МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет «Запорізька політехніка»

МЕТОДИЧНІ ВКАЗІВКИ

з виконання лабораторних робіт дисципліни

«Основи технічного програмування»

для студентів спеціальності
G3 (141) – Електроенергетика, електротехніка та електромеханіка
усіх форм навчання

2026

Методичні вказівки з виконання лабораторних робіт дисципліни «Основи технічного програмування» для студентів спеціальності G3 (141) – Електроенергетика, електротехніка та електромеханіка усіх форм навчання. /Укл: В.О. Волков – м.Запоріжжя: НУ «Запорізька політехніка», 2026. – 80 с.

Укладачі:
В.О. Волков, к.т.н., доцент

Рецензент: А.В. Пирожок, к.т.н., доцент

Відповідальний за випуск: В.О. Волков, к.т.н., доцент

Затверджено
на засіданні кафедри
Електропривода і автоматизації
промислових установок
протокол № 6 від 10.03.2026 р.

Рекомендовано
до видання НМК ЕТФ
протокол № ___ від __.__.2026 р.

ЗМІСТ

ПЕРЕДМОВА

Методичні вказівки містять опис восьми лабораторних робіт з дисципліни «Основи технічного програмування» у відповідності до навчальних планів ОКР бакалаврів спеціальності G3 (141) – Електроенергетика, електротехніка та електромеханіка і рекомендації до їх виконання.

Лабораторні роботи містять короткі теоретичні відомості згідно теми роботи, завдання, рекомендації щодо їх виконання і контрольні запитання для кращого засвоєння матеріалу і перевірки отриманих студентом знань та навичок.

Лабораторна робота №1

КОДУВАННЯ. АРФИМЕТИЧНІ ОПЕРАЦІЇ З БІНАРНИМ КОДОМ

Мета роботи: розбір та опанування способів кодування інформації в різних системах числення, та виконання арифметичних операцій.

1.1 Короткі теоретичні відомості.

1.1.1 Двоїчні числа

Цифрові обчичслювальні машини працюють з двоїчними числами. Двоїчна система числення або система з основою "2" використовує тільки цифри "0" й "1". Ці двоїчні числа звуться **бітами**. Фізично в цифрових електроних системах біт "0" являє собою нізький (Low) рівень напруги, а біт "1" – високий (High) рівень напруги [2, 3].

Люди в загальному випадку використовуються дисятичну систему, або систему з основою "10", що складаэться з десяти фицр від 0 до 9. Вона також характеризується значенням (вагою) позиції згідно до табл. 1.1.

Приклад 1. Запис числа 1327.

Таблиця 1.1 – Значення позицій десятичних чисел

Ступінь основи 10	10^3	10^2	10^1	10^0
Значення позиції	1000	100	10	1
Дестичні	1	3	2	7

З табл. 1.1 видно, що десятичне число можна розкласти на складові частини, що відповідають кожній ступені основи:

$$1327 = 1*1000+3*100+2*10+7*1. \quad (1.1)$$

Аналагічно розглянемо приклад запису двоичного числа.

Приклад 2. Запис двоїчного числа 1001_2 в десятичній системи счислення

Таблиця 1.2 – Значення позицій двоїчних чисел

Ступінь основи 2	2^3	2^2	2^1	2^0
Значення позиції	8	4	2	1
Двоїчні	СБ 1	0	0	МБ 1
Дестичні	8	0	0	1

$$1001_2 = 8 + 0 + 0 + 1 = 9_{10}. \quad (1.2)$$

Тобто двоїчному числу 1001_2 відповідає число 9_{10} в десятичній формі. СБ – старший біт двоїяного числа. МБ – молодший біт двоїчного числа.

Таблиця 1.3 – Перевід десятичних чисел від 0 до 15 в їх двоїчні еквіваленти

Десятичне		Двоїчне				Десятичне		Двоїчне			
0	0	0	0	0	0		8	1	0	0	0
0	1	0	0	0	1		9	1	0	0	1
0	2	0	0	1	0	1	0	1	0	1	0
0	3	0	0	1	1	1	1	1	0	1	1
0	4	0	1	0	0	1	2	1	1	0	0
0	5	0	1	0	1	1	3	1	1	0	1
0	6	0	1	1	0	1	4	1	1	1	0
0	7	0	1	1	1	1	5	1	1	1	1

Число 1001_2 читається як – один нуль нуль один. Основа системи числення вказується індексами.

Приклад 3. Перевід двоїчного числа 10110110_2 в десятичну форму.

Таблиця 1.4 – Значення позицій двоїчних чисел

Ступінь основи 2	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Значення позиції	128	64	32	16	8	4	2	1
Двоїчні	1	0	1	1	0	1	1	0
Дестичні	128	0	32	16	0	4	2	0

$$10110110_2 = 128 + 0 + 32 + 16 + 0 + 4 + 2 + 0 = 182_{10}\,. \qquad (1.3)$$

Приклад 4. Перевід десятичного числа 155_{10} в двоїчну форму методом ділення.

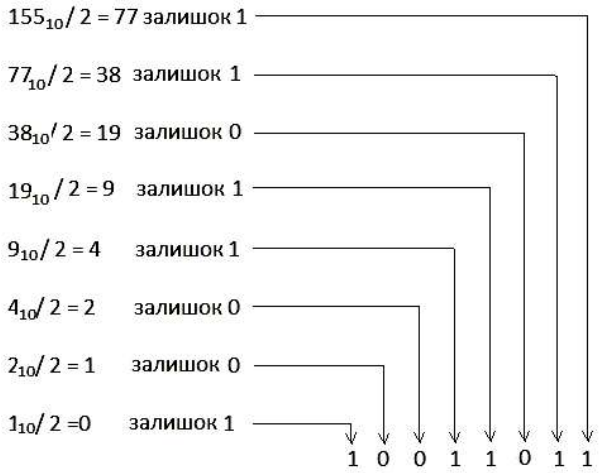


Рисунок 1.1 – Схема перетворення десятичного числа в двоїчний код

Тобто десятичному числу 155_{10} відповідає 10011011_2 двоїчний код.

1.1.2 Шестандцатеричні числа

Шестандцатерічна система числення або система з основною "16", використовує 16 символів від "0" до "9" й A, B, C, D, E, F.

З табл. 1.5 видно, що кожний шістнадцятерічний символ можить бути представлено одним сполученням четирьох біт. Тому для того щоб перетворити двоїчне число 111010_2 в шістнадцятеричне потрібно починаючі з молодшого біту (МБ) розділити двоїчне число на групи з 4 біт: $111010_2 \rightarrow 0011\ 1010_2 \rightarrow$ де 0011_2 – це 3_{16}, а 1010_2 – це A_{16} (зідно табл. 1.5) → замінемо відповідні тетраїди (чотирибітні коди) в початковому виді й отримаємо число в шістнадцятерічної формі: $3A_{16}$.

Для того, щоб перетворити шістандцяторичне число $7F_{16}$ перетворити в двоїчне, потрібно кожну шістнадцяторичну цифру або літеру замінити на свій двоїчний еквівалент згідно табл. 1.5.

$7F_{16}\rightarrow$ згідно табл. 1.5: $7_{16} = 0111_2$; $F_{16} = 1111_2 \rightarrow$ отримаємо 01111111_2

Таблиця 1.5 – Перевід десятичних чисел від 0 до 15 в їх двоїчні еквіваленти

10-е	16-е	Двоїчне				10-е	16-е	Двоїчне			
		8	4	2	1			8	4	2	1
0	0	0	0	0	0	8	8	1	0	0	0
1	1	0	0	0	1	9	9	1	0	0	1
2	2	0	0	1	0	10	A	1	0	1	0
3	3	0	0	1	1	11	B	1	0	1	1
4	4	0	1	0	0	12	C	1	1	0	0
5	5	0	1	0	1	13	D	1	1	0	1
6	6	0	1	1	0	14	E	1	1	1	0
7	7	0	1	1	1	15	F	1	1	1	1

Приклад 5. Перетворення шістнадцятеричного числа $2C6E_{16}$ в десятичне.

Таблиця 1.6 – Значення позицій двоїчних чисел

Ступінь основи 16	16^3	16^2	16^1	16^0
Значення позиції	4096	256	16	1
Шістнадцятерічні	2	C	6	E
Дестичні	8192	3072	96	14

$$2C6E_{16} = 2*4096 + C*256 + 6*16 + E*1 = 2*4096 + 12*256 + 6*16 + + 14*1 = 8192 + 3072 + 96 + 14 = 11374_{10} \quad (1.4)$$

Приклад 6. Перетворення десятичного числа 15797_{10} в шістнадцятеричне.

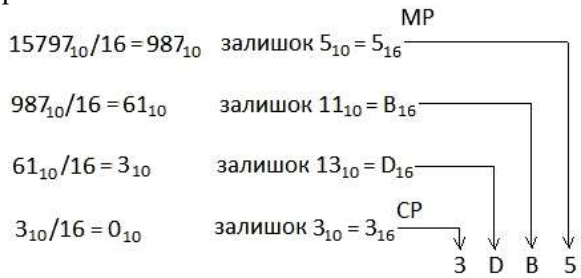


Рисунок 1.2 – Схема перетворення десятичного числа в шістнадцятеричний код

Тобто десятичному числу 15797_{10} відповідає $3DB5_{16}$ шістнадцятеричний код, МР – молодший розряд, СР – старший розряд.

1.1.3 Восьмирічні числа

Восьмирічна система містить 8 цифр від "0" до "7" й є відповідно системою з основою "8".

Таблиця 1.7 – Десятичні, восьмирічні та двоїчні еквіваленти

Десятичні	Восьмирічні	Двоїчні		
		4	2	1
0	0	0	0	0
1	1	0	0	1
2	2	0	1	0
3	3	0	1	1
4	4	1	0	0
5	5	1	0	1
6	6	1	1	0
7	7	1	1	1

При перетворенні двоїчного числа 11111000100_2 в його восьмирічній еквівалент, починаючі з молодшого біту (МБ) двоїчного числа, ділимо його на групи з трьох біт. Потім, застосовуючі табл. 1.7, перетворюєму кожну тріаду (групу з трьох біт) в еквівалентну восьмирічну цифру.

Двоїчне число: $011\ 111\ 000\ 100_2$.
Восьмирічне число: $3\quad 7\quad 0\quad 4_8$.

Тобто двоїчному числу 11111000100_2 відповідає 3704_8 восьмирічний код.

При перетворенні восьмирічного числа 6521_8 в його двоїчний еквівалент. Кожну восьмирічну цифру замінюємо двоїчною тріадою згідно табл. 1.7 й отримаємо:

$$6521_8 = 110\ \ 101\ \ 010\ \ 001_2 . \qquad (1.5)$$

Приклад 7. Перетворення восьмирічного числа 2357_8 в десятичне.

Таблиця 1.8 – Значення позицій двоїчних чисел

Ступінь основи 16	8^3	8^2	8^1	8^0
Значення позиції	512	64	8	1
Восмирічні	2	3	5	7
Десятичні	1024	192	40	7

$$2357_8 = 2*512 + 3*64 + 5*8 + 7*1 = 1024 + 192 + 40 + 7 = 1263_{10}. \quad (1.6)$$

Приклад 8. Перетворення десятичного числа 3336_{10} в його восьмирічний еквівалент.

