано порядок виконання робіт. Надано вимоги до оформлення звітів, перелік літературних джерел. Завдання орієнтовані на закріплення знань, отриманих студентами при вивченні лекційного матеріалу та придбання навичок захисту інформації за темою «Основи шифрування та криптографічні перетворення».

Укладачі: Н.О. Маслова, доцент кафедри ПМІ, к.т.н., доцент
Т. О. Черняк, асистент кафедри ПМІ
О.М. Любименко, доцент кафедри

Рецензент: доц., к.т.н., доцент каф. ЕТ Штепа О.А.

Відповідальний за випуск:
Н.О. Маслова, в.о. завідувача кафедри прикладної математики і інформатики

Затверджено на засіданні навчально-методичного відділу ДонНТУ,
протокол № 3 від «29» грудня 2022 р.

Ухвалено на засіданні кафедри прикладної математики та інформатики,
протокол № 12 від «21» грудня 2022 р.

ЗМІСТ

ВСТУП

Метою навчальної дисципліни «Технології захисту інформації» є надання завдань для набуття практичних навичок щодо застосування технологій захисту інформації, та відпрацьовування методів підвищення безпеки програмних реалізацій з застосуванням базових алгоритмами кодування інформації. Слід звернути увагу, що методичні вказівки деталізують одну з найважливіших тем курсу - «Основи шифрування та криптографічні перетворення».

З урахуванням того, що методичні вказівки призначені для студентів старшого курсу бакалаврату, обрання мови програмування, або опис послідовності дій не є основою вимогою. Головна задача – застосування досконально відомого, ефективного або зручного сучасного інструментального засобу, створення програмних версій елементів захисту інформації й демонстрація їх працездатності та застосовності.

Методичні вказівки включають пояснення та приклади до виконання практичних робіт, що в цілому повинно сприяти формуванню у студентів здатностей із захисту даних, застосуванню криптографічних методів та алгоритмів шифрування.

Всі практичні роботи мають сформульовану мету роботи, опис порядку виконання, вимоги до змісту звіту та стислі теоретичні відомості та питання до самоконтролю.

В методичниі вказівки включено завдання та приклад виконання індивідуальної науково-дослідної роботи студента, яка є логічним продовженням тематики практичних робіт й акумулює отримані навички.

Розв’язання завдань практичних робіт, включених в методичні вказівки може бути корисним й в якості практичнх завдань для самостійного опанування студентами різних спеціальностей галузі 12 – «Інформаційні технології» в якості примірника базових розробок для включення в кріптографічну бібліотеку студента, в якій певні блоки реалізують криптографічні алгоритми на рівні бітового представлення даних.

ПРАКТИЧНА РОБОТА 1. ФОРМУВАННЯ ТАБЛИЦЬ ПРОСТИХ ЧИСЕЛ

Мета роботи: сформувати критерії та виконати оцінку базових алгоритмів формування таблиць простих чисел.

Теоретичний матеріал до виконання практичної роботи №1

Алгоритми пошуку простих чисел

Простий перебор

Це найбільш простий алгоритм, який реалізує практично наївний підхід − просто ділити на всі числа підряд до кореня з N у послідовності з B елементів (межа B – наперед задана).

На початку ми можемо мати або не мати у своєму розпорядженні досить велику таблицю простих чисел:

p[1] = 2,
p[2] = 3,
... ,
p[k],

де k > 3, масив t := [6,4,2,4,2,4,6,2], та індекс j, такий що якщо p[k] mod 30 дорівнює 1,7,11,13,17,19,23 або 29, то j := 0,1,2,3,4,5,6 або 7 відповідно.

Оберемо деяку верхню грань B, таку що B >= p[k].

Отримавши на вході додатне ціле число N, цей алгоритм намагається розкласти його на множники, і, якщо це не вдається, то N − число без простих дільників, менших або дорівнюючих B.

Опис алгоритму складається з 5 етапів, котрі мають умовни назви. На першому виконується ініціалізація й це, відповідно етап ініціалізації. Другий – етап – пошук наступного простого. Третій - простий поділ. Четвертий – фіксація знайденого простого числа. П’ятий – пошук наступного простого дільника.

1. [Ініціалізація]
   Якщо N <= 5, вивести розкладання 1 = 1, 2 = 2, 3 = 3, 4 = 2^2, 5 = 5 в залежності від N та вийти. Інакше, i := −1, m := 0, L := [$N^{1/2}$]
2. [Наступне просте]
   Нехай m := m + 1. Якщо m > k, то i : = j − 1 та йти до кроку 5, інакше d=p[m].
3. [Простий поділ]
   Нехай r : = N mod d. Якщо r = 0, то вивести d як нетривіальний дільник N та вийти (або N := N / d, L : = [$N^{1/2}$] і повторити крок 3, якщо хочемо продовжити знаходження дільників N).
4. Якщо d >= L, то у випадку N > 1 вивести повідомлення, що частка N, що лишилася – просте число і вийти. Інакше, якщо i < 0 − перейти на крок 2.
5. [Наступний дільник]
   Нехай i := i + 1 mod 8, d := d + t[i]. Якщо d > B, то вивести повідомлення про те, що прості дільники N, що лишилися більше B, інакше перейти на крок 3.

Зверніть увагу, що i = −1 поки використовується таблиця простих чисел, та i >= 0 якщо ні.

Метод корисний для видалення малих множників, проте його не слід використовувати для повної факторизації, окрім тих випадків, коли N − дуже маленьке (наприклад, $N < 10^8$).

Решето Ератосфена

Алгоритм, придуманий більше 2000 років тому грецьким математиком Эратосфеном Киренским, вважається першим у своєму роді. Його єдина задача – знаходження всіх простих чисел до деякого заданого числа N. Термін «решето» передбачає фільтрацію, а саме фільтрацію всіх чисел за винятком простих. Так, обробка алгоритмом числової послідовності залишить лише прості числа, всі ж складені числа відсіються.

Розглянемо у загальних рисах роботу методу. Дана упорядкована за зростанням послідовність натуральних чисел, обмежена деяким наперед заданим числом N. Дотримуючись методу Ератосфена, візьмемо деяке число P . Спочатку вважвємо, що це число дорівнює 2 – першому простому числу, і викреслимо з послідовності всі числа кратні P: 2P, 3P, 4P, ..., iP (iP≤N). Далі, з отриманого списку в якості P береться наступне за двійкою число – трійка, викреслюються всі кратні їй числа (6, 9, 12, ...). За таким принципом алгоритм продовжує виконуватися для решти послідовності, відсіваючи всі складені числа в заданому діапазоні.

У наведеній таблиці (Таблиця 1) записані натуральні числа від 2 до 100. Червоним кольором позначені ті, які видаляються в процесі виконання алгоритму «Решето Ератосфена».

Програмна реалізація алгоритму Ератосфена вимагає:

- організувати логічний масив і надати його елементам з діапазону від 2 до N значення логічної одиниці;
- вільну змінну P записати число 2, що є першим простим числом;
- видалити з масиву всі числа кратні P, ступаючи з кроком по P;
- записати в P наступне за ним не закреслене число;
- повторювати дії, описані у двох попередніх пунктах, поки це можливо.

Таблиця 1 - Решето Ератосфена

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100

Зверніть увагу: на третьому кроці ми виключаємо числа, починаючи відразу з P2, це пов'язано з тим, що всі складені числа менші за P будуть вже закреслені. Тому процес фільтрації слід зупинити, коли P2 стане перевищить N. Це важливе зауваження дозволяє поліпшити алгоритм, зменшивши число виконуваних операцій.

Так буде виглядати умовний алгоритму (псевдокод):

```
P←2
поки P²≤N виконувати { i← P ²
якщо B[P]=true то
поки i≤N виконувати {
B[i]←false i←i+P
} P←P+1 }
```

Він складається з двох циклів: зовнішнього і внутрішнього. Зовнішній цикл виконується доти, поки P^2 не перевищить N. Саме ж P змінюється з кроком P+1. Внутрішній цикл виконується лише в тому разі, якщо на черговому кроці зовнішнього циклу виявиться, що елемент з індексом P не закреслений. Саме у внутрішньому циклі відбувається відсіювання всіх складених чисел.

Решето Ератосфена для виявлення всіх простих чисел в заданій послідовності обмеженою деяким N потребує O(Nlog (log N) операцій. Тому доречніше використовувати даний метод ніж, наприклад, найбільш тривіальний і витратний перебір дільників.

Решето Сундарама

Решето Сундарама – алгоритм пошуку всіх простих чисел в деякому заданому діапазоні. Він був розроблений в 1934 році студентом з Індії С. П. Сундарамом.