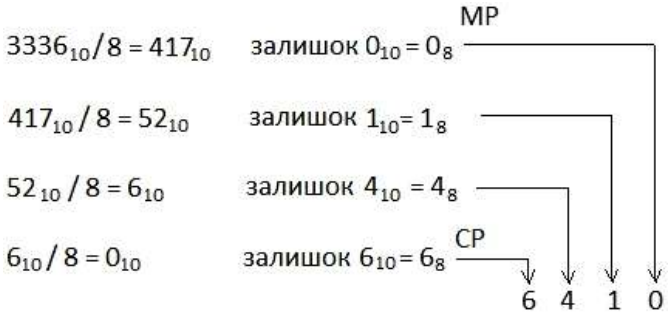


Рисунок 1.3 – Схема перетворення десятичного числа в восьмирічний код

Тобто десятичному числу 3336_{10} відповідає 6410_8 восьмирічний код.

Більшість мікропроцесорів опрацьовують групи з 4, 8 та 16 біт. З цього слідкує, що за звичай частіше використовується шістнадцятиричний запис числа, ніж восьмирічний. Восьмиричний запис більш зручний, коли група біт ділеться на три, наприклад групи з 12 біт.

1.1.4 Арифметичні операції з двоїчними числами

Розглянемо основні арифметичні операції, правила та приклади застосування наведені на рис. 1.4 – рис. 1.9.

1) Додавання.

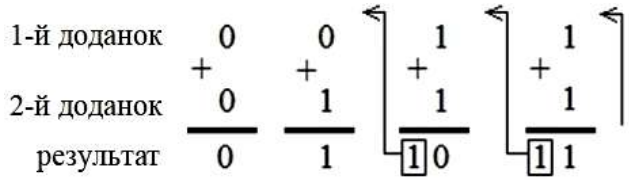


Рисунок 1.4 – Правило побітового додавання

+	0	0	1	1	$= 3_{10}$
	0	1	0	1	$= 5_{10}$
	1	0	0	0	$= 8_{10}$

Рисунок 1.5 – Приклад побітового додавання

2) Віднімання.

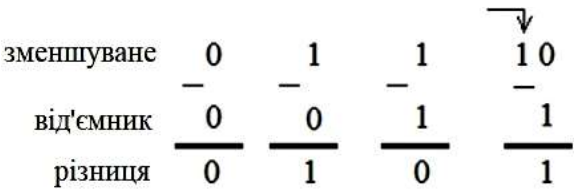


Рисунок 1.6 – Правило побітового віднімання

−	0	1	1	1	$= 7_{10}$
	0	0	1	0	$= 2_{10}$
	0	1	0	1	$= 5_{10}$

Рисунок 1.7 – Приклад побітового відніманя

3) Добуток.

множене	0	1	0	1
	*	*	*	*
множник	0	0	1	1
добуток	0	0	0	1

Рисунок 1.8 – Правило побітового добутка

				1	1	0	1	$= 13_{10}$
*					1	0	1	$= 5_{10}$
				1	1	0	1	
+			0	0	0	0		
		1	1	0	1			
	1	0	0	0	0	0	1	$= 65_{10}$

Рисунок 1.9 – Приклад побітового добутка

1.1.5 Додатковий код

Коли виникає необхідність використання числа зі знаком, використовують спеціальний додатковий код.

На рис. 1.10, а наведено звичайне зображення регистру МП або клітинки пам'яті зовні мікропроцесора. Такий регістр зображують простіром з 8 біт даних.

Позиції бітів пронумеровано від "7" до "0", а вага двійкових позицій вказана в основі регистра, біт 7 має вагу 128, біт 6 – 64 й т.д.

На рис. 1.10, б, в зображуються типові структури 8-ми розрядних регістрів для розміщення чисел зі знаком. В обох випадках біт 7 є знаковим бітом. Він вказує, чи є число додатнім (+) або від'ємним (–). При "0" в знаковому біті число додатнє, при "1" – від'ємне.

Приклади перетворення десятичного числа із знаком у додатковий код та зворотнтого перетворення наведені відповідно на рис. 1.11 та рис. 1.12.

а – розташування двійкових позицій; б – ідентифікація додатніх чисел нулем в знаковому біті; в – ідентифікація від'ємних чисел одиницею в знаковому біті.

Рисунок 1.10 – Зображення регістру мікропроцесора або клітинки пам'яті

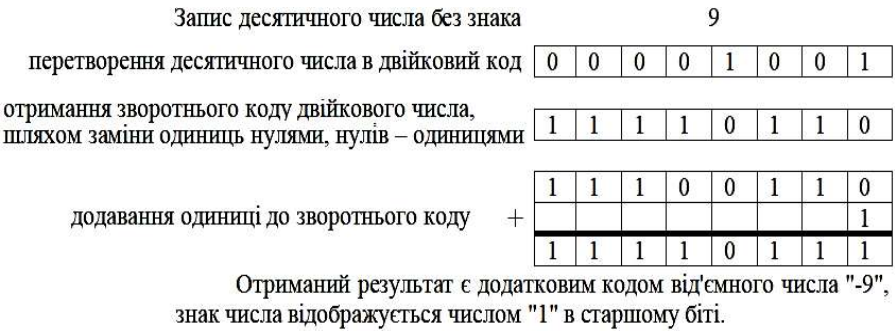


Рисунок 1.11 – Приклад перетворення десятичного від'ємного числа «-9» в додатковий код

Рисунок 1.12 – Приклад зворотного перетворення додаткового коду до десятичного числа з урахуванням знаку

Таблиця 1.9 – Значення позицій додаткового коду

Десятичні	Додатковий код	Примітка	Десятичні	Додатковий код	Примітка
+127	0111 1111	Додатні числа, зображуються в тій же формі, що й звичайні двійкові числа	-1	1111 1111	Від'ємні числа, що зображені в формі додаткового коду
...	...		-2	1111 1110	
+8	0000 1000		-3	1111 1101	
+7	0000 0111		-4	1111 1100	
+6	0000 0110		-5	1111 1011	
+5	0000 0101		-6	1111 1010	
+4	0000 0100		-7	1111 1001	
+3	0000 0011		-8	1111 1000	
+2	0000 0010		...	...	
+1	0000 0001				
+0	0000 0000		-128	1000 0000	

Отриманий результат на рис. 1.12 двійкового відповідає десятичному числу 16, однак потрібно пом'ятати, що у віхідному запису додаткового коду в старшому біті була одиниця, тому отриманому результату відповідає десятичний еквівалент "-16".

Приклади перетворення десятичних числе зі знаком в додатковий код наведені в табл. 1.9.

1.1.6 Арифметика в додатковому коді

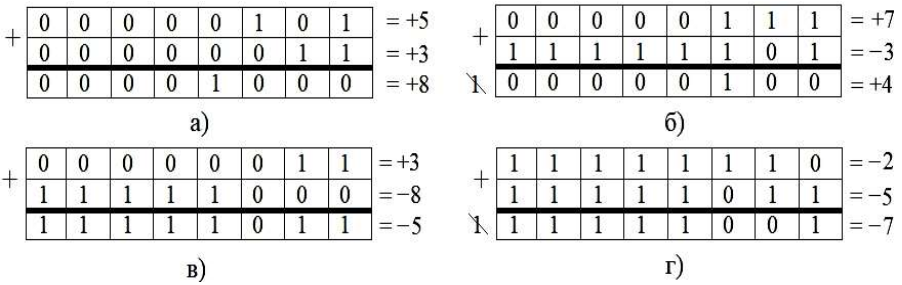
Приклади застування операції додавання в додатковому коді наведені на рис. 1.13.

Старші біти (біт № 9) на рис. 1.13, б,г є переповненням 8-ми розрядного регістру, тому вони викреслюється.

1.1.7 Алфавітно-цифровий код ASCII

Коли виконується взаємодія з відеотерміналом виникає необхідність у використанні коду, який одночасно включає в собі числові й алфавітні знаки. Такі коди мають назву – алфовітно-цифрові.

Найбільш розповсюдженим алфавітно-цифровим кодом є код ASCII – це стандартний американський код обміну інформації. В табл. 1.10 наведено декілько значень 7-и розрядного кода ASCII.



а – вираз: 5 + 5; б – вираз: 7 – 3; в – вираз: 3 – 8; г – вираз: – 2 – 5.

Рисунок 1.13 – Приклади операції додавання в додатковому коді

1.2 Порядок виконання лабораторної роботи № 1

1) Виконуємо обчисленя виразів за своїм номером варіанта з табл. 1.11. Для отриманого результуючого значення у двоїчній системі числення потрібно знайти відповідні значення у восьмирічній, шестандцатирічній та десятичній системі числення.

2) Своє повне ПІБ потрібно перевести (задопомогою коду ASCII, табл. 1.10) у двоїчний, восьмирічній та шестандцатирічній вид.

3) Оформляємо звіт.

Таблиця 1.10 – Деякі значення символів з алфавітно-цифрового коду ASCII

Сим-вол	10-й код	2-й код	Сим-вол	10-й код	2-й код	Сим-вол	10-й код	2-й код	Сим-вол	10-й код	2-й код
	32	00100000	8	56	00111000	P	80	01010000	h	104	01101000
!	33	00100001	9	57	00111001	Q	81	01010001	i	105	01101001
“	34	00100010	:	58	00111010	R	82	01010010	j	106	01101010
#	35	00100011	;	59	00111011	S	83	01010011	k	107	01101011
$	36	00100100	<	60	00111100	T	84	01010100	l	108	01101100
%	37	00100101	=	61	00111101	U	85	01010101	m	109	01101101
&	38	00100110	>	62	00111110	V	86	01010110	n	110	01101110
‘	39	00100111	?	63	00111111	W	87	01010111	o	111	01101111
(	40	00101000	@	64	01000000	X	88	01011000	p	112	01110000
)	41	00101001	A	65	01000001	Y	89	01011001	q	113	01110001
*	42	00101010	B	66	01000010	Z	90	01011010	r	114	01110010
+	43	00101011	C	67	01000011	[	91	01011011	s	115	01110011
,	44	00101100	D	68	01000100	\	92	01011100	t	116	01110100
-	45	00101101	E	69	01000101	]	93	01011101	u	117	01110101
.	46	00101110	F	70	01000110	^	94	01011110	v	118	01110110
/	47	00101111	G	71	01000111	_	95	01011111	w	119	01110111
0	48	00110000	H	72	01001000	`	96	01100000	x	120	01111000
1	49	00110001	I	73	01001001	a	97	01100001	y	121	01111001
2	50	00110010	J	74	01001010	b	98	01100010	z	122	01111010
3	51	00110011	K	75	01001011	c	99	01100011	{	123	01111011
4	52	00110100	L	76	01001100	d	100	01100100	|	124	01111100
5	53	00110101	M	77	01001101	e	101	01100101	}	125	01111101
6	54	00110110	N	78	01001110	f	102	01100110	~	126	01111110
7	55	00110111	O	89	01001111	g	103	01100111	□	127	01111111

Таблиця 1.11 – Індивідуальні завдання

№ варіанта	Вирази
1	$01101_2 + (5FC_{16}+24_8)*5_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $-45 + 80 - 16$
2	$01110_2 + (AFC_{16}+27_8)*15_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $54 + 36 - 50$
3	$101001_2 + (3BA_{16}+134_8)*35_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $100 - 16 - 42$
4	$01001_2 + (2FF_{16}+35_8)*8_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $120 - 40 - 86$
5	$1101101_2 + (DFB_{16}+54_8)*9_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $56 - 28 - 76$
6	$1001101_2 + (BC6_{16}+15_8)*25_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $-64 - 32 + 52$
7	$011011_2 + (3FA_{16}+11_8)*11_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $- 52 - 38 - 46$
8	$01111_2 + (5FC_{16}+24_{10})*5_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $-16 - 15 - 32$
9	$101101_2 + (8FC_{16}+56_{10})*3_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $15 - 16 - 45$
10	$1101101_2 + (9FB_{16}+38_{10})*2_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $- 24 + 46 - 80$
11	$0101101_2 + (6DC_{16}+42_{10})*7_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $- 80 + 32 + 16$
12	$01101101_2 + (3DD_{16}+25_{10})*6_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $- 24 - 46 + 68$
13	$1101101_2 + (4AF_{16}+12_{10})*5_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $- 45 - 12 - 18$
14	$1001001_2 + (A2F_{16}+54_{10})*3_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $- 63 + 35 - 85$
15	$1000101_2 + (BF2_{16}+76_{10})*4_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $-124 + 80 + 24$
16	$1101101_2 + (DFB_{16}+54_8)*9_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $124 - 65 - 32$
17	$BC6_{16} + (1001101_2+15_8)*25_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ $-14 - 16 - 24$

Продовження таблиці 1.11

№ варіанта	Вирази
18	$3FA_{16}+ (011011_2+11_8)*11_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ – 56 – 12 + 36
19	$5FC_{16} + (01111_2+24_{10})*15_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ – 82 + 56 – 38
20	$8FC_{16} + (101101_2+56_{10})*32_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ – 55 – 25 + 35
21	$9FB_{16} + (1101101_2 +38_{10})*27_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ – 54 – 27 + 38
22	$6DC_{16} + (0101101_2+42_{10})*71_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ –78 +38 – 50
23	$3DD_{16} + (01101101_2+25_{10})*63_8= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ –92 + 28 – 32
24	$24_8 + (5FC_{16} + 01101_2)*51_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ – 38 – 62 + 44
25	$27_8+ (AFC_{16}+01110_2)*45_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ 44 + 15 – 76
26	$134_8 + (3BA_{16}+101001_2)*37_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ 45 + 92 – 26
27	$35_8 + (2FF_{16}+01001_2)*18_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ 47 + 25 – 16
28	$54_8 + (DFB_{16}+1101101_2)*13_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ 52 – 68 + 76
29	$15_8 + (BC6_{16}+1001101_2)*28_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ 92 – 104 + 23
30	$24_8 + (5FC_{16}+01101_2)*12_{10}= (\ )_2 = (\ )_8 = (\ )_{16} = (\ )_{10}$ 117 – 125 + 92

Лабораторна робота №2

УМОВНИЙ ОПЕРАТОР «IF … ELSEIF … ELSE … END». ЦИКЛИ ТИПУ «FOR … END»

Мета роботи: розбір та опанування практичних навичок в застосуванні у своєму програмному коді умовного оператора «if … elseif … else … end» та циклів типу «for ... end».

2.1 Короткі теоретичні відомості

2.1.1 Умовний оператор «if … elseif … else … end»

Умовний оператор if в загальному вигляді записується наступним чином [2, 3]:

```
if  Умова_1
    Інструкція_1
elseif  Умова_2
    Інструкція_2
else
    Інструкція_3
end
```

В простішому випадку можна скористатися скороченим записом:

```
if  Умова
    Інструкція
end
```

Доки *Умова* повертає логічне значення «1» (тобто «Істина»), виконуються *Інструкції*, що складають тіло структури «if … end». При цьому оператор end показує на кінець переліку *Інструкцій.* Інструкції в переліку розділяються оператором «,» (кома) або «;» (крапка з комою). Якщо *Умова* не виконується (дає логічне значення «0», «Фальш»), то *Інструкції* також не виконуються.

Конструкція

```
if  Умова
    Інструкція_1
else
    Інструкція_2
end
```

працює наступним чином: якщо виконується *Умова* (повертається значення логічної «1»), – виконується *Інструкція_1*, якщо *Умова* не виконується (повертається значення логічного «0»), – виконуєтсья *Інструкція_2*.

Умова записується у вигляді:

Вираз_1 Оператор_відношення Вираз_2

причому в якості *Оператора_відношення* використовуються наступні оператори: == (дорівнює), < (менше), > (більше), <= (менше дорівнює), >= (більше дорівнює), ~= (не дорівнює).

2.1.2 Цикли типу «for ... end»

Цикли типу «for ... end» звичайно використовуються для організації обчислень з заданим числом циклів, що повторюються. Конструкція такого циклу має наступний вид:

```
for var = Вираз
    Інструкція_1;
        ...
    Інструкція_n;
end
```

Вираз найчастіше записується у вигляді: s : d : k, де s – початкове значення змінної циклу var, d – приріст цієї змінної та k – кінцеве значення змінної циклу var, при досягненні котрого цикл закінчується. Можливий скорочений запис: s : k (в цьому випадку d = 1). Список *інструкцій*, що повинні бути виконані завершується оператором end.

2.1.3 Приклади

Приклад 1. Зробити вивід на екран внутрішні елеменити матриці-рядка A з вказуванням порядкового номеру елемента у матриці.

```
A = [1, 2, 5, 6, 8, 7, 4, 9, 8, 3];
for i = 1 : length(A);
   fprintf('element number %d', i);
   fprintf(' : %d  \n', A(i));
end
```

Результат буде мати вигляд:

```
element number 1: 1
element number 2: 2
element number 3: 5
element number 4: 6
element number 5: 8
element number 6: 7
element number 7: 4
element number 8: 9
element number 9: 8
element number 10: 3
```

Приклад 2. Зробити вивід на екран позиції елементів тієї ж матриці-рядка A, де значення елементів менші ніж значення 5.

```
A = [1, 2, 5, 6, 8, 7, 4, 9, 8, 3];
 for i=1:length(A);
   if A(i) < 5
        fprintf('element number %d',i);
        fprintf(': %d \n', A(i));
   end
end
```

Результат буде мати вигляд:

```
element number 1: 1
element number 2: 2
element number 7: 4
element number 10: 3
```

2.2 Порядок виконання лабораторної роботи № 2

1) Написати программу, яка буде обробляти значення елементів деякої матриці та виводити на екран статистику в наступному вигляді:

Матриця-рядок містить: … елементів;
парних елементів: … ;
непарних елементів: …;
додатніх елементів: …;
від'ємних елементів: …;
максимальне значення елемента: … , його порядковий номер …;
мінімальне значення елемента: … , його порядковий номер ….

Значення елементів матриці наведено відповідно до номеру варіанта в табл. 2.1.

Таблиця 2.1 – Індивідуальні завдання

№ вар.	Значення елементів матриці
1	34, 90, 37, 11, -78, 39, 24, 40, -10, 13, 94, 96, 58, 6, 23, 35, 82, 2, 4, 17
2	65, 73, 65, -45, 55, 30, 74, 19, 69, 18, 37, 63, 78, 8, 93, 78, 49, 44, 45, 31
3	51, 51, 82, 79, 64, -38, 81, 53, 35, -94, 88, 55, 62, 59, 21, 30, 47, 23, 84, 19
4	23, 17, 23, 44, 31, 92, 43, -18, 90, 98, 44, 11, 26, 41, 59, 26, 60, 71, 22, 12
5	30, 32, -42, 51, 9, 26, 80, 3, 93, -73, -49, 58, 24, 46, 96, 55, 52, 23, 49, 62
6	68, 40, 37, 99, -4, 89, 91, 80, -10, 26, 34, 68, 14, 72, 11, 65, 49, 78, 72, 90
7	89, 33, 70, 20, 3, -74, 50, 48, 90, 61, 62, 86, -81, 58, 18, 24, 89, 3, 49, 17
8	98, 71, 50, 47, 6, -68, 4, 7, 52, -10, 82, -82, 72, 15, -66, 52, 97, 65, 80, 45
9	43, 83, 8, 13, -17, 39, -83, 80, -6, 40, 53, 42, 66, 63, 29, 43, 2, 98, 17, 11
10	37, 20, 49, 34, 95, -92, 5, 74, 27, 42, 55, 94, 42, 98, 30, 70, 67, 54, 70, 67
11	18, 13, 100, 17, 3, 56, 88, 67, 19, 37, -46, 98, 16, 86, 64, 38, 19, 43, 48, 12
12	59, 23, 38, 58, 25, 29, 62, -27, 82, 98, 73, 34, 58, 11, 91, 88, 82, 26, 59, 2
13	43, 31, 16, 18, -42, 9, -60, 47, 70, -70, 64, 3, 7, -32, 53, -65, 41, 82, 72, 97
14	53, 33, 11, 61, 78, 42, 9, 27, 15, 28, -44, 53, 46, -88, 52, 94, 64, 96, 24, 68
15	29, 67, 70, -7, 25, 22, -67, 84, 34, -78, 68, 1, -60, 39, 92, 0, -46, 42, 46, 77
16	32, 78, 47, 4, 18, -72, 47, -15, 34, 61, 19, 74, 24, 92, 27, 77, 19, 29, 9, 58
17	68, 55, 43, 64, 65, -68, 64, 95, 21, 71, 24, 12, 61, 45, 46, 66, 77, 35, 66, 42
18	84, 83, 26, 61, 58, 54, 87, -26, 32, 12, 94, 65, 48, 64, 54, 65, 54, 72, 52, 99
19	22, 11, 11, 6, 40, 45, 37, -76, 63, 77, 93, -97, 19, 14, -70, 9, 53, 53, 86, 48

Продовження таблиці 2.1

№ вар.	Значення елементів матриці
20	39, 67, 74, -52, 35, 15, 59, 26, 4, -75, 24, 44, 69, 36, 74, 39, 68, 70, 44, 2
21	33, 42, 27, 20, 82, 43, 89, 39, -77, 40, 81, 76, 38, 22, 79, 95, 33, 67, 44, 83
22	77, 17, 86, 99, 51, 88, 59, -15, 20, 41, 75, 83, 79, -32, 53, 9, 11, 14, 68, 50
23	19, 50, 15, 5, -85, 56, 93, 70, 58, 82, -88, 99, 0, 87, -61, 99, 53, 48, 80, 23
24	50, 90, 57, 85, 74, 59, -25, 67, 8, 63, 66, -73, 89, 98, 77, 58, 93, 58, 2, 12
25	86, 48, 84, 21, 55, 63, 3, -61, 36, 5, 49, -19, 12, 21, -15, 19, 4, 64, 28, 54
26	70, 50, 54, 45, -12, 49, 85, 87, -27, 21, 56, 64, 42, 21, 95, 8, 11, 14, 17, 62
27	57, 5, 93, -73, 74, 6, 86, -93, 98, 86, 79, -51, -18, 40, 13, 3, 94, 30, 30, 33
28	47, 65, 3, 84, -56, 85, 35, 45, 5, -18, 66, 33, 90, 12, 99, 54, 71, 100, 29, 41
29	46, 76, 82, 10, -18, 36, 6, 52, 34, 18, -21, 91, 68, 47, 91, 10, 75, 74, 56, 18
30	60, 30, 13, 21, 89, 7, -24, 5, 44, 1, 90, -20, 9, -31, 46, -10, -100, -33, 30, 6

3) Оформляємо звіт.

Лабораторна робота №3

ЦИКЛИ ТИПУ «WHILE … END». ПЕРЕМИКАЧ «SWITCH … CASE … END»

Мета роботи: розбір та опанування практичних навичок в застосуванні у своєму програмному коді циклів типу «while … end» та перемикача «switch … case … end».

3.1 Короткі теоретичні відомості.

3.1.1 Цикли типу «while … end»

Цикл типу «while … end» виконується до тих пір, доки виконується *Умова* [2, 3]:

```
while   Умова
     Інструкції.
        …
end
```

Дострокове завершення циклів реалізується задопомогою операторів «break» та «continue».