Принцип роботи алгоритму Сундарама зводиться до послідовного відсіювання всіх непотрібних чисел. Його особливість: результатом роботи алгоритму буде послідовність простих чисел з діапазону від 2 до подвоєного значення деякого числа N. Тож, необхідно отримати всі прості числа до деякого N, тоді вхідними даними будуть всі прості числа від 2 до 2N+1.

Решето Сундарама з ряду натуральних чисел, що не перевищують N, виключає числа виду 2ij+i+j. Результат даного виразу, ні при яких значеннях вхідних у нього змінних, не перевищує N (2ij+i+j≤N). Дотримуючись цієї умови, а також того, що і завжди менше або дорівнює j, змінні і та j пробігають всі натуральні значення з множин:

$$i = 1, 2, 3, \ldots, \frac{\sqrt{2N+1}-1}{2}$$

$$j = i, i+1, i+2, \ldots, \frac{N-i}{2i+1}$$

Після виключення всіх непотрібних чисел необхідно збільшити кожне

залишене число в два рази і додати одиницю (2i+1). У підсумку множина міститиме числа: 2, 3, …, 2N+1.

C**Решето Аткіна**

Алгоритм складається з наступних кроків:

1. Виписуються всі натуральні числа з діапазону від **1** до **n**.

2. Перебираються всі можливі пари чисел **x**, **y**, де **x<=sqrt(n)** і **y<=sqrt(n)**. Тобто (1,1), (1,2),…, (1,sqrt(n)), (2,1), (2,2),…, (sqrt(n),sqrt(n)).

3. Для кожної пари чисел розраховуються значення наступних трьох рівнянь:

a) $\mathbf{4*x^2+y^2}$;

b) $\mathbf{3*x^2+y^2}$;

c) $\mathbf{3*x^2-y^2}$,

й значення обчислюється тільки при **x>y**.

4. Для кожного обчисленого значення рівнянь розраховуються залишки від ділення на **12.**

Причому,

a) якщо залишок дорівнює **1** або **5** для значення першого рівняння;

b) якщо залишок дорівнює **7** для значення другого рівняння;

c) якщо залишок дорівнює **11** для значення третього рівняння,

то у вхідному ряду чисел від **1** до **n** число позначається як просте.

Зауваження*: якщо якесь число Z присутнє в декількох значеннях рівнянь (припустимо a і b), і залишок від ділення на 12 цього числа задовольняє умовам обох груп, то число позначається двічі: спочатку як просте, а потім як складне.*

5. На останньому етапі перевіряється кратність позначених чисел квадратам простих чисел з діапазону від 5 до sqrt(n).

Якщо число кратне квадрату, то воно позначається як складене.

Приклад роботи алгоритму для **n = 40** наведено та описано нижче .

Послідовність виконання алгоритму:

1. Виписуємо всі натуральні числа з діапазону від **1** до **40**.

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	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40

2. Перебираємо всі можливі пари чисел від (1,1) до (6,6) (Тому що sqrt(40) ~ 6).

3. Обчислюємо значення рівнянь для кожної пари чисел x та y.

.

x	y	$4x^2+y^2$	$3x^2+y^2$	$3x^2-y^2$
1	1	**5**	4	–
1	2	8	**7**	–
1	3	**13**	12	–
1	4	20	**19**	–
1	5	**29**	28	–
1	6	40	39	–

x	y	$4x^2+y^2$	$3x^2+y^2$	$3x^2-y^2$
2	1	**17**	13	**11**
2	2	20	16	–
2	3	**25**	21	–
2	4	32	28	–
2	5	41(–)	37	–
2	6	52(–)	48(–)	–

x	y	$4x^2+y^2$	$3x^2+y^2$	$3x^2-y^2$
3	1	**37**	28	26
3	2	40	**31**	**23**
3	3	45(–)	36	–
3	4	52(–)	43(–)	–
3	5	61(–)	52(–)	–
3	6	72(–)	63(–)	–

x	y	$4x^2+y^2$	$3x^2+y^2$	$3x^2-y^2$
4	1	65(–)	49(–)	47(–)
4	2	68(–)	52(–)	44(–)
4	3	73(–)	57(–)	39
4	4	80(–)	64(–)	–
4	5	89(–)	73(–)	–
4	6	100(–)	84(–)	–

5. Обчислюємо залишок від ділення на **12** і позначаємо прості числа.

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 26 27 28 29 30
31 32 33 34 35 36 37 38 39 40

6. Перевіряємо кратність позначених чисел квадратам простих з діапазону від 5 до 6.

5 – просте число, 6 – складене, значить перевіряємо на кратність $5^2=25$ позначені числа. У результаті 25 – потрібно викреслити. У підсумку залишаються тільки прості числа від 1 до 40.

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 26 27 28 29 30
31 32 33 34 35 36 37 38 39 40

Порядок виконання практичної роботи:

1. Повторити основні терміни теорії алгоритмів і теорії чисел, способи отримання таблиць простих чисел (наприклад, методи перебору (простого

ділення); алгоритми Ератосфена; Сундарама і Аткіна і ін.).

2. Для заданих алгоритмів (див. табл. 1 Варіанти завдань до практичної роботи№1, де m та n – граничні значення інтервалу, для якого відшукуються прості числа) вибрати мову програмування і отримати результати (таблиця простих чисел в заданому інтервалі).

3. Проаналізувати результати, відобразити їх у звіті.

Вимоги до оформлення звіту

1) Титульний аркуш

2) Введення (про застосування простих чисел в захисті інформації);

3) Основний розділ, в якому:

– описати алгоритм і відобразити схеми формування таблиць простих чисел обраними методами (див. табл. 1 Варіанти завдань до практичної роботи№1);

– привести програмний код розробки з коментарями;

– відобразити отримані результати (таблицю простих чисел в заданому інтервалі; таблицю порівняння, привести копії екранів.

4) Сформулювати висновки.

5) Надати перелік літератури за темою.

Таблиця 1. Варіанти завдань до практичної роботи №1

№ п\п	**m (нижня межа інтервалу)**	**n (верхня межа)**	**Алгоритм**
1	1500	2113	Аткіна
2	1550	2053	Сундарама
3	500	998	перебору
4	700	1163	Ератосфена
5	560	1143	Сундарама
6	700	1285	Аткіна
7	1000	1583	перебору
8	1250	1727	Ератосфена
9	750	1242	Сундарама
10	500	1021	Аткіна
11	1350	1289	перебору
12	700	1261	Ератосфена
13	1050	1565	Аткіна
14	1150	1617	перебору
15	600	1153	Ератосфена

Питання до самоконтролю

1. Як повинна виглядати й чим обмежуватися вхідна послідовність, придатна для роботи методу Ератосфена?
2. Опишіть мовно алгоритм роботи методу Ератосфена й пропонуйте інструментальний пакет для його графічного відображення.
3. Вкажіть складнисть методу Ератосфена й наведіть методи, перед котрими він має перевагу в швидкодії.
4. Опишіть принцип роботи алгоритму Сундарама.
5. Охарактеризуйте механізм перебору. Наведіть власний приклад

Література до Практичної роботи 1:

Рекомендовані навчальні джерела до виконання Практичної роботи 1: [1-7]

ПРАКТИЧНА РОБОТА 2. ПРОГРАМНА РЕАЛІЗАЦІЯ КЛАСИЧНИХ АЛГОРИТМІВ ШИФРУВАННЯ

Мета роботи: набуття навичок розробки та застосування алгоритмів шифрування, отримати навички програмування та використання класичних алгоритмів.

Теоретичний матеріал до виконання практичної роботи №2

Базові (класичні) алгоритми шифрування

Заміна. В алгоритмі кожна літера відкритого тексту замінюється на якусь букву алфавіту (можливо, на ту ж саму). Для цього відправник повідомлення повинен знати, на яку букву в шифрованому тексті слід замінити кожну літеру відкритого тексту. Часто це робиться шляхом зведення потрібних відповідностей літер у вигляді двох алфавітів. Шифрограма виходить шляхом заміни кожної букви відкритого тексту на записану безпосередньо під нею букву шифрувального алфавіту.

Наприклад, наступне повідомлення отримано шифром заміни літер:

О и г й ь ц я и с д _ с п д с ь _ к и м и ь

Сказати про нього щось однозначне важко, бо замало інформації. Але ключ тут дуже простий, досить побачити відкритий текст.

П і д к л ю ч а й т е_ т р е т ю _ л і н і ю.

Кожна буква тут замінена на її попередницю в алфавіті ("о" замість "п", "к" замість "л", "ц" замість "ч", "д" замість "е", "м" замість "н" тощо). Букви, характерні для українського алфавіту (є, і, ї, ґ) не застосовуються, щоб не полегшувати розшифрування. Це один з варіантів так званого шифру Цезаря. Достовірно відомо, що Ю. Цезар застосовував шифр заміни зі зсувом на три букви вперед при шифруванні.