3.1.2 Перемикач «switch … case … end»

Для здійснення багаточисельного вибору використовується конструкція з перемикачем типу switch:

```
switch    Змінна
        case     Значення змінної_1
             Інструкція_1
        case     {Значення змінної_2, Значення змінної_3, …}
             Інструкція_2
             …
        case     Значення змінної_N
             Інструкція_N
        otherwise
             Інструкція_Error
end
```

3.1.3 Приклади

Приклад 1. Розробка коду, задомогою якого на екран будуть виводитись числа значення котрих не повинно бути вище 50, де кожне наступне число дорівнює сумі попереднього з поточною позицією.

```
i = 1;
tmp = i;

while (tmp < 50)

   fprintf('%d', tmp);
   fprintf(', ');
   i=i+1;
   tmp = tmp + i;

end
```

Результат буде мати вигляд:

1, 3, 6, 10, 15, 21, 28, 36, 45

Приклад 2. Записати текстом значення введеного числа, яке може бути задано з проміжку від 0 до 9.

```
number = 4;

switch number
  case {0}
      disp('Нуль')
  case {1}
      disp('Один')
   case {2}
      disp('Два')
   case {3}
      disp('Три')
   case {4}
      disp('Чотири')
```

```
  case {5}
      disp('П_ять')
  case {6}
      disp('Шість')
  case {7}
      disp('Сім')
  case {8}
      disp('Вісім')
  case {9}
      disp('Дев_ять')
  otherwise
      disp('Введене число за межами завдання')
end
```

Результат буде мати вигляд: Чотири.

3.2 Порядок виконання лабораторної роботи № 3

1) Написати программу, яка буде перетворювати деяке слово (що наведене відповідно до номеру варіанта у табл. 3.1) задомогою таблиці ASCII (табл. 3.2) у двоїчний код.

Приклад:

R2D2 = 1010010 0110010 1000100 0110010

Таблиця 3.1 – Індивідуальні завдання

№ вар.	Значення	№ вар.	Значення	№ вар.	Значення
1	electric	11	flux.linkage	21	valence
2	drive	12	force.energy	22	balance
3	motor	13	capacity	23	compensation
4	enjine	14	conductivity	24	efficiency
5	voltage	15	induction	25	battery
6	current	16	vector	26	windmill
7	resistanse	17	direction	27	transmission
8	frequency	18	pump	28	electron
9	distance	19	conveyor	29	thyristor
10	speed	20	reactor	30	transistor

Таблиця 3.2 – Деякі значення символів з коду ASCII

Сим-вол	10-й код	2-й код	Сим-вол	10-й код	2-й код	Сим-вол	10-й код	2-й код	Сим-вол	10-й код	2-й код
	32	00100000	8	56	00111000	P	80	01010000	h	104	01101000
!	33	00100001	9	57	00111001	Q	81	01010001	i	105	01101001
“	34	00100010	:	58	00111010	R	82	01010010	j	106	01101010
#	35	00100011	;	59	00111011	S	83	01010011	k	107	01101011
$	36	00100100	<	60	00111100	T	84	01010100	l	108	01101100
%	37	00100101	=	61	00111101	U	85	01010101	m	109	01101101
&	38	00100110	>	62	00111110	V	86	01010110	n	110	01101110
‘	39	00100111	?	63	00111111	W	87	01010111	o	111	01101111
(	40	00101000	@	64	01000000	X	88	01011000	p	112	01110000
)	41	00101001	A	65	01000001	Y	89	01011001	q	113	01110001
*	42	00101010	B	66	01000010	Z	90	01011010	r	114	01110010
+	43	00101011	C	67	01000011	[	91	01011011	s	115	01110011
,	44	00101100	D	68	01000100	\	92	01011100	t	116	01110100
-	45	00101101	E	69	01000101	]	93	01011101	u	117	01110101
.	46	00101110	F	70	01000110	^	94	01011110	v	118	01110110
/	47	00101111	G	71	01000111	_	95	01011111	w	119	01110111
0	48	00110000	H	72	01001000	`	96	01100000	x	120	01111000
1	49	00110001	I	73	01001001	a	97	01100001	y	121	01111001
2	50	00110010	J	74	01001010	b	98	01100010	z	122	01111010
3	51	00110011	K	75	01001011	c	99	01100011	{	123	01111011
4	52	00110100	L	76	01001100	d	100	01100100	|	124	01111100
5	53	00110101	M	77	01001101	e	101	01100101	}	125	01111101
6	54	00110110	N	78	01001110	f	102	01100110	~	126	01111110
7	55	00110111	O	89	01001111	g	103	01100111	□	127	01111111

2) Написати власноруч алогритм-програму, яка буде (використовуючи для цього цикл «while … end»):

- перетворювати матрицю-рядок (табл. 3.3) у матрицю стовбець;
- виводити значення деякої змінної Sum, котра повинна містити в собі суму парних чисел елементів вхідної матриці.

Таблиця 3.3 – Значення елементів матриці для другого завдання

№ вар.	Значення елементів матриці
1	29, 67, 70, 7, 25, 22, 67, 84, 34, 78, 68, 1, 60, 39, 92, 0, 46, 42, 46, 77
2	32, 78, 47, 4, 18, 72, 47, 15, 34, 61, 19, 74, 24, 92, 27, 77, 19, 29, 9, 58
3	68, 55, 43, 64, 65, 68, 64, 95, 21, 71, 24, 12, 61, 45, 46, 66, 77, 35, 66, 42

Продовження таблиці 3.3

№ вар.	Значення елементів матриці
4	84, 83, 26, 61, 58, 54, 87, 26, 32, 12, 94, 65, 48, 64, 54, 65, 54, 72, 52, 99
5	22, 11, 11, 6, 40, 45, 37, 76, 63, 77, 93, 97, 19, 14, 70, 9, 53, 53, 86, 48
6	39, 67, 74, 52, 35, 15, 59, 26, 4, 75, 24, 44, 69, 36, 74, 39, 68, 70, 44, 2
7	33, 42, 27, 20, 82, 43, 89, 39, 77, 40, 81, 76, 38, 22, 79, 95, 33, 67, 44, 83
8	77, 17, 86, 99, 51, 88, 59, 15, 20, 41, 75, 83, 79, 32, 53, 9, 11, 14, 68, 50
9	19, 50, 15, 5, 85, 56, 93, 70, 58, 82, 88, 99, 0, 87, 61, 99, 53, 48, 80, 23
10	50, 90, 57, 85, 74, 59, 25, 67, 8, 63, 66, 73, 89, 98, 77, 58, 93, 58, 2, 12
11	86, 48, 84, 21, 55, 63, 3, 61, 36, 5, 49, 19, 12, 21, 15, 19, 4, 64, 28, 54
12	70, 50, 54, 45, 12, 49, 85, 87, 27, 21, 56, 64, 42, 21, 95, 8, 11, 14, 17, 62
13	57, 5, 93, 73, 74, 6, 86, 93, 98, 86, 79, 51, 18, 40, 13, 3, 94, 30, 30, 33
14	47, 65, 3, 84, 56, 85, 35, 45, 5, 18, 66, 33, 90, 12, 99, 54, 71, 100, 29, 41
15	46, 76, 82, 10, 18, 36, 6, 52, 34, 18, 21, 91, 68, 47, 91, 10, 75, 74, 56, 18
16	34, 90, 37, 11, 78, 39, 24, 40, 10, 13, 94, 96, 58, 6, 23, 35, 82, 2, 4, 17
17	65, 73, 65, 45, 55, 30, 74, 19, 69, 18, 37, 63, 78, 8, 93, 78, 49, 44, 45, 31
18	51, 51, 82, 79, 64, 38, 81, 53, 35, 94, 88, 55, 62, 59, 21, 30, 47, 23, 84, 19
19	23, 17, 23, 44, 31, 92, 43, 18, 90, 98, 44, 11, 26, 41, 59, 26, 60, 71, 22, 12
20	30, 32, 42, 51, 9, 26, 80, 3, 93, 73, 49, 58, 24, 46, 96, 55, 52, 23, 49, 62
21	68, 40, 37, 99, 4, 89, 91, 80, 10, 26, 34, 68, 14, 72, 11, 65, 49, 78, 72, 90
22	89, 33, 70, 20, 3, 74, 50, 48, 90, 61, 62, 86, 81, 58, 18, 24, 89, 3, 49, 17
23	98, 71, 50, 47, 6, 68, 4, 7, 52, 10, 82, 82, 72, 15, 66, 52, 97, 65, 80, 45
24	43, 83, 8, 13, 17, 39, 83, 80, 6, 40, 53, 42, 66, 63, 29, 43, 2, 98, 17, 11
25	37, 20, 49, 34, 95, 92, 5, 74, 27, 42, 55, 94, 42, 98, 30, 70, 67, 54, 70, 67
26	18, 13, 100, 17, 3, 56, 88, 67, 19, 37, 46, 98, 16, 86, 64, 38, 19, 43, 48, 12
27	59, 23, 38, 58, 25, 29, 62, 27, 82, 98, 73, 34, 58, 11, 91, 88, 82, 26, 59, 2
28	43, 31, 16, 18, 42, 9, 60, 47, 70, 70, 64, 3, 7, 32, 53, 65, 41, 82, 72, 97
29	53, 33, 11, 61, 78, 42, 9, 27, 15, 28, 44, 53, 46, 88, 52, 94, 64, 96, 24, 68
30	34, 90, 37, 11, 78, 39, 24, 40, 10, 13, 94, 96, 58, 6, 23, 35, 82, 2, 4, 17

3) Оформляємо звіт.

Лабораторна робота №4

РЯДКОВІ ЗМІННІ

Мета роботи: розбір та опанування практичних навичок в застосуванні у своєму програмному коді рядкових змінних.

4.1 Короткі теоретичні відомості

До операцій над рядками зазвичай відносять пошук входження одних рядків в інші, заміну регістрів символів, об'єднання рядків тощо [3, 4].

Наступні функції здійснюють операції над рядками:

1) **findstr(str1,str2)** – забезпечує пошук початкових індексів коротшого рядка всередині довшого і повертає вектор цих індексів. Індекси вказують положення першого символу коротшого рядка в довшому рядку;

```
str1 = 'Example of the function Is the findstr function';
str2 = 'the';
k = findstr(strl,str2)
k =
    12   28
```

2) **lower('str')** – повертає рядок символів str, у якому символи верхнього регістру перетворюються на нижній регістр, проте інші символи залишаються без змін;

```
str = 'Example Of The Function';
t = lower(str)
t =
   example of the function
```

3) **upper('str')** – повертає рядок символів str, де всі символи нижнього регістру переводяться у верхній регістр, а решта символів залишаються без змін;

```
str = 'danger!';
t = upper(str)
t =
    DANGER!
```

4) **strcat(sl,s2,s3,...)** – виконує горизонтальне об'єднання відповідних рядів масивів символів s1, s2, s3 і т.д., причому прогалини в кінці кожного ряду відкидаються, і повертає об'єднаний рядок (ряд) результуючого масиву символів, пробіли додаються заново після аналізу рядків отриманому масиві. Всі вхідні масиви повинні мати однакову кількість рядків (у окремому випадку мають бути представлені у вигляді одного рядка символів), але якщо один із вхідних аргументів – не масив символів, а рядковий масив осередків, то будь-який з інших вхідних аргументів може бути скаляром або будь-яким масивом. Тієї ж розмірності і того ж розміру. Якщо вхідний масив складається лише з символів, вихідний масив також буде масивом символів. Якщо будь-який з вхідних масивів є рядковим масивом осередків, то функція strcat повертає рядковий масив осередків, сформований з відповідних об'єднаних елементів масивів s1, s2, s3, при цьому будь-який з елементів може бути скаляром і т.д;

```
s1 =  'home' :
s2 = 'work';
t = strcat(sl,s2)
t =
   'homework'
```