Більш формальний підхід до розшифрування заснований на використанні середньої частоти появи літер у текстах. Вперше метод був запропонований наприкінці 15-го століття італійським математиком Леоном Баттіста Альберті. Він використовував властивість нерівномірності зустрічі різних букв алфавіту.

Пізніше були визначені середні частоти використання літер мови в текстах. Деякі з них наведені в таблиці 2.1.

Таблиця 2.1 – Частота використання літер в різних мовах (%)

Англійська мова	**E**	(12,9)	**T**	(9,7)	**A**	(8,0)	**I**	(7,5)	**N**	(7,0)	**R**	(7,0)
Німецька мова	**E**	(19,2)	**N**	(10,2)	**I**	(8,2)	**S**	(7,0)	**R**	(7,0)	**T**	(5,9)
Українська мова	**О**	(10,0)	**И**	(6,7)	**Е**	(7,4)	**А**	(7,9)	**Н**	(6,7)	**Ї**	(7,2)

Тепер, маючи шифровані тексти, можна було провести в них частотний аналіз використання символів та на його основі одержати (при необмеженій кількості зашифрованих повідомлень) точні значення всіх букв. Але оскільки

матеріалу, як правило, не дуже багато, то частотний аналіз дає приблизні результати, тому можна лише з високою часткою ймовірності припустити буквене значення найчастіше вживаних символів, а далі - необхідно підставляти їх в шифровані тексти і намагатися вгадувати слова, однак результати з'являються досить швидко. На слабкість шифрів однозначної заміни звернули увагу ще в 15-му столітті. Випадкові припущення - хто і кому пише, назви міст і селищ, часто вживані слова, начебто прийменників, могли призвести до майже миттєвого розкриття шифру. Спроби модифікації ґрунтувалися на багатозначній заміні букв відкритого тексту з використанням ключової послідовності (ключового слова або ключа).

В найбільш чистому вигляді цей підхід можна викласти так. Нехай ми хочемо отримати 10 варіантів (0, 1, ..., 9) заміни кожної букви вхідного тексту (в таблиці 2 варіант заміни визначає величину зсуву за алфавітом).

Придумаємо ключову послідовність цифр 0...9 довільної довжини (наприклад, 1 5 9 8 2 3 7 4 4 8 2 4 7 5 4 3 5). Для відкритого тексту надпишемо над літерами цифри ключа (періодично) і виконаємо зашифрування, вибираючи варіант заміни за цифрою ключа (табл. 2.2). Добре видно, що одні і ті ж букви замінюються по різному, а різні літери можуть бути представлені однаково:

1 5 9 8 2 3 7 4 4 8 2 4 7 5 4 3 5 1 5 9 8
О й, _ н а _ г о р і _ ч е р в о н а _ к а л и н а
П о, _ ц и _ е с ч л _ ю н т ж х т д _ н е м н ц иг

Таблиця 2.2 – Варіанти заміни (літери є, і, ї, ґ не застосовуємо)

	0	1	2	3	4	5	6	7	8	9
А	а	б	в	г	д	е	ж	з	и	й
Б	б	в	г	д	е	ж	з	и	й	к
В	в	г	д	е	ж	з	и	й	к	л
Г	г	д	е	ж	з	и	й	к	л	м
Е	е	ж	з	и	й	к	л	м	н	о
Й	й	к	л	м	н	о	п	р	с	т
И	и	й	к	л	м	н	о	п	р	с
К	к	л	м	н	о	п	р	с	т	у
Л	л	м	н	о	п	р	с	т	у	ф
Н	н	о	п	р	с	т	у	ф	х	ц
О	о	п	р	с	т	у	ф	х	ц	ч
Р	р	с	т	у	ф	х	ц	ч	ш	щ
Ч	ч	ш	щ	ь	ю	я	а	б	в	г
Я	я	а	б	в	г	д	є	е	ж	з

Ідея була запропонована в 16-му столітті французьким дипломатом Блезом де Вижинером. Замість цифр їм використовувалися літери, і ключова послідовність являла собою слово.

Помітно, що алгоритм визначає сукупність перетворень шифру, що відрізняються параметром - ключовою послідовністю шифрування (ключем). Це дозволяє будувати надійну криптосистему на підставі фіксованого (несекретного) алгоритму шифрування, але секретного ключа, який регулярно змінюється. Теоретично такий шифр піддається дешифруванню на основі частотного аналізу вживання букв, але для цього потрібно, щоб довжина шифроповідомлень, зроблених з цим ключем, значно перевищувала довжину самого ключа.

Перестановка. У цьому шифрі всі букви відкритого тексту залишаються без змін, але переставляються за заздалегідь обумовленим правилом. Вітається використання ключа, який керує процедурою шифрування. Ключове слово може бути використане для отримання шифрованої числової послідовності шляхом нумерації букв ключового слова (відносно одна одної) в порядку їхнього проходження зліва направо в стандартному алфавіті. Далі під числовою послідовністю в рядках, рівних за довжиною ключовому слову, записаний відкритий текст. Далі, у процесі шифрування текст виписується вже за окремими стовпцями в порядку, визначеному даною числовою послідовністю.

У класичному методі перестановки застосовується перестановка стовпців. Більш пізні варіанти пропонують й інші «маршрути» перестановки (діагональна - зліва направо або справа наліво, або ж почергово лівий і правий напрямки; або по спіралі,або у вигляді різних геометричних фігур). Допускається й подвійне шифруванні шляхом повторної перестановки стовпців. Ключове слово може бути як одне і те ж, так і є можливість використовувати різні ключові слова (подвійна перестановка).

Найпростіший варіант перестановки - прямокутна таблиця з секретним розміром стовпця (рис. 2.1), куди вхідний текст записується по стовпцях, а шифроповідомлення зчитується за рядками.

П	л	й	т	ю	н
і	ю	т	р	-	і
д	ч	е	е	л	ю
к	а	-	т	і	-

Рисунок 2.1 – Шифр табличної перестановки

Відкритий текст: Підключайте третю лінію
Шифроповідомлення: Плйтюніютр ідчеелюка ті

При розшифруванні довжину повідомлення слід розділити на довжину стовпчика, щоб визначити довжину рядка, вписати шифроповідомлення в таблицю по рядках, а потім прочитати відкритий текст.

Інший варіант – кодування перестановкою по групах символів, використовуючи деякі зигзагоподібні шаблони, наприклад, як показано на рисунку 2.2.

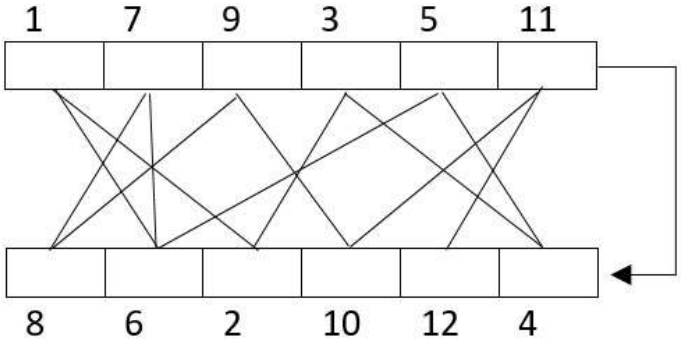


Рисунок 2.2 – Зигзагоподібний шифр перестановки

Варто записати символи відкритого тексту за зигзагом, а прочитати по колу (або навпаки) – і шифоповідомлення готове, якщо здається ненадійним, то можна ввести додаткові ускладнення.

Використовуються і більш складні (ручні) системи, а також групові. Наприклад, у квадраті (рис. 2.3), що складається з 4 малих квадратів з певною нумерацією клітин, вирізують 4 клітини під різними номерами. Квадрат кладеться в початкове положення («1» – згори) і в отвори (зліва направо/зверху вниз) вписуються букви відкритого повідомлення. Потім квадрат повертається проти годинникової стрілки на 90 градусів («2» – згори) і знов вписуються наступні літери, потім повторюємо процес для положення «3» і «4». Якщо залишаються вільні клітини – вони заповнюються довільними символами. Шифр повідомлення отримують, прочитавши по стовпцях або рядках послідовність записаних у прямокутнику букв.

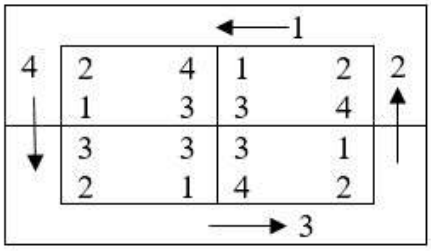


Рисунок 2.3 – Шифрувальний квадрат

Розглянуті вище шифри перестановки з сучасної точки зору є абсолютно подібними, бо представляють собою послідовність елементарних

процедур перестановки групи символів.

Дешифрування повідомлень, отриманих шифром перестановки, значно важче, ніж при використанні шифрів заміни. Теоретичної передумови, крім перебору варіантів, не існує, хоча окремі здогадки можуть спростити завдання.

Дроблення. При цьому кожній букві відкритого тексту відповідає більше одного символу шифрованого тексту, після чого символи перемішуються (переставляються) у певному порядку. Нижче наведена система, що демонструє процедуру дроблення з використанням знаменитого шифру Bifid, авторство якого приписується французькому криптографу Феліксу Марі Деластеллю.

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

	1	2	3	4	5	6
1	A	B	C	D	E	F
2	G	H	I	J	K	L
3	M	N	O	P	Q	R
4	S	T	U	V	W	X
5	Y	Z	1	2	3	4
6	5	6	7	8	9	0

Рисунок 2.4 - Полібіанський квадрат 5x5 (а) та 6x6 (в)

Спочатку складається шифрувальна таблиця розміром 5x5 (так званий полібіанський квадрат), куди послідовно, порядкам вписується шифрувальний алфавіт з ключовою фразою, причому заради того, щоб загальне число букв алфавіту не перевищувало 25, літера J опускається (оскільки ця буква, з одного боку, маловживана в англійських текстах, а з іншого – цілком може бути замінена буквою I, без будь-якої шкоди для змісту).

Далі в процесі шифрування під кожною буквою відкритого тексту записуються в стовпчик її табличні координати – номер рядка, а нижче номер стовпця, а потім отримана цифрова послідовність перекладається за допомогою тієї ж таблиці назад в буквену форму, але на цей раз вона читається вже в рядок. При такому шифруванні координата рядка і координата стовпця кожної букви виявляються роз’єднаними, що характерно саме для роздроблюючого шифру.

Порядок виконання практичної роботи №2

1. Ознайомитися з теоретичним матеріалом до практичної роботи.

2. Використовуючи запропонований в таблиці варіантів алгоритм, скласти блок-схему і програму шифрування та дешифрування повідомлень одним з базових методів.

3. Запропонувати модифікацію алгоритму, що покращує його криптографічну стійкість (наприклад, додавання ключа, використання «складеного» алфавіту, іншої кодової таблиці та інше), або підвищує

ефективність (швидкодія, пам’ять, кількість обчислювальних операцій).

4. Виконати шифрування довільного, обраного студентом повідомлення.

5. Виконати дешифрування отриманого в п.3. повідомлення, порівняти з вхідним.

6. Проаналізувати основні характеристики базового і модифікованого алгоритму.

Вимоги до оформлення звіту

1. Титульний лист.

2. Вступ (про основні алгоритми шифрування); 3.

Основний розділ, в якому:

– описати схему отримання зашифрованого повідомлення, пояснити основні особливості;

– скласти алгоритм (відобразити блок-схему та програму роботи базового алгоритму (вибір мови - довільний);

– описати запропоновану модифікацію алгоритму, відобразити алгоритмічно і програмно;

– показати результати порівняння алгоритмів і роботи програми, зашифрувавши, а потім і розшифрувавши повідомлення.

4. Сформулювати висновки.

5. Надати перелік літератури за темою.

Таблиця 2. Варіанти завдань до практичної роботи №2

	Базовий алгоритм	**Особливості ключа, метод**
1	Однозначна заміна	Зворотній алфавіт
2	Перестановка	Шифрувальний квадрат
3	Дроблення	Полібіанський квадрат 6*6
4	Багатозначна заміна	Кількість стовпчиків = 6
5	Перестановка	діагональ зліва направо
6	Дроблення	З використанням шифру Bifid на 5 символів
7	Однозначна заміна	Зсув на 2 попередніх букви
8	Перестановка	За довільним шаблоном
9	Дроблення	Шифр Bifid на 6 символів
10	Однозначна заміна	Схема Цезаря
11	Перестановка	Таблична 7*4
12	Дроблення	Полібіанський квадрат 7*7
13	Багатозначна заміна	Кількість стовпчиків = 8
14	Однозначна заміна	Зсув на 5 символів назад
15	Перестановка	Шифр – таблична перестановка 5*7

Питання до самоконтролю

1. Охарактеризуйте механізм «Однозначна Заміна». Наведіть власний приклад
2. Охарактеризуйте механізм «Багатозначна Заміна». Наведіть власний приклад

2. Охарактеризуйте механізм «Перестановка». Наведіть власний приклад
3. Охарактеризуйте механізм «Дроблення». Наведіть власний приклад

Література до Практичної роботи 2:

Рекомендовані навчальні джерела до виконання Практичної роботи 2: [8-21]

ПРАКТИЧНА РОБОТА 3. АЛГОРИТМИ СТИСНЕННЯ ІНФОРМАЦІЇ

Мета роботи: отримати навички програмування та використання алгоритмів стиснення інформації.

Теоретичний матеріал до виконання практичної роботи №3

Алгоритми стиснення інформації

Метод Шеннона-Фано. Цей метод вимагає впорядкування вихідної множини символів по не зростанню їх частот. Потім виконуються наступні кроки:

а) список символів ділиться на дві частини (назвемо їх першою і другою частинами) так, щоб суми частот обох частин (назвемо їх Σ_1 і Σ_2) були точно або приблизно рівні. У разі, коли точної рівності досягти не вдається, різниця між сумами повинна бути мінімальна;

б) кодовим комбінаціям першої частини дописується біт «1», кодовим комбінаціям другої частини дописується біт «0»;

в) аналізують першу частину: якщо вона містить тільки один символ, робота з нею закінчується, – вважається, що код для її символів побудований, і виконується перехід до кроку г) для побудови коду другої частини. Якщо символів більше одного, переходять до кроку а) і процедура повторюється з першою частиною як з самостійним упорядкованим списком;

г) аналізують другу частину: якщо вона містить тільки один символ, робота з нею закінчується і виконується звернення до списку, що залишився (крок д). Якщо символів більше одного, переходять до кроку а) і процедура повторюється з другою частиною як з самостійним списком;

д) аналізується список, що залишився: якщо він порожній-код побудований, робота закінчується. Якщо ні, – виконується крок а).

Приклад 1. Дано символи «a», «b», «c», «d» з частотами p_a=0,5; p_b=0,25; p_c=0,125; p_d=0,125. Побудувати ефективний код методом Шеннона-Фано. Зведемо вихідні дані в таблицю, упорядкувавши їх по не зростанню частот:

Вхідні символи	Частоти символів
a	0,5
b	0,25
c	0,125
d	0,125

Перша лінія поділу проходить під символом a: відповідні суми Σ_1 і Σ_2 рівні між собою і рівні 0,5. Тоді формованим кодовим комбінаціям дописується 1 для верхньої (першої) частини і 0 для нижньої (другої) частини. Оскільки це перший крок формування коду, двійкові цифри не дописуються, а тільки починають формувати код:

Вхідні символи	Частоти символів	Формований код
a	0,5	1
b	0,25	0
c	0,125	0
d	0,125	0

В силу того, що верхня частина списку містить тільки один елемент (символ a), робота з нею закінчується, а ефективний код для цього символу вважається сформованим (в таблиці, наведеній вище, ця частина списку частот символів виділена заливкою). Другий поділ виконується під символом b: суми частот Σ_1 і Σ_2 знову рівні між собою і рівні 0,25. Тоді кодової комбінації символів верхньої частини дописується 1, а нижньої частини – 0. Таким чином, до отриманих на першому кроці фрагментів коду, рівним 0, додаються нові символи:

Вхідні символи a	Частоти символів	Формований код
b c	0,5	1
d	0,25	01
	0,125	00
	0,125	00

Оскільки верхня частина нового списку містить тільки один символ (b), формування коду для нього закінчено (відповідний рядок таблиці знову виділена заливкою). Третій поділ проходить між символами c і d: до кодової комбінації символу c приписується 1, коду символу d приписується 0:

Вихідні символи	Частоти символів	Формований код
a	0,5	1
b	0,25	01
c	0,125	001
d	0,125	000

Обидві половини вихідного списку, які залишилися, містять по одному елементу, тому робота зі списком в цілому закінчується. Таким чином, отримали коди: a – 1, b – 01, c – 001, d – 000.

Визначимо ефективність побудованого коду за формулою: $I_{ср}$ = 0,5*1 + 0,25*2 + 0,125*3 + 0,125*3 = 1,75.

Оскільки при кодуванні чотирьох символів кодом постійної довжини потрібно два двійкових розряду, заощаджено 0,25 двійкового розряду в середньому на один символ.