5) **strvcat(t1,t2,t3....)** – виконує вертикальне об'єднання рядків t1, t2, t3,.. в масив символів S;

```
t1 = 'string';
t2 = 'concatenation';
S  =  strvcat(t1,t2)
S =
     string
     concatenation
```

6) **strcmp(str1,str2)** – повертає логічну одиницю, якщо два порівнювані рядки str1 і str2 ідентичні, і логічний нуль якщо інакші;

```
str1 = 'home';
str2 = 'home';
result = strcmp(str1,str2)
result =
          1

str1 = 'home';
str2 = 'Home';
result = strcmp(str1,str2)
result =
          0
```

7) **strncmp(str1,str2,n)** – повертає логічну одиницю, якщо два порівнювані рядки strl і str2 містять n перших ідентичних символів, і логічний нуль інакше;

```
str1 = 'work';
str2 = 'world';
result = strncmp(str1,str2,3)
result =
          1
result = strncmp(str1,str2,4)
result =
          0
```

8) **strmatch(str,text)** – переглядає масив символів або рядковий масив осередків text по рядках, знаходить рядки символів, що починаються з рядка str, і повертає відповідні індекси рядків;

```
arrayString = strvcat('one', 'two', 'three', 'one', 'four', 'five')
arrayString =
              one
              two
              three
```

```
                one
                four
                five
str = 'one';
result = strmatch(str,arrayString)

result =
           1
           4
```

9) **strrep(text,before,after)** – замінює всі слова before знайдені в тексті text на нові значення after;

```
text  = 'This is a good example for me.';
before = 'good';
after = 'best';
newText = strrep(text,before,after)
newText =
              This is a best example for me.
```

10) **strtok(str, delimiter)** – повертає частину текстового рядка str до появи символу delimiter, за замовчуванням delimiter = ' ';

```
text  = 'This is a good example for me.';
[firstWorld, otherText] = strtok(text)
firstWorld =
                This
otherText =
                 is a good example for me.

result = strtok(text,'e')
result =
         This is a good
```

4.2 Порядок виконання лабораторної роботи № 4

1) Написати программу, яка буде обробляти деякий введений текст (згідно до номеру варіанта з табл. 4.1) та виводити на екран статистику в наступному вигляді:

Текст містить: … слів;
кількість речень: … ;
кількість прогалин (пропусків): … ;
кількість знаків без прогалин (пропусків): … ;
кількість твердих літер: … ;
кількість голосних літер: … ;
кількість маленьких літер: … ;
кількість великих літер: … .

2) Написати программу, котра буде обробляти деякий текст, та повертати текст де кожне слово у реченні буде з великої літери. Текст для завдання беріть той самий як и для першого завдання.

Таблиця 4.1 – Індивідуальні завдання для першого і другого завдання

№ вар.	Текст завдання
1	The electricity consumption in commercial places like universities has tremendously increased recently. Modern and advanced energy efficient appliances are highly needed to substitute the conventional ones. Energy saving is of great important instead of its wastage, as Utilizing the energy Efficiently reduces the cost of energy. Energy consumption varies for commercial building due to several factors such as electrical appliance usage, electrical appliance type, management, etc. Due to the advancement in Technology, there are new emergence appliances that are of high efficiency and have less energy consumption.

Продовження таблиці 4.1

№ вар.	Текст завдання
2	A case study is conducted on selected five tutorial rooms, level 4 buildings in the Faculty of Electrical Engineering 19 A, Universiti Teknologi Malaysia. The paper proposes new emergence equipments with high efficiency and less power consumption to replace the existing ones. A survey is conducted on the number of electrical appliances used for each of the tutorial rooms, time table for each tutorial room and the Tenaga Nasional Berhad pricing and tariff are taken into consideration in the analysis of the energy consumption and the cost of energy.
3	This paper aims at reducing the amount of energy consumption by replacing the existing electrical equipments with high efficient electrical equipments; it also tends to reduce the cost of energy paid to the utility. By observing the results, it shows that the proposed efficient electrical equipments are more efficient, less power consumption and less cost compared to the existing electrical equipments.
4	However, an efficient Management of Electrical energy Regulatory was introduced on 15 December, 2008 for the purpose of promoting energy efficiency in Malaysia. Thus, they make it compulsory for large commercial and industrial electrical consumers to manage their equipments so as to develop and implement EEMs to reduce energy losses, cost of energy and enforce efficient utilization of electrical energy [3].
5	Due to the advancement of technology, researchers and engineers are always working to see that they produce an apparatus or equipments which are very efficient in terms of energy. Energy efficiency can be defined as using less energy to produce the same amount of services or useful output, for example, residential sector, commercial sector and industrial sector. Energy efficiency in terms of mathematical expression can be defined as the ratio of the useful output of a process and the energy input into a process, and it is expressed in percentage.

Продовження таблиці 4.1

№ вар.	Текст завдання
6	The factors that contribute to high energy usage can be grouped into three. Firstly, electricity consumption of the equipments itself. That is the purchase of fairly used equipment and non-energy efficient equipments. Secondly, the number of equipments used for a particular place. As it is known that the number of equipments is directly proportion to the energy consumption. The used of many numbers of equipments such as fans, lights, air conditioner etc. than the design requirements.
7	Tutorial Room 1 to 5 located in block P19 at the Faculty of Electrical Engineering (FKE), UTM was selected as the case study area. The classrooms selected are used as a lecture theatre for graduate studies. Electrical wattage and quantity of each of the equipments were studied and recorded. We are able to count and record all the equipments from tutorial Room 1 to 5. Since the tutorial rooms have the same electrical equipments, therefore Table 1 shows the quantity of electrical equipments of one of the tutorial rooms, which is TR1.
8	Abstract. In order to study the electromechanical coupling dynamics characteristics of the drive motor and gear transmission system in the electric vehicle electric drive system, firstly, an electromechanical coupling dynamics model of the electric vehicle electric drive system is established; the electromechanical coupling vibration characteristics of the electric drive system under current harmonic excitation is compared and analyzed.
9	The influence law of permanent magnet synchronous motor current harmonics on the meshing vibration of gear transmission system is revealed. The simulation results show that there is an obvious electromechanical coupling effect in the electric drive system; the harmonics of the drive motor can aggravate seriously the vibration state of the mechanical drive system; the research results provide a theoretical reference for further research on the vibration source positioning and active vibration reduction control strategy for the electric drive system of electric vehicles.

Продовження таблиці 4.1

№ вар.	Текст завдання
10	The electric drive system is one of the core components of electric vehicles, and its performance directly affects the safety and reliability of electric vehicles. The electric drive system is mainly composed of a drive motor and a gear transmission system, which is a typical electromechanical coupling system. As the electric drive system develops towards high speed and integration, the problem of coupled vibration has become more prominent, and seriously deteriorating the reliability and dynamic characteristics of the electric drive system of electric vehicles.
11	In recent years, the electromechanical coupling characteristics of electric drive systems have become a research hotspot for scholars. Yi et al. [1] established a shearer cutting system dynamic model including a cutting motor and a cutting drive system, and the influence of electromagnetic stiffness on the inherent characteristics of the gear system was studied. Chen [2] established a permanent magnet synchronous motor model of hybrid electric vehicles based on park transform and Fourier transform, which considered magnetic tension, saturated magnetic field, space harmonics and other factors, and the influence of electromagnetic parameters and structural parameters on the vibration of the electromechanical coupling system was studied.
12	Wenyu Bai et al. [3-4] considered factors such as the material of the motor with constant inductance, the spatial distribution of the magnetic field and the time-varying stiffness of the transmission system, and then, an electromechanical coupling dynamic model including the flux network motor and planetary gears was established. The study found that during process of transient change of voltage and loads, the electromechanical coupling system would vibrate violently, especially under the excitation of voltage transients.

Продовження таблиці 4.1

№ вар.	Текст завдання
13	The problem of search new design schemes of a traction mechanism for a locomotive with mass drive partial springing is under consideration. New design schemes are offered: a drive with a movable joint of a drive motor and an axial reducing gear and a drive with a disk motor with the separation of a stator and rotor. The authors offer to return to the development of supportframe drives with an axial reducing gear as it is simpler in manufacturing and assemblage on the basis of updated data of their operating modes. The authors also offer their own design of an integrated traction drive with a swivel of a motor and reducing gear which is simpler in manufacturing and assemblage as compared with the foreign analogues.
14	The introduction of nonsynchronous traction electric motors having a smaller mass and higher reliability as compared with commutator motors resulted in new designs of a supportaxial drive by foreign manufacturers and their introduction in the market of domestic rolling-stock. This work shows an attempt to determine possibilities to eliminate dependence mentioned by means of the analysis of basic problems in the development of supportaxial drive design and new structural scheme searches.
15	HEVs are vehicles propelled by more than one power source such as an engine and electric motor. They are classified by type and level. Advantages of HEVs are improved fuel economy, efficiency, and reduced emissions. The disadvantage of HEVs is cost. The cost aspect may be offset in years to come due to higher gas prices and improved HEV technologies. BEVs run only on electric An electric vehicle is one that operates on an electric motor, instead of an internal c mbustion engine that generates power by burning a mix of fuel and gases. A hybrid electric vehicle (HEV) augments an electric vehicle (EV) with a seco d source of power referred to as the alternative power unit.

Продовження таблиці 4.1

№ вар.	Текст завдання
16	A hybrid can achieve the power stored in the batteries and do not have an engine. cruising range and perfor ance advantages of conventional They emit zero emissions from the vehicle and are more vehicles with the low-noise, low-exhaust emissions, and energy efficient than HEVs. However, BEVs must be energy independence benefits of electric vehicles. charged from a plug, have a shorter driving range, and Accordingly, the hybrid concept, where the Accordingly, expensive batteries.
17	Although some B Vs, such as the Tesla Model S, have a range as high as 265 miles on a full charge, most are limited to around 100 miles per charge. the hybrid concept, where the Alternative power unit is used as a second source of energy, is gaining acceptance and is overcoming some of the problems of Hybrid power Despite this low range, in a study by the U.S. Department of Transportation Federal Highway Administration, "100 miles is sufficient for more than 90% of all household vehicle trips in the United States". Examples of BEVs on systems were conceived as a way to compensate for shortfall in battery technology. Because batteries could supply only enough energy for short trips, an onboard generator, powered by an internal combustion engine, could the market today are the Ford Focus EV, Nissan Leaf, Mitsubishi MiEV, and the Tesla Model S.
18	Advances in battery and other technologies, new federal standards for carbon-dioxide emissions and fuel economy, state zero-emission-vehicle requirements, and the current administration's goal of putting millions of alternative-fuel vehicles on the road have all highlighted PEVs as a transportation alternative. Consumers are also beginning to recognize the advantages of PEVs over conventional vehicles, such as lower operating costs, smoother operation, and better acceleration; the ability to fuel up at home; and zero tailpipe emissions when the vehicle operates solely on its battery. There are, however, barriers to PEV deployment, including the vehicle cost, the short all-electric driving range, the long battery charging time, uncertainties about battery life, the few choices of vehicle models.