Метод Хаффмана. Цей метод має дві переваги в порівнянні з методом Шеннона-Фано: він засвоює неоднозначність кодування, що виникає через приблизні рівності сум частот при поділі списку на дві частини (лінія поділу проводитися неоднозначно), і має, в загальному випадку, велику ефективність коду. Початкова безліч символів впорядковується за не зростанням частот, після чого виконуються наступні кроки:

1) об’єднання частот:

– дві останні частоти списку складаються, а відповідні символи виключаються зі списку;

– після виключення символів залишившийся список поповнюється сумою частот і знову впорядковується;

– попередні кроки повторюються до тих пір, поки ні вийде одиниця в результаті підсумовування і список ні зменшиться до одного символу.

2) побудова кодового дерева:

– будується двійкове кодове дерево: коренем його є вершина, отримана в результаті об’єднання частот, рівна 1, листям – вихідні вершини;

– інші вершини відповідають або сумарним, або вихідним частотам, причому для кожної вершини ліва підпорядкована вершина відповідає більшому доданку, а права – меншого;

– ребра дерева пов’язують вершини – суми з вершинами – складовими.

Структура дерева показує, як відбувалося об’єднання частот. Ребра дерева кодуються: кожне ліве кодується одиницею, кожне право – нулем.

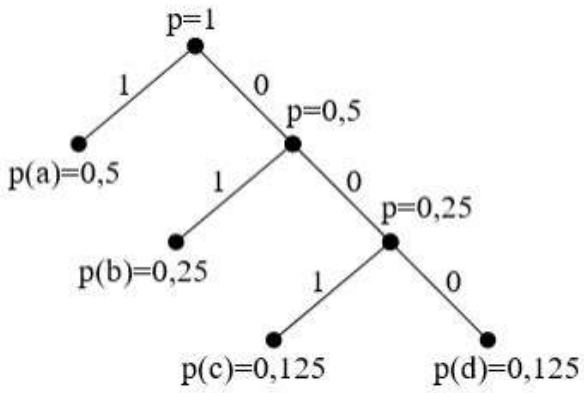
3) формування коду: для отримання кодів листя (вихідних кодованих символів) просуваються від кореня до потрібної вершині і «збирають» ваги прохідних ребер.

Розглянемо приклад. Дано символи a, b, c, d з частотами p(a) = 0,5; p(b) = 0,25; p(c) = 0,125; p (d) = 0,125. Побудувати ефективний код методом Хаффмана.

1) об’єднання частот (результат об’єднання двох останніх частот в списку виділено в правому сусідньому стовпці штрихуванням).

Вхідні символи	Частоти символів	Етапи об’єднання		
		перший	другий	третій
a	0,5	0,5	0,5	1
b	0,25	0,25	0,5	
c	0,125	0,25		
d	0,125			

2) Побудова кодового дерева.



3) формування коду:

K(a) = 1; K(b) = 01; K(c) = 001; K(d) = 000.

Як видно, отримані коди збігаються з тими, що були сформовані методом Шеннона-Фано, отже, вони мають однакову ефективність.

5) Перелік застосованих джерел.

Порядок виконання

1. Ознайомитися з теоретичним матеріалом до практичної роботи.
2. Використовуючи обраний варіант алгоритму, скласти блок-схему і програму шифрування та дешифрування повідомлень одним з базових методів.
3. Виконати програмну реалізацію методу Шеннона-Фано або методу Хаффмана (на вибір) на будь якій мові програмування. Програма повинна виводити результати в текстовому виді.

Варіант завдання (фрагмент символів) вибирається відповідно до номера студента в журналі групи (N) з таблиці 3.1. При роботі з даним фрагментом слід вважати всі заголовні букви строковими (маленькими), а знаки пунктуації ігнорувати. Пробіл слід розглядати як самостійний символ.

Вимоги до оформлення звіту

1. Титульний лист.
2. Вступ (про основні алгоритми стиснення інформації).
3. Відобразити блок-схему та програму роботи базового алгоритму (вибір мови - довільний).
4. Лістинг програми.
5. Результат роботи програми.
6. В исновки.
7. Перелік джерел за темою.

Таблиця 3.1. Варіанти завдань до практичної роботи №3

N	Фрагмент тексту
1	У усмішки закохана обліплена поглядами в блакитному небі червона калина
2	Зібрала Маргарита маргаритки на горі, втратила Маргарита щастя на війні
3	Виростив Толя тополю за полем. Гарна тополя, та не піде до тополі Толя
4	Орел на горі, перо на орлі. Гора під орлом, орел під пером. Ото ж дивно
5	Бредуть сувові бобри у сирі бори. Бобри суворі, а до бобрят своїх ласкаві
6	В семеро саней семеро Семенів з вусами впряглися самі та трясуть чубами
7	Два хлопця в куточку щока до щоки розв'язуютьщось та гризуть олівці
8	Косар Касян косою косить косо. Не скосить косар Касян косовиці
9	Ой, над полем – небо синє, поле висохло, в полині, стало полем бою нині
10	Стоїть піп на копиці, ковпак на потилиці. Копиця під попом, піп під ковпаком
11	Протокол про протокол хутко підписали, протоколом запротоколювали
12	Рапортував, та не дорапортував, дорапортував, так зарапорувався
13	Чотири чорненьких чумазих чортеня креслили чорним чорнилом креслення
14	Лізуть кози в грозу в лозу – лозу кози в грозу гризуть
15	Близько кола дзвони, там де звони – горе, біля воріт коловоріт

Питання до самоконтролю

1. Опишіть основні кроки виконання методу Шеннона-Фано
2. Вкажіть умову закінчення роботи зі строкою при застосуванні методу Шеннона-Фано
3. Вкажіть переваги метода Хаффмана відносно методу методу Шеннона-Фано
4. Що таке кодове дерево. Поясніть принципи його формування

Література до Практичної роботи 3:

Рекомендовані навчальні джерела до виконання Практичної роботи 3: [22-30]

ПРАКТИЧНА РОБОТА 4. ШИФРУВАННЯ ІНФОРМАЦІЇ З ВИКОРИСТАННЯМ ОБОРОТНИХ МАТЕМАТИЧНИХ ФУНКЦІЙ

Мета роботи: отримати навички шифрування інформації з використанням оборотних математичних функцій.

Теоретичний матеріал до виконання практичної роботи №4

1. Кодування кожного блоку інформації виконується шляхом *послідовного* застосування двох оборотних функцій. Наприклад, спочатку прочитаний з файлу блок обробляється операцією XOR, а потім – операцією зсуву. Дешифрування виконується в зворотному порядку: спочатку над прочитаним з файлу зашифрованим блоком робиться операція зсуву у зворотний бік, а потім – операція XOR.

2. Обробка інформації, згідно із завданням, виконується над блоками заданої довжини. Це означає, що крім самого алгоритму шифрування/розшифрування, студент повинен використовувати в своїй програмі функції та структури даних для побітової роботи з файлами. Оскільки в стандартних бібліотеках мови Сі такі функції відсутні, вони повинні бути розроблені студентом самостійно.

3. Довжина ключа (кількість символів ключа) повинна бути обмежена, наприклад, не менше 5 і не більше 15 символів. Дані вимоги повинні підтримуватися інтерфейсом програми.

4. Важливим моментом у завданні є вибір того, як буде використовуватися ключ в процесі шифрування даних. На кожен шифрований блок інформації повинна бути витрачена певна частина інформації ключа. Та ж частина ключа повинна бути витрачена на розшифрування інформації. Вибір способу, за яким буде використовуватися ключ при шифруванні даних, залишається за студентом. Інформація, що міститься в ключі, повинна використовуватися в оборотних функціях, зазначених у таблиці 1.

Один з очевидних рішень є наступне: для кожного блоку інформації довжиною S біт використовувати частину ключа довжиною S біт. Однак це не завжди доцільно. Потрібно намагатися, щоб всі частини ключа в процесі шифрування і розшифрування використовувалися рівномірно і однаково ефективно.

Порядок виконання

Написати програму на довільній мові програмування, що виконує наступні дії:

1. Введення ключа (послідовності символів, що використовується в подальшому для шифрування інформації).

2.1. Послідовне читання вмісту довільного файлу блоками заданого розміру.

2.2. Шифрування кожного прочитаного блоку інформації з використанням двох заданих оборотних функцій на основі інформації, що міститься в ключі.

2.3. Послідовний запис зашифрованого блоку інформації у вихідний файл (відмінний від вхідного).

3. Розшифрування вмісту зашифрованого файлу на основі того ж ключа шляхом взаємно-оборотного застосування оборотних функцій, які були використані при шифруванні файлу.

Програма повинна мати елементарний інтерфейс, в якому здійснюються введення ключа, вибір імен файлів для шифрування і розшифрування.

Варіант завдання визначається на підставі значення N, який є номером студента в журналі групи.

З таблиці 4.1 студент вибирає дві оборотних функції, що використовуються при шифруванні, з таблиці 4.2 – розмір блоку інформації (в бітах), який буде шифруватися на одному кроці роботи.

Таблиця 4.1. Оборотні функції за варіантом

N mod 8	Додавання по модулю	XOR	Циклічне зрушення	Таблиця підстановки
0		+		+
1	+			+
2		+	+	
3	+	+		
4	+		+	
5			+	+
6	+			+
7		+	+	

Таблиця 4.2. Розмір блоку інформації

N mod 10	Розмір блоку (біт)	N mod 10	Розмір блоку (біт)
0	5	5	10
1	9	6	6
2	13	7	12
3	7	8	14
4	11	9	8

Вимоги до оформлення звіту

1. Титульний аркуш.
2. Варіант завдання.
3. Лістинг програми.
4. Екрані форми роботи програми.
5. Висновки за результатами роботи.
6. Перелік джерел за темою.

Питання до самоконтролю

1. Розкрийте поняття «Оборотна математична функція»
2. Охарактеризуйте особливості першого етапу роботи алгоритму
3. Поясніть, що таке побітова робота з файлами та особливості цього процесу при створенні програмних реалізацій
4. Поясніть вимоги алгоритму до роботи з ключами

Література до Практичної роботи 4:

Рекомендовані навчальні джерела до виконання Практичної роботи 4: [32-37]

ПРАКТИЧНА РОБОТА 5. ПОБУДОВА І ДОСЛІДЖЕННЯ ГЕНЕРАТОРІВ ПСЕВДОВИПАДКОВИХ ЧИСЕЛ

Мета роботи: отримати навички побудови та дослідження генераторів псевдовипадкових чисел.

Теоретичний матеріал до виконання практичної роботи №5

1. Спочатку слід задатися розмірами графічної області, в якій будуть малюватися точки. Мінімальними рекомендованими розмірами області виведення точок є 500 на 500 точок (краще – більше).

2. Кожен генератор формує наступне випадкове число на підставі попереднього випадкового числа. Початкове число генератора («затравка», «зерно», «seed»), на підставі якого генерується перше випадкове число, повинно формуватися на підставі ключа (введеного користувачем набору символів).

3. При малюванні точок повинні виконуватися наступні дії:

– з допомогою генератора ПВЧ генерується число, яке приводиться до формату горизонтальної координати точки в обраній графічної області;

– з допомогою генератора ПВЧ генерується число, яке приводиться до формату вертикальної координати точки в обраній графічної області; – за отриманими координатами малюється точка, колір якої контрастний по відношенню до кольору фону графічної області;

– цикл малювання точок повторюється нескінченну кількість разів, до тих пір, поки не буде перервано користувачем.

4. Програма повинна мати можливість малювати крапки з використанням кожного з трьох розроблених генераторів ПВЧ. Тобто, спочатку «підключаємо» перший генератор і дивимося, наскільки рівномірно він заповнює графічну область (чекаємо приблизно хвилину). Потім підключаємо другий генератор, потім третій. По рівномірності заповнення графічної області можна судити про ефективність розробленого генератора ПВЧ.

Порядок виконання

Дане завдання виконується всіма студентами з одного і того ж завдання – різних варіантів завдань тут не передбачено. Тим не менше, кожен студент повинен виконувати це завдання самостійно, не використовуючи фрагменти програмного коду, написані іншими студентами.

1. Розробити три генератора ПВЧ:

– генератор цілих чисел на основі лінійного конгруентного генератора; – генератор чисел з плаваючою точкою на основі алгебраїчних функцій; –

генератор бітової послідовності на основі моделі зсувного регістру. Кожен генератор повинен ініціалізується значенням, отриманим на підставі введеного ключа (набору символів певної довжини).

2. Виконати графічну візуалізацію ефективності кожного з розроблених генераторів шляхом заповнення псевдовипадковими точками прямокутної області на екрані комп'ютера.

3. Вибрати найбільш ефективний генератор ПВЧ серед трьох розроблених генераторів.

Вимоги до оформлення звіту:

1. Титульний аркуш. 2.
Варіант завдання. 3.
Лістинг програми.
4. Екрані форми роботи програми. 5.
Висновки за результатами роботи.
6. Перелік джерел за темою

Питання до самоконтролю

1. Поясніть поняття «Генератор псевдовипадкових чисел»
2. Опишіть значеннята застосування генераторів псевдовипадкових чисел в захисті інформації
3. Вкажіть особливість алгоритму роботи генератора при формуванні нового (наступного) випадкового числа
4. З якими складнощами зустрічається розробник при створенні графічного образу роботи генератора ?

Література до Практичної роботи 5:

Рекомендовані навчальні джерела до виконання Практичної роботи 5: [12-13, 38-44]

ПРАКТИЧНА РОБОТА 6. ШИФРУВАННЯ ДАНИХ НА ОСНОВІ БАЗОВИХ КРИПТОГРАФІЧНИХ МЕТОДІВ

Мета роботи: отримати навички шифрування даних на основі базових криптографічних методів.

Теоретичний матеріал до виконання практичної роботи №6

1. Фактично, в даному завданні реалізується шифрування із застосуванням одноразового блокнота. Сам одноразовий блокнот генерується за допомогою генератора ПВЧ на підставі деякого заданого ключа. Для зашифрованого подібним чином файлу робиться неможливим застосування частотного аналізу, що ускладнює процес розшифрування. У програмі не потрібна будь-яка візуалізація результатів роботи. Інтерфейс програми повинен бути схожий з інтерфейсом першого завдання. Як і в разі першого завдання, дана програма повинна вміти шифрувати і розшифровувати будь-які типи файлів, розглядаючи їх як послідовність однакових бітових блоків заданої довжини.

Порядок виконання

Розробити програму, що реалізує шифрування і розшифрування з використанням псевдовипадкової послідовності.

Дана Практична робота за своєю суттю повинна об'єднувати в собі третю і четверту лабораторні роботи. Шифрування і дешифрування файлу виробляються тими ж методами і тими ж блоками, що і в першому завданні, але при цьому використовується не інформація заданого ключа, а інформація, що формується генераторами ПВЧ, розробленими в другому завданні. А ось ініціалізація генераторів повинна здійснюватися за допомогою ключа, що вводиться з клавіатури.

Вимоги до оформлення звіту:

1. Титульний аркуш.
2. Варіант завдання.
3. Лістинг програми та екрані форми роботи програми.
4. Висновки за результатами роботи.
5. Перелік літератури до теми

Питання до самоконтролю

1. З застосуванням лекційного матеріалі надайте визначення поняттям «шифр», «шифрування!, «дешифрування»
2. Вкажіть основні типи криптографічних алгоритмів, щозастосовуються в процесі захисту інформації
3. До якоготипу криптографічних алгоритмів можна віднести перетворення «Одноразовий блокнот»?

Література до Практичної роботи 6:

Рекомендовані навчальні джерела до виконання Практичної роботи 6: [40-48]

ІНДИВІДУАЛЬНА НАУКОВО-ДОСЛІДНА РОБОТА. ДОСЛІДЖЕННЯ АЛГОРИТМІВ СТИСНЕННЯ ІНФОРМАЦІЇ

Завдання:

Варіант завдання (фрагмент символів) вибирається відповідно до номера студента в журналі групи (N) з таблиці 1. При роботі з даним фрагментом слід вважати всі заголовні букви строковими (маленькими), а знаки пунктуації ігнорувати. Пробіл слід розглядати як самостійний символ.

Для заданого фрагменту символів виконати наступне:

1. Закодувати фрагмент символів рівномірним кодом мінімально достатньої розрядності. Визначити ентропію (середнє число біт на один символ) і довжину закодованого тексту в бітах.

2. Закодувати фрагмент символів рівномірним однобайтовим кодом. Визначити ентропію і довжину закодованого тексту в бітах.

3. Закодувати фрагмент символів методом Шеннона-Фано. Визначити ентропію і довжину закодованого тексту в бітах.

4. Закодувати фрагмент символів методом Хаффмана. Визначити ентропію і довжину закодованого тексту в бітах.

5. Результати кодування занести в зведену таблицю результатів кодування:

Метод кодування	Ентропія	Довжина закодованого фрагмента тексту (в бітах)
1. Рівномірне кодування кодом мінімально достатньої розрядності		
2. Рівномірне кодування однобайтовим кодом		
3. Кодування методом Шеннона-Фано		
4. Кодування методом Хаффмана		

6. Зробити висновки про ефективність тих чи інших способів кодування.