Продовження таблиці 4.1

№ вар.	Текст завдання
19	What should industry do to improve the performance of PEVs and make them more attractive to consumers? At the request of Congress, Overcoming Barriers to Deployment of Plug-in Electric Vehicles identifies barriers to the introduction of electric vehicles and recommends ways to mitigate these barriers. This report examines the characteristics and capabilities of electric vehicle technologies, such as cost, performance, range, safety, and durability, and assesses how these factors might create barriers to widespread deployment.
20	Overcoming Barriers to Deployment of Plug-in Electric Vehicles provides an overview of the current status of PEVs and makes recommendations to spur the industry and increase the attractiveness of this promising technology for consumers. Through consideration of consumer behaviors, tax incentives, business models, incentive programs, and infrastructure needs, this book studies the state of the industry and makes recommendations to further its development and acceptance. © 2015 by the National Academy of Sciences. All rights reserved.
21	In modern days, electric vehicles are quickly industrialized as well as their penetration is also increased highly, which brings more challenges for the power system. The electric vehicle charge scheduling process is vital to encourage the daily usage of the electric vehicle. However, irregular charging methods for electric vehicles may disturb voltage security areas because of their stochastic characteristics. Moreover, an electric vehicle requires recurrent charging owing to its constrained battery capacity, but it is a time-consuming process. In this article, an effective charge scheduling model is devised using the fractional social sea lion optimization (Fr-SSLO) algorithm.

Продовження таблиці 4.1

№ вар.	Текст завдання
22	At first, IoEV network is simulated along with charge station and electric vehicle location. Furthermore, multi aggregator-based charge scheduling is done for increasing the profit and amount of scheduled electric vehicles. Then, routing is performed based on developed Fr-SSLO algorithm. Moreover, several fitness measures, including distance, energy and variable energy purchase are included. Here, the devised Fr-SSLO model is designed by integrating fractional calculus (FC) and sea lion optimization (SLnO) technique along with SOA. After the completion of routing process, charge scheduling is performed based on developed Fr-SSLO approach. Moreover, various fitness functions are also considered for computing better performance.
23	The demand for public and private transportation has soared astronomically during the last decades. Increasing the number of vehicles benefitting from internal combustion engines means fossil fuel consumption increase and though considerable amount of toxic contaminants' spread to the atmosphere, causing cancers and global warming. The automotive industry players have tried their best to mitigate these drawbacks, and electrification of transport industry is an important solution.
24	Then electric vehicle driving forces is reviewed; the items that have encouraged automotive engineers to create, develop and commercialize the use of vehicles running on electricity and the discouragement about not driving vehicles burning fossil fuels. Furthermore, the chapter discusses the changes in sales and markets of electric vehicles during the previous decades, at which the government's role in EV sales and utilization institutionalization, EV global market share, the items hindering global EV sales' growth, impact of Covid-19, and also statistics about heavy duty trucks electrification and relevant charging infrastructure development and deployment are clearly mentioned.

Продовження таблиці 4.1

№ вар.	Текст завдання
25	Internal combustion engine vehicles are a major cause of global warming and environmental air pollution. Electric Vehicles are being launched as an innovative green product in the Indian market as an alternative to conventional vehicles. This effort not only planned to moderate the GHG emission but will also help in reducing oil imports of the country. The average fueling time for the internal combustion engine vehicle goes in minutes, whereas for an EV it can go up to an hour. The major impediments in the path of adoption of the EV are its high purchase cost and charging time. This paper provides an overview of the studies of Battery Electric vehicle, Hybrid electric vehicle, Plug-in Hybrid electric vehicle, advances of EVs regarding battery technology trends, charging methods and adoption in several countries.
26	Approach to solve the integrated electric vehicle and crew scheduling problem. • Influence of vehicle and crew scheduling on the design of electric buses. • Comparing electric bus concepts taking into account local conditions. • Case study investigation for a real-world bus route. • Most cost-effective electric bus concept depends on the crew scheduling assumptions. Public transportation Electric bus Depot and opportunity charging Integrated vehicle and crew scheduling Adaptive large neighborhood search Total cost of ownership. Encouraged by international efforts to reduce greenhouse gases and local emissions, many public transport operators are converting their fleets to battery-powered electric buses.
27	Public transport operators can choose between different electric bus concepts, with the total cost of ownership being the most important decision criterion. The associated strategic decisions regarding charging strategy, vehicle concept, and charging infrastructure have a significant impact on the operational planning of the electric buses. Motivated by this, this paper aims to analyze the interactions between electrification and operational planning , especially vehicle scheduling and crew scheduling. This allows us to make a more comprehensive comparison of different electrification concepts.

Продовження таблиці 4.1

№ вар.	Текст завдання
28	Prior work has addressed the impact of electrification on vehicle scheduling but has neglected the interactions with crew scheduling. Crew scheduling dominates operational costs and planning for many public transport operators and must therefore be considered in all strategic decisions. For this reason, in this work we focused on integrated electric vehicle and crew scheduling problem. This allows us to calculate the total cost of ownership of different electric bus concepts under better representation of local conditions.
29	We tested the developed methodology for a real-world bus route. Our results indicate that the constraints for crew scheduling significantly impact the total cost of ownership and the required number of vehicles of the different electrification concepts. Our case study suggests that the choice of the most cost-effective concept depends significantly on crew scheduling constraints. These findings imply that crew scheduling constraints should be considered as part of the local framework for bus fleet electrification.
30	This chapter analyzes the bidding strategy problem of an aggregator managing a set of electric vehicles. The electric vehicle aggregator determines the bidding decisions in the market that minimize the charging costs of electric vehicles and, at the same time, meets the driving requirements of electric vehicle users. The models developed in this chapter take into account the uncertainty in both market prices and driving needs using a set of scenarios, which allows formulating the problem using a stochastic programming approach. The impact of the bidding decisions of the electric vehicle aggregator on market prices is also addressed in the decision-making tool. A number of illustrative examples explain how to formulate and solve this type of problems.

3) Оформляємо звіт.

Лабораторна робота №5

БАГАТОВИМІРНІ МАСИВИ

Мета роботи: розбір та опанування практичних навичок в застосуванні у своєму програмному коді багатовимірних масивів.

5.1 Короткі теоретичні відомості

Усі змінні всіх типів даних у MATLAB є багатовимірними масивами. Вектор – це одновимірний масив, а матриця – це двовимірний масив (дивіться рис. 5.1). Масиви осередків – це масиви індексованих осередків, де кожен осередок може зберігати масив різних вимірів та типів даних. В цій лабороторній роботі розглянемо деякі функції, які створюють спеціальні масиви [3, 4].

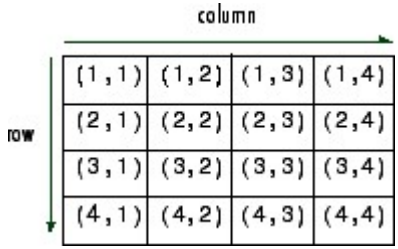


Рисунок 5.1 – Вид двовимірного масиву

Спеціальні функції:

1) **zeros(n)** – створює матрицю з n-рядками та n-стовбцями, заповнену нульовими значеннями;

```
zeros(4)
ans =
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
```

2) **ones(n)** – створює матрицю з n-рядками та n-стовбцями, заповнену значеннями, що дорівнюють «1»;

```
ones(4)
ans =
     1     1     1     1
     1     1     1     1
     1     1     1     1
     1     1     1     1
```

```
ones(4,3)
ans =
     1     1     1
     1     1     1
     1     1     1
     1     1     1
```

3) **eye(n)** – створює одиничну матрицю n x n;

```
eye(4)
ans =
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
```

4) **rand(row, column)** – створює масив рівномірно розподілених випадкових чисел від 0 до 1;

```
rand(3, 5)
ans =
    0.8147    0.9134    0.2785    0.9649    0.9572
    0.9058    0.6324    0.5469    0.1576    0.4854
    0.1270    0.0975    0.9575    0.9706    0.8003
```

5) **size(matrix)** – повертає розмір матриці (кількість рядків, та кількість стовбців);

```
A = rand(3,5)
A =
    0.1419    0.7922    0.0357    0.6787    0.3922
    0.4218    0.9595    0.8491    0.7577    0.6555
    0.9157    0.6557    0.9340    0.7431    0.1712
size(A)
ans =
     3     5
```

6) **length(matrix)** – повертає довжину вектора або найбільший розмір масиву;

```
length(A)
ans =
     5
```

7) **ndims(matrix)** – повертає розмірність масиву;

```
ndims(A)
ans =
     2
```

8) **numel(array)** – повертає кількість елементів масиву;

```
numel(A)
ans =
     15
```

9) **isempty(array)** – визначає, чи є масив пустим. Якщо масив пустий фунція повертає «1», інакше – «0»;

```
isempty(A)
ans =
     0
```

10) **sort(array)** – сортирує елементи масиву у бік збільшення;

```
B = [1, 5, 6, 8, 3, 2, 4];
sort(B)
ans =
     1    2    3    4    5    6    8

sort(A)
ans =
    0.1419   0.6557   0.0357   0.6787   0.1712
    0.4218   0.7922   0.8491   0.7431   0.3922
    0.9157   0.9595   0.9340   0.7577   0.6555
```

11) **cell** – використовується для створення масива;

```
C = cell(dim)
C = cell(dim1, ... , dimN)
% де C – масив осередок;
% dim – скалярне ціле число або вектор цілих чисел, котрий
% визначє розміри масива осередків C;
% dim1, ... , dimN – скалярні цілі числа, котрі визначають розміри C;

c = cell(2, 5);
c = {'Red', 'Blue', 'Green', 'Yellow', 'White'; 1 2 3 4 5}
c =
   'Red'    'Blue'    'Green'    'Yellow'    'White'
   [  1]    [   2]    [    3]    [     4]    [    5]
c{1}
ans =
       Red
c{1,2}
ans =
       Blue
c{2,2}
ans =
         2
c(1:2,1:2)
ans =
        'Red'    'Blue'
          [1]      [2]
```

Способи звертання до елементів масива:

```
% створюємо матрицю з довільними елементами
a = rand(3,5)
a =
            0.8147    0.9134    0.2785    0.9649    0.9572
            0.9058    0.6324    0.5469    0.1576    0.4854
            0.1270    0.0975    0.9575    0.9706    0.8003
```

1) звертання до конкретної ячейки масиву

```
% a(номер рядка, номер стовбця)
>> a(1,2)
ans =
            0.9134
```

2) звертання до секції масиву

```
% a(номер рядка, [номер 1 стовбця,  номер 2 стовбця])
>> a(1,[1,2])
ans =
          0.8147    0.9134

% або  a(номер рядка, [номер 1 стовбця,  номер 3 стовбця])
>> a(1,[1,3])
ans =
          0.8147    0.2785
```

3) використання оператора «:» при зверненні до всях рядків або всіх стовбців

```
>> a(1,:)
ans =
          0.8147    0.9134    0.2785    0.9649    0.9572
```

в цьому прикладі елементи беруться з першого рядка для всіх стовбців;

```
>> a(1,2:end)
ans =
          0.9134    0.2785    0.9649    0.9572
```

в цьому прикладі елементи беруться з першого рядка та для всіх стовбців починаючи з другого;

```
>> a(1,2:end-1)
ans =
          0.9134    0.2785    0.9649
```

елементи беруться з першого рядка та для всіх стовбців починаючи з другого та закінчуючі передостаннім стовбцем;

4) використання спеціальної функції **find(умова)**, в якості результату функція повертає лінійні індекси (індекси в порядку рахунку по стовбцям) елементів котрі задоволняють умові

```
ind = find(A < 0.5)
ind =
```

```
        3
        6
        7
        11
        14
```

також можна вивести окремо індекси рядків та стовбців:

```
[rows, columns] = find(A < 0.5)
rows =
        3
        3
        1
        2
        2
columns =
        1
        2
        3
        4
        5
```

Багатовимірні масиви (дивіться рис. 5.2) є розширенням 2-D матриць і використовують додаткові індекси для індексування. Тривимірний масив, наприклад, використовує три нижні індекси. Перші два схожі на матрицю, але третій вимір представляє сторінки або аркуші елементів.