Вимоги до оформлення звіту:

1. Титульний аркуш
2. Завдання до розрахункової роботи за варіантом
3. Теоретичний опис кожного з методів
4. Ручний розрахунок закодування фрагменту символів
5. Висновки
6. Перелік літератури за темою

Таблиця 7.1– Варіанти завдань до розрахункової роботи

N	Фрагмент тексту
1	Коли стелиться доріжка, козакові не до ліжка
2	То не козак, що не думає отаманом бути
3	Козацькому роду нема переводу
4	Або волю добути, або дома не бути
5	А шо козакові треба: степу, хліба й неба!
6	Береженого Бог береже, а козака – шабля
7	Вольному – Воля, скаженому – круча, спасенному – рай
8	Добрий козак баче, де отаман скаче
9	Здобувши перемогу, про пильність не забувай
10	Кінь, шаблюка, вітер в полі і ніхто не здолає козацької волі
11	Козацькому роду нема переводу
12	Козак без коня, що воїн без пістоля
13	Козак дружбу знає: в біді коня не покидає
14	Козак душа не вередлива, в пеклі не мерзне, в ополонці не пріє
15	Козаку зібратись тільки підпоясатись.
16	Козак із пригорщі нап'ється, а з долоні пообіда.
17	Козача потилиця панамляхам не хилиться
18	Коли навколо вороги то не життя, честь бережи
19	Коли всім народом дмухнути, то ураган буде
20	Не той козак, що за водою пливе, а той, що проти води
21	Не лише силою боротися треба, а і вмінням
22	Не той сильний, що камінь верже, а той, що серце в собі держе
23	Рана від кулі глибока, а від шаблі широка
24	У нас де крак там і козак, а де байрак там сто козаків
25	Хто по морю плавав, тому калюжа не страшна.
26	Козак не боїться ні хмари, ні чвари, ні чортової кари
27	Знищ ворога сьогодні, бо завтра нашкодить.
28	Не пив води дунайської, не їв каші козацької
29	Кров не водиця, проливати не годиться.
30	Не наше діло у ряжі стрибати, наше діло конем по степу гуляти

Приклад виконання розрахункової роботи

Нехай задана наступна фраза: «Ледве-ледве Олена їла, від ліні не хотіла».

Розпишемо фразу по символам, залишивши (для спрощення) тільки пробіли між словами, а також зробивши всі літери в одному регістрі. Для обробки реального тексту (наприклад, при програмної реалізації методів кодування) подібні спрощення неприйнятні.

л	е	д	в	е		л	е	д	в	е		о	л	е	н	а		ї	л
а		в	і	д		л	і	н	і		н	е		х	о	т	і	л	а

Відзначимо, що даний фрагмент тексту, містить 40 символів. Складемо таблицю 7.2, в якій перерахуємо всі унікальні символи, їх кількість у тексті і частоти.

Частота символу визначається як відношення числа появи цього символу до загального числа символів в тексті. В останньому стовпці обчислено кількість інформації на символ з урахуванням його ймовірності.

Таблиця 7.2

Символ	Кількість	Частота (p)	$-p*\log_2 p$
пробіл	7	0,175	0,440125
л	6	0,15	0,41055
е	6	0,15	0,41055
д	3	0,075	0,280275
в	3	0,075	0,280275
о	2	0,05	0,2161
н	3	0,075	0,280275
а	3	0,075	0,280275
ї	1	0,025	0,440125
і	4	0,1	0,3322
х	1	0,025	0,440125
т	1	0,025	0,440125

Підсумувавши значення в останньому стовпчику таблиці, отримаємо «чисту» ентропія тексту (без урахування способу кодування символів), що дорівнює $4{,}251 \approx 4$.

Вважається, що не можна закодувати текст таким чином, при якому середнє число бітів на символ (середня ентропія) буде менше цього значення.

1. Кодування символів рівномірним кодом мінімальної розрядності.

Наш текст містить 12 різних символів, для кодування яких досить $]\log_2 12[= 4$ двійкових розрядів. Можна закодувати символи послідовними кодами, починаючи з нуля (таблиця 7.3).

Таблиця 7.3

Символ	_	л	е	д	в	о	н	а	ї	і	х	т
Код	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011

Ентропія (достатнє число біт на один символ) складе (для випадку рівномірного кодування) рівно 4 біта на символ: $I_1 = 4$.

Література до виконання ІНДР:

Рекомендовані навчальні джерела до виконання Індивідуальної науково-дослідної роботи: [22-30]

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

Основна

1. Колмогоров, А.Н. Алгебра и начала анализа : учеб. для 10-11 кл. общеобразоват. учрежд. / под ред. А.Н. Колмогорова. – М. : Просвещение, 2002. – 384 с.
2. Сборник задач по математике с решениями. 7-11 классы / М.И. Сканави, В.В. Зайцев, А.М. Суходский ; под ред. М.И. Сканави. – М. : ОНИКС, 2011. – 624 с.– (Сер. «Ваш домашний репетитор»).
3. Шейко, В.М.Організація та методика науково-дослідницької діяльності : підручник / В.М. Шейко, Н.М. Кушнаренко. – 6-те вид., перероб. та допов. – К. : Знання, 2008. – 310 с.
4. Решето Аткина [Електронний ресурс]. – Режим доступу: https://pantini.gitbook.io/pantini-co/bits/resheto-atkina
5. Atkin, O.L. Prime sieves using binary quadratic forms / O.L. Atkin, D.J. Bernstein // Mathematics of Computation. – V.73, N.246. – 2003. – P. 1023-1030.
6. Василенко, О.Н. Теоретико-числовые алгоритмы в криптографии : монография / О.Н. Василенко. – М. : МЦНМО, 2003. – 328 с.
7. Закон України «Про інформацію» від 2 жовтня 1992 року № 2657-XII із змінами та доповненнями [Електронний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/2657-12#Text
8. Лужецький, В.А. Основи інформаційної безпеки [Електронний ресурс] : навч. посіб. / В.А. Лужецький, А.Д. Кожухівський, О.П. Войтович. – Вінниця : ВНТУ, 2009. – 268 с. – Режим доступу: https://studfile.net/preview/6012701/
9. Основи інформаційної безпеки : навч. посіб. / В. Б. Вишня, О. С. Гавриш, Е. В. Рижков. – Дніпро : Дніпроп. держ. ун-т внутріш. справ, 2020. – 128 с.
10. Гулак, Г. М. Основи криптографічного захисту інформації : підручник / Г.М.Гулак, В.А. Мухачев, В.О. Хорошко та ін. – Вінниця : ВНТУ, 2012. – 822 с.
11. Лужецький, В.А. Основи інформаційної безпеки / В.А. Лужецький, А.Д. Кожухівський, О.П. Войтович : навч. посіб. – Вінниця : ВНТУ, 2013. – 268 с.
12. Кузнецов, О. О. Захист інформації в інформаційних системах / О. О. Кузнецов, С. П. Євсеєв, О. Г. Король. – Харків : ХНЕУ, 2011. – 510 с.
13. Кузнецов О. О. Методи традиційної криптографії / О. О. Кузнецов, С. П. Євсеєв, О. Г. Король. – Харків : ХНЕУ, 2012. – 316 с.
14. Нужний, В. Використання технологій захисту даних THALES // Енергетика та електрифікація. – 2020. – № 5. – С. 17-20.
15. Shannon, Cloude. Communication Theory of Secrecy Systems / Cloude Shannon // Bell System Technical Journal, Vol.28(4) (Oct.). – 1949. – P. 656-715.
16. Denning, D. Cryptography and data security / Dorothy E. R. Denning. –