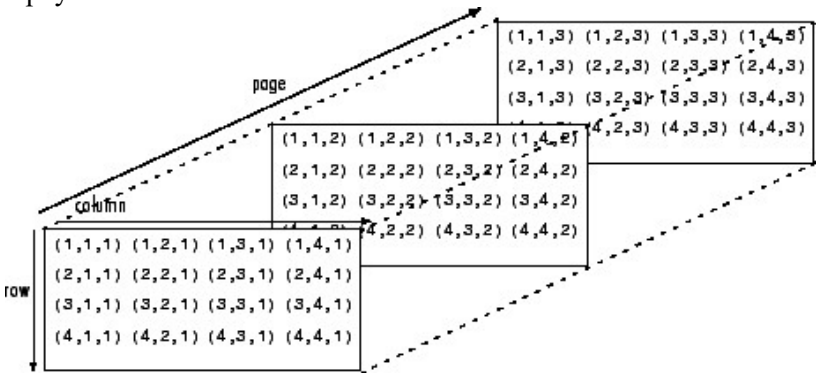


Рисунок 5.2 – Вид багатовимірного масиву

Можемо створити багатовимірний масив, спочатку створивши 2-D матрицю, а потім розширивши її. Наприклад, спочатку створивши матрицю 3 на 3 як першу сторінку в тривимірному масиві.

```
>> A = [1 2 3; 4 5 6; 7 8 9]
A =
     1     2     3
     4     5     6
     7     8     9
```

Тепер додамо другу сторінку. Для цього призначимо іншу матрицю 3 на 3 значенню індексу 2 у третьому вимірі. Синтаксис A(:,:,2) використовує двокрапку в першому та другому вимірах, щоб включити всі рядки та всі стовпці з правої сторони призначення.

```
A(:,:,2) = [10 11 12; 13 14 15; 16 17 18]
A(:,:,1) =
          1     2     3
          4     5     6
          7     8     9
A(:,:,2) =
         10    11    12
         13    14    15
         16    17    18
```

Спеціальна функція **cat** створює багатовимірні масиви. Наприклад, створемо новий тривимірний масив B, об’єднавши A з третьою сторінкою. Перший аргумент буде вказувати, який вимір потрібно об’єднати.

```
B = cat(3,A,[3 2 1; 0 9 8; 5 3 7])
B(:,:,1) =
          1     2     3
          4     5     6
          7     8     9
B(:,:,2) =
         10    11    12
         13    14    15
         16    17    18
```

```
B(:,:,3) =
     3     2     1
     0     9     8
     5     3     7
```

Ще один спосіб швидко розширити багатовимірний масив – призначити один елемент усій сторінці. Наприклад, додатb четверту сторінку до B, яка містить усі нулі.

```
>> B(:,:,4) = 0
B(:,:,1) =
     1     2     3
     4     5     6
     7     8     9
B(:,:,2) =
    10    11    12
    13    14    15
    16    17    18
B(:,:,3) =
     3     2     1
     0     9     8
     5     3     7
B(:,:,4) =
     0     0     0
     0     0     0
     0     0     0
```

Щоб отримати доступ до елементів у багатовимірному масиві, потрібно використовувати цілочисельні індекси так само, як і для векторів і матриць.

Наприклад, знайдемо елемент, який знаходиться в першому рядку, другому стовпці та на другій сторінці багатовимірного масиву A.

```
>> A
A(:,:,1) =
     1     2     3
     4     5     6
     7     8     9
```

```
A(:,:,2) =
        10    11    12
        13    14    15
        16    17    18
>> elementA = A(1,2,2)
elementA =
        11
```

При використанні вектору індексів [1 3] у другому вимірі, – отримується доступ лише до першого та останнього (третього) стовпців кожної сторінки багатовимірного масиву А.

```
>> C = A(:,[1 3],:)
C(:,:,1) =
        1    3
        4    6
        7    9
C(:,:,2) =
        10   12
        13   15
        16   18
```

Для того щоб знайти другий і третій рядки кожної сторінки, потрібно використати оператор двокрапки, щоб створити вектор індексу.

```
>> D = A(2:3,:,:)
D(:,:,1) =
        4    5    6
        7    8    9
D(:,:,2) =
        13   14   15
        16   17   18
```

Елементи багатовимірних масивів можна також переміщувати різними способами (подібно до векторів і матриць), змінювати форму, переставляти та стиснувати корисними функціями для перевпорядкування елементів. Розглянемо тривимірний масив із двома сторінками, що зображено на рис. 5.3.

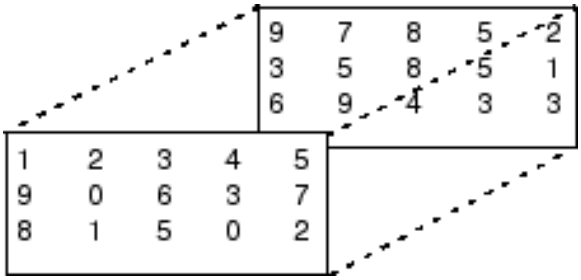


Рисунок 5.3 – Тривимірний масив із двома сторінками

Зміна форми багатовимірного масиву може бути корисною для виконання певних операцій або візуалізації даних. Наприклад, застосовуючі функцію для зміни форми **reshape**, щоб переставити елементи 3-D масиву в матрицю 6 на 5. **Reshape** працює по стовпцях, створюючи нову матрицю шляхом розміщення послідовних елементів у кожному стовпці A, починаючи з першої сторінки, а потім переходячи до другої сторінки. Перестановки використовуються для зміни порядку розмірів масиву.

```
>> A = [1 2 3 4 5; 9 0 6 3 7; 8 1 5 0 2];
A(:,:,2) = [9 7 8 5 2; 3 5 8 5 1; 6 9 4 3 3];
B = reshape(A,[6 5])

B =
    1    3    5    7    5
    9    6    7    5    5
    8    5    2    9    3
    2    4    9    8    2
    0    3    3    8    1
    1    0    6    4    3
```

Розглянемо тривимірний масив M. Використовуючи функцію перестановки **permute**, щоб поміняти нижні індекси рядків і стовпців на кожній сторінці, вказавши порядок розмірів у другому аргументі. Вихідні рядки M тепер стали стовпцями, а стовпці тепер рядками.

```
>> M(:,:,1) = [1 2 3; 4 5 6; 7 8 9];
M(:,:,2) = [0 5 4; 2 7 6; 9 3 1]
M(:,:,1) =
     1     2     3
     4     5     6
     7     8     9
M(:,:,2) =
     0     5     4
     2     7     6
     9     3     1
>> P1 = permute(M,[2 1 3])
P1(:,:,1) =
     1     4     7
     2     5     8
     3     6     9
P1(:,:,2) =
     0     2     9
     5     7     3
     4     6     1
```

Подібним чином змінемо місцями нижні індекси рядків і сторінок M.

```
>> P2 = permute(M,[3 2 1])
P2(:,:,1) =
     1     2     3
     0     5     4
P2(:,:,2) =
     4     5     6
     2     7     6
P2(:,:,3) =
     7     8     9
     9     3     1
```

5.2 Порядок виконання лабораторної роботи № 5

1) Створити прямокутну матрицю A, що має рядків N і M стовпців (значення N, M, L та K потрібно взяти відповідно до номеру варіанта з табл. 5.1) з випадковими цілими елементами. Виконати наступні завдання:

- розділити елементи матриці A на елемент матриці з максимальним значенням;
- розділити елементи кожного стовпця матриці A на елемент цього стовпця з найбільшим значенням;
- розділити елементи матриці A кожного рядка на елемент цього рядка з максимальним значенням;
- підсумувати елементи кожного рядка матриці з відповідними елементами L рядка;
- визначити, скільки нульових елементів міститься в рядку L матриці A і в K стовпці матриці;
- знайти найменше значення серед середніх значень кожного рядка матриці;
- знайти найменший елемент стовпця матриці A, котрого сума абсолютних значень елементів максимальна;
- знайти найбільше значення серед середніх значень кожного рядка матриці;
- визначити відсоток додатніх та від'ємних чисел від загальної кількості елементів матриці.

Таблиця 5.1 – Значення N, M, L та K

№ вар.	N	M	L	K	№ вар.	N	M	L	K
1	3	4	2	2	16	4	3	3	2
2	4	5	2	4	17	5	4	4	4
3	5	6	3	4	18	6	5	5	4
4	3	5	2	1	19	5	3	2	3
5	2	4	1	2	20	4	2	1	2
6	3	6	2	3	21	6	3	3	3
7	5	7	5	4	22	7	5	5	4
8	4	6	4	3	23	6	4	4	3
9	2	5	1	2	24	5	2	1	2
10	4	8	3	4	25	8	4	3	4
11	3	7	3	2	26	7	3	3	2
12	2	6	2	2	27	6	2	2	2
13	4	5	4	4	28	5	4	4	4
14	5	6	5	5	29	6	5	2	5
15	6	7	6	6	30	7	6	6	3

2) Створити та заповнити різними значеннями масив, котрий містить дані споживання електричної енергії в F поверхових H будинках. Будинки мають P під'їздів, на кожному поверсі будинку розтошовані R квартир. Інформація по кожній квартирі повинна містити: номер квартири, ПІБ мешканця, номінал встановленого автомату струму (обирайте з ряду: 10А, 16А, 25А, 32А, 40А, 50А, 63А), значення споживаної активної потужності.

Написати код виводу інформації за певним (згідно до 6 стовпчика у табл. 5.2) на екран. На екран результат потрібно виводити в наступному форматі:

Номер квартири – ПІБ мешканця – «Результат умови запиту»

Таблиця 5.2 – Значення F, H, P та R

№ вар.	F	H	P	R	Умова запиту
1	2	3	4	2	Вивести на екран 10 мешканців з найбільшим енергоспоживанням.
2	4	2	2	2	Вивести на екран 10 мешканців з найменшим енергоспоживанням.
3	3	2	2	1	Вивести на екран мешканців з автоматом струму 16А.
4	3	2	3	1	Вивести на екран мешканців з автоматом струму більше 16А.
5	2	4	2	2	Вивести на екран мешканців з автоматом струму менше 16А.
6	2	4	2	2	Вивести на екран 5 мешканців з найбільшим енергоспоживанням в порядку зростання.
7	4	2	4	2	Вивести на екран 5 мешканців з найменшим енергоспоживанням в порядку зростання.
8	5	1	2	2	Вивести на екран мешканців з автоматом струму 25А.
9	1	3	2	4	Вивести на екран мешканців з автоматом струму більше 25А.
10	2	2	3	1	Вивести на екран мешканців з автоматом струму менше 25А.
11	3	1	2	2	Вивести на екран 3 мешканців з найбільшим енергоспоживанням в порядку зростання.
12	3	3	2	2	Вивести на екран 3 мешканців з найменшим енергоспоживанням в порядку зростання.
13	4	2	4	2	Вивести на екран мешканців з автоматом струму 10А в порядку спадання.

Продовження таблиці 5.2

№ вар.	F	H	P	R	Умова запиту
14	2	1	2	2	Вивести на екран мешканців з автоматом струму більше 10А.
15	5	3	2	1	Вивести на екран мешканців з автоматом струму менше 10А.
16	1	2	3	4	Вивести на екран 10 мешканців з найбільшим енергоспоживанням в порядку спадання.
17	2	1	2	2	Вивести на екран 10 мешканців з найменшим енергоспоживанням в порядку спадання.
18	3	3	2	2	Вивести на екран мешканців з автоматом струму 16А.
19	4	2	4	2	Вивести на екран мешканців з автоматом струму більше 16А.
20	5	1	2	2	Вивести на екран мешканців з автоматом струму менше 16А.
21	2	3	2	4	Вивести на екран 10 мешканців з найбільшим енергоспоживанням в порядку спадання.
22	1	3	2	4	Вивести на екран 10 мешканців з найменшим енергоспоживанням в порядку спадання.
23	4	2	4	2	Вивести на екран мешканців з автоматом струму 40А.
24	5	1	2	2	Вивести на екран мешканців з автоматом струму більше 40А.
25	2	3	2	1	Вивести на екран мешканців з автоматом струму менше 10А.
26	1	2	3	1	Вивести на екран 6 мешканців з найбільшим енергоспоживанням.
27	4	1	2	2	Вивести на екран 8 мешканців з найменшим енергоспоживанням.
28	2	3	2	2	Вивести на екран мешканців з автоматом струму 10А.
29	3	2	4	2	Вивести на екран мешканців з автоматом струму більше 25А в порядку зростання.
30	5	1	2	2	Вивести на екран мешканців з автоматом струму менше 25А в порядку спадання.

3) Оформляємо звіт.

Лабораторна робота №6

ОПЕРАТОРИ ВВОДУ/ВИВОДУ. ФУНКЦІЇ

Мета роботи: розбір та опанування практичних навичок в застосуванні у своєму програмному коді операторів вводу/виводу.

6.1 Короткі теоретичні відомості

6.1.1 Оператор вводу input

Функція введення **input** дозволяє попросити користувача ввести певну інформацію в програму та зберегти цю інформацію у змінній, яку програма може обробити [1 – 4]:

```
age = input('how old are you: ');
% At this point, the variable: age, will contain
% whatever value the user types
```

За замовчуванням функція **input** очікує читання числа, і якщо користувач вводить: hello, припускатиметься, що hello — це змінна, яка містить число. Якщо вам дійсно потрібен рядок «hello», тоді вам доведеться ввести «hello» (галочки). Щоб наказати програмі Matlab читати рядок символів безпосередньо без необхідності вводити галочки, ви повинні використовувати синтаксис 's', як показано тут:

```
>> name = input('write your name: ', 's');
        name = Dan
% At this point, the variable: name, will contain whatever
% value the user types (as a string of characters), in this case 'Dan'

>> name = input('what is your name: ');
% user types jim without tick marks
      ??? Error using ==> input
      Undefined function or variable 'Dan'.

>> name = input('what is your name: ');
% user types 'jim' with tick marks
      name = Dan
```

6.1.2 Оператор виводу fprintf

Для виведення даних на екран широке застосування отримала функція **fprintf**. Розглянемо роботу з цією функцією на наступному прикладі.

Приклад 1.

```
>> A1 = [9.9, 9900];
A2 = [8.8,  7.7 ; ...
     8800, 7700];
formatSpec = 'X is %4.2f meters or %8.3f mm\n';
fprintf(formatSpec,A1,A2)
X is 9.90 meters or 9900.000 mm
X is 8.80 meters or 8800.000 mm
X is 7.70 meters or 7700.000 mm
```

В цьому прикладі: **%4.2f** – у вхідних даних formatSpec вказує, що перше значення в кожному рядку виведення є числом з плаваючою комою з шириною поля чотири цифри, включаючи дві цифри після коми; **%8.3f** – у вхідних даних formatSpec вказує, що друге значення в кожному рядку виводу є числом з плаваючою комою з шириною поля у вісім цифр, включаючи три цифри після коми; **\n** – символ керування, з якого починається новий рядок.

Приклад 2.

```
>> a = [1.02   3.04   5.06];
fprintf('%d \n',  round(a));
1
3
5
```

В цьому прикладі: **%d** у вхідних даних друкує кожне значення у векторі; **round(a)** – округляє число до цілих значень зі знаком.

В табл. 6.1 наведені службові символи перетворення та форматування числових і символьних даних як тексту.

Таблиця 6.1 – Службові символи перетворення та форматування числових і символьних даних як тексту

Тип значення	Символ	Примітка
Ціле число зі знака	%d	Число десятичної системи
Ціле число без знака	%u	Число десятичної системи
	%o	Число восьмирічної системи (octal)
Число з плаваючою комою	%x	Число шістнадцятирічної системи (hexadecimal)
	%f	Оператор точності, котрий вказує кількість цифр до та після коми
	%e	Експоненціальний запис, наприклад 3.141593e+00 (використовується оператор точності, котрий вказує кількість цифр до та після коми)
	%g	Більш компактний запис з %e або %f, без кінцевих нулів (використовується оператор точності, для вказання кількості значущих цифр)
Символи або рядки	%c	Один символ
	%s	Символьний вектор або масив рядків
	\a	Тривога
	\b	Backspace
	\n	Перехід на новий рядок
	\r	Повернення каретки (курсора)
	\t	Горизонтальна табуляція
	\v	Вертикальна табуляція

6.1.3 Спеціальна функція function

Функція записується в окремому файлі з розширенням *.m. Ім'я файлу повинно мати те саме значення як і ім'я функції.

Розгянемо декілька способів застосування **function**:

1) з одним вхідним аргументом

M-файл з ім'ям myfunc.m

```
function out=myfunc(in)
out=in^2;
end
```

Запуск m-файлу виконується наступним чином:

```
>> myfunc(4)
ans =
      16
```

2) декілька вхідних аргументів

М-файл з ім'ям myfunc.m

```
function out=myfunc(in1,in2)
out=in1+in2;
end
```

Запуск m-файлу виконується наступним чином:

```
>> myfunc(4,5)
ans =
       9
```

3) один вхідний аргумент та декілька вихідних елементів

```
function [out1,out2]=myfunc(in)
out1=in^2;
out2=in^3;
end
```

Запуск m-файлу виконується наступним чином:

```
>> [a,b]=myfunc(5)
a =
     25
b =
     125
```

4) декілька вхідних та вихідних аргументів

```
function [out1,out2]=myfunc(in1,in2,in3)
out1=in1+in2;
out2=in2+in3;
end
```

Запуск m-файлу виконується наступним чином:

```
>> [a,b]=myfunc(2,4,5)
a =
     6
b =
     9
```

6.2 Порядок виконання лабораторної роботи № 6

1) Написати функцію яка буде задовольняти умовам згідно індивидуальним варіантам з табл. 6.1. При виконанні завдань обов'язкове використання функцій: input, fprintf та function.

Таблиця 6.1 – Індивідуальні завдання

№ вар.	Завдання
1	Функція повертає двоїчний еквівалентний код, що відповідає будьякому введеному слову
2	Функція визначає чи є введений рік високосним.
3	Функція повертає шістнадцятирічний еквівалентний код, що відповідає будьякому введеному слову
4	Функція повертає відсортировний масив рядок у бік збільшення. Значення елементів масиву вводиться користувачем з клавіатури до тих пір, – доки користувач не напише кодове слово «end» і натисне Enter.
5	Функція повертає корні рівняння типу: $Ax^B + Cx^D + K = 0$. Значення коефіцієнтів вводиться користувачем з клавіатури.
6	Функція повертає слово навпаки (дзеркальне зображення), в наступному вигляді: «Введене слово - дзеркальне зображення»
7	Функція повертає корні рівняння типу: $Ax^{B*D-C} + (Cx + K * A)^{C/D} = 0$. Значення коефіцієнтів вводиться користувачем з клавіатури.
8	Функція повертає транслітерацію на англійську мову введеного користувачем слова: «Введене слово – слово-транслітирація»

Продовження таблиці 6.1

№ вар.	Завдання
9	Розробити функцію, котра повертає ряд Фібоначчі (послідовність натуральних чисел, де кожне наступне число є сумою двох попередніх: 1 1 2 3 5 8 13 21 34 55 89…). Користувач вводить з клавіатури число, що відповідає кількості результуючих елементів.
10	Функція генерує псевдовипадкове число, яке записується в внутрішню змінну a. Доки число a не співпаде з числом b (котре вводить користувач з клавіатури), користувачеві пропонуватиметься ввести чергове число. При цьому якщо b > a, то на екран буде видаватися повідомлення "Багато", інакше – "Мало". Коли користувач вкадає повенне з'явитися повідомлення "Вгадав".
11	Функція повертає для кожного натурального числа в проміжку від M до N вивести всі дільники, крім одиниці та самого числа. Значення M та N вводяться з клавіатури.
12	Функція повертає число з цифрами розташованими в зворотньому порядку (наприклад, якщо введено число 3486, треба вивести на екран: числу 3486 відповідає число 6843).
13	Функція повертає корні рівняння типу: $Ax^{B\text{-}D} + (Cx + K)^{1/D} = 0$. Значення коефіцієнтів вводиться користувачем з клавіатури.
14	Функція видаляє введеного користувачем цифру з числа (також введеного користувачем з клавіатури) Наприклад, задано число 5683. Потрібно видалити з нього цифру 8. Вийде число 563.
15	Написати функцію, котра знаходить всі комбінації із трьох чисел до певної межі, які в сумі дають число, що введене користувачем з клавіатури.
16	Написати функцію, в якій вводяться два числа-операнда x та y та знак арифметичної операції (+, –, /, *). Функція визначити результат z залежно від знака арифм. операції.
17	Функція виводить у порядку зростання цифри, що входять до десятичного натурального числа, введеного користувачем (приклад: 4663299 → 2 3 4 6 6 9 9)

Продовження таблиці 6.1

№ вар.	Завдання
18	Функція повертає корні рівняння типу: $Ax^B + (Cx + K)^D = 0$. Значення коефіцієнтів вводиться користувачем з клавіатури.
19	В функцію в якості аргументів з клавіатури передаються три числа, що позначають число, місяць, рік. Функція повертає номер дня (або кількість днів), починаючи з початку введеного користувачем року.
20	Функція перевіряє чи є введене користувачеми слово паліндромом.
21	Функція повертає корні рівняння типу: $Ax^B + \sqrt[D]{Cx^A + K} = 0$. Значення коефіцієнтів вводиться користувачем з клавіатури.
22	Функція повертає відсортирований масив у бік збільшення методом «сортировка бульбашкою» (Bubble sort). Значення елементів масиву вводиться користувачем з клавіатури до тих пір, – доки користувач не напише кодове слово «end» і натисне Enter.
23	Функція повертає відсортирований масив у бік збільшення методом «сортировка шейкерна» (Shaker sort). Значення елементів масиву вводиться користувачем з клавіатури до тих пір, – доки користувач не напише кодове слово «end» і натисне Enter.
24	Функція повертає відсортировний масив рядок у бік зменшення. Значення елементів масиву вводиться користувачем з клавіатури до тих пір, – доки користувач не напише кодове слово «end» і натисне Enter.
25	Функція повертає відсортирований масив у бік збільшення методом «сортировка вставками» (Insertion sort). Значення елементів масиву вводиться користувачем з клавіатури до тих пір, – доки користувач не напише кодове слово «end» і натисне Enter.
26	Функція повертає символьний ряд, що відповідає будьякому введеному шістандцятирічному коду

Продовження таблиці 6.1

№ вар.	Завдання
27	Функція повертає корні рівняння типу: $Ax^{B*D} + (Cx + K * A)^{1/D} = 0$. Значення коефіцієнтів вводиться користувачем з клавіатури.
28	Функція повертає відсортированый масив у бік збільшення методом «сортировка вибором» (Selection sort). Значення елементів масиву вводиться користувачем з клавіатури до тих пір, – доки користувач не напише кодове слово «end» і натисне Enter.
29	Функція повертає найбільший спільний дільник двох введених користувачем чисел з клавіатури.
30	Функція повертає символьний ряд, що відповідає будьякому введеному двоїчному коду

2) Оформляємо звіт.

Лабораторна робота №7

ШИФРОВАННЯ