Addison-WeslayPublishing Company, 1982. – 400 p.
17. Russel, D.Computer Security Basics / Deborah Russell, G. T. Gangemi, G. T. Gangemi, Sr. – O`Reilli &Associates, Inc., 1991. – 448 p.
18. Jackson, K. Computer Security Reference Book / K M Jackson, J. Hruska. – Butterworth-Heinemann Ltd., 1994. – 984 p.
19. Методи безпечної оброрбки інформації у багатопозиційних системах радіолокації : монографія / І. Пархомей, В. Козловський, С. Гнатюк та ін. – К. : ЦУЛ, 2019. – 230 с.
20. Шаньгин, В.Ф. Защита компьютерной информации. Эффективные методы и средства / В.Ф. Шаньгин. – М. : ДМК Пресс, 2008. – 544 с.
21. Архипова, О.О. Частотний аналіз використання букв української мови / О.О. Архипова, В.М. Журавльов // Радіоелектроніка. Інформатика. Управління. – 2009. – № 2(21). – С. 53-56.
22. Методи стиснення даних. Пристрій архіваторів, стиск зображень і відео / Д. Ватолін, А. Ратушняк, М. Смирнов та ін. – М. : ДІАЛОГ-МІФІ, 2002. – 384 с.
23. Сэломон, Д. Сжатие данных, изображений и звука / Д. Сэломон. – М. : Техносфера, 2004. – 368 с. – (Сер. «Мир программирования»).
24. Жураковсъкий, Ю.П. Теорія інформації та кодування / Ю.П. Жураковсъкий, В.П. Полторак. – К. : Вища школа, 2001. – 255 с.
25. Лидовский, В. В. Теория информации : учеб. пос. / В.В. Лидовский. – М. : Компания Спутник+, 2004. – 112 с.
26. Локазюк, В.М. Надійність, контроль, діагностика і модернізація ПК / В.М. Локазюк, Ю.Г. Савченко. – Київ : Академія, 2004. – 375 с.
27. Подлевський, Б.М. Теорія інформації / Б.М. Подлевський, Р.Є. Рикалюк. – Львів : ЛНУ ім. І. Франка, 2016. – 339 с.
28. Туликова, Н.О. Теорія інформації: навч. посіб. / Н.О. Туликова. – Суми : СумДУ, 2008. – 212 с.
29. Шульгин, В. И. Основы теории передачи информации : учеб. пос. / В.И. Шульгин – Харьков : НАУ ХАИ, 2003. – 102 с.
30. Machado, A.W. The Nimbus Cipher [Електронний ресурс] / A.W. Machado // New European Schemes for Signatures, Integrity, and Encryption, 2000. – P. 7. – Режим доступу: http://www.cosic.esat.kuleuven.be/nessie/workshop/submissions/nimbus.zip.
31. Сокирук, В.В. Побудова статистично безпечного БСШ на основі арифметичних операцій за модулем / В.В. Сокирук, В.А. Лужецький // Інформаційні технології та комп'ютерна інженерія. – 2006. – № 1. – С. 158-163.

32.Допоміжна

33. Куклин, А.Е. Шифрование и дешифрование текстовых файлов методом XOR-шифрования / А.Е. Куклин // Материалы МСНК «Студенческий научный форум 2023». – 2021. – № 7. – С. 70-72.
34. Безпека інформаційних систем і технологій / В.І. Єсін, О.О. Кузнецов, Л.С. Сорока. – Х. : ХНУ ім. В. Н. Каразіна, 2013. – 632 с.

35. Горбенко, І.Д. Захист інформації в інформаційно-телекомунікаційних системах. Криптографічний захист інформації / І.Д. Горбенко, Т.О. Гріненко. – Х. : ХНУР, 2004. – 368 с.
36. Грайворонський, М.В. Безпека інформаційно-комунікаційних систем / М.В. Грайворонський, О.М. Новіков. – К. : BHV, 2009. – 608 с.
37. Захист інформації в мережах передачі даних / О.К. Юдін, О.Г. Корченко, Г.Ф. Конахович. – К. : НВП ІНТЕРСЕРВІС, 2009. – 716 с.
38. Защита информации в компьютерных системах и сетях / Ю.В. Романец, П.А. Тимофеев, В.Ф. Шаньгин. – М. : Радио и связь, 2001. – 376 с.
39. Граничные оценки числа рёбер GL-моделей поведения отказоустойчивых многопроцессорных систем в потоке отказов / А.М. Романкевич, В.А. Романкевич, И.В. Майданюк // Электронное моделирование. – 2008. – Т. 30, № 1. – С. 59–70.
40. Корченко, О.Г. Охорона конфіденційної інформації підприємства / О.Г. Корченко, Ю.О. Дрейс. – Житомир : ЖВІ НАУ, 2011. – 172 с.
41. Математичні основи криптографії : навч. посіб. / Г.В. Кузнецов, В.В. Фомичов, С.О. Сушко та ін. – Дніпропетровськ : Нац. гірн. ун-т, 2004. – 391 с.
42. Математичні основи криптоаналізу / С.О. Сушко, Г.В. Кузнецов, Л.Я. Фомичова та ін. – Дніпропетровськ : Нац. гірн. ун-т, 2010. – 465 с.
43. Панасенко, С.П. Алгоритмы шифрования. Специальный справочник. – СПб. : БХВ-Петербург, 2009. – 576 с.
44. Петров, А.А. Компьютерная безопасность. Криптографические методы защиты. – М. : ДМК, 2000. – 448 с.
45. Богуш, В.М. Криптографічні застосування елементарної теорії чисел : навч. посіб. / В.М. Богуш, В.А. Мухачов. – К. : ДУІКТ, 2006. – 126 с. Режим доступу: https://dut.edu.ua/uploads/l_605_40821442.pdf
46. Горбенко, І.Д. Прикладна криптологія. Теорія. Практика. Застосування : монографія / І. Д. Горбенко, Ю. І. Горбенко. – Харків : Форт, 2012. – 880 с.
47. Методи та алгоритми симетричної криптографії / О.О. Кузнецов, С.П. Євсеев, О.А. Смірнов та ін. – Кіровоград : КНТУ, 2012. – 316 с.
48. Стеганографія / О.О. Кузнецов, С.П. Євсеєв, О.Г. Король. – Х. : ХНЕУ, 2011. – 232 с.

49.Методична

50. Методичні вказівки до виконання лабораторних робіт з дисципліни «Безпека програм та даних» для студентів спеціальностей 121 Інженерія програмного забезпечення, 122 Комп'ютерні науки та інформаційні технології, 123 Комп'ютерна інженерія [Електронний ресурс] / уклад. : Н.О.Маслова, Т.В.Скрипник. – Покровськ : ДонНТУ, 2017. − 60 с
51. Методичні вказівки до практичних занять з дисципліни «Основи інформаційної безпеки» для студентів спеціальності «Кібербезпека» денної форми навчання [Електронний ресурс] / уклад. : Н.О.Маслова, О.І.Патрушева. – Покровськ : ДонНТУ, 2018. – 60 с.

52. Методичні вказівки для самостійної роботи з дисципліни «Основи інформаційної безпеки» для студентів спеціальності «Кібербезпека» денної форми навчання [Електронний ресурс] / уклад. : Н.О.Маслова, О.І.Патрушева. – Покровськ : ДонНТУ, 2019. – 49 с.

ДОДАТОК А. Вимоги до оформлення звітів

Основний текст оформлюється у наступному стилі:

- шрифт Times New Roman текстового редактора Word;
- кегль – 14 п.;
- полуторний міжрядковий інтервал;
- вирівнювання по ширині;
- абзацний відступ 1см.

В якості графічних матеріалів можуть бути застосовані копії екранів, що підтверджують працездатність розробок та наявність результатів, ілюстрації, таблиці (якщо є необхідність розміщення не повної талиці, а її частки, скрину), графіки, діаграми тощо.

Оформлення графічного матеріалу: по центру, без абзацного відступу; назва – також по центру, без абзацного відступу. Графічний матеріал відокремлюється від основного тексту роботи порожнім рядком – до та після назви рисунку. На графічний матеріал слід робити посилання (наприклад, «на рисунку 2 відображено…»).

Таблицю, як і ілюстрацію, слід розташовувати безпосередньо після тексту, у якому вона згадується вперше або на наступній сторінці. На всі таблиці мають бути посилання в тексті звіту.

Таблиці нумеруються арабськими цифрами порядковою нумерацією в межах звіту. Таблиця повинна мати назву, яку друкують малими літерами (крім першої великої) і розміщують над таблицею. Назва має бути стислою і відбивати зміст таблиці.

Таблицю з великою кількістю рядків можна переносити на наступну сторінку. При перенесенні таблиці на наступну сторінку назву вміщують тільки над її першою частиною. Слово “Таблиця __” вказують один раз зліва над першою частиною таблиці, над іншими частинами пишуть: « Продовження таблиці __». В таблицях допустимо зменшення розміну шрифту та встановлення однарного міжрядкового інтервалу.

Частини програмного коду надаються з поясненнями й у наступному стилі: шрифт Times New Roman, кегль – 10 п.; одинарний міжрядковий інтервал; вирівнювання вліво; абзацний відступ 1см.

Вимоги до викладення матеріалів звіту з виконання лабораторних робіт. У звіті до роботи повинно бути наведено.

1. Титул
2. Мета роботи

3 Головна частина з описом об’єкту дослідження та оглядом сфери застосування.

4. Покроковий опис виконання завдань з копіями екранів, що підтверджують отримання результатів.
5. Висновки
6. Перелік застосованих джерел

ДОДАТОК Б. Приклад оформлення титульного листа

ДВНЗ «ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ»
Факультет комп'ютерно-інформаційних технологій та автоматизації
Кафедра прикладної математики і інформатики

ЗВІТ З ВИКОНАННЯ ПРАКТИЧНОЇ РОБОТИ

з дисципліни «Технології захисту інформації»
за темою:
«_____»

Виконав:

Студент (-тка) гр. _____

_____ _____

(дата, підпис) (ПІБ)

Перевірив:

_____ _____

(дата, підпис) (ПІБ)

Луцьк, 20_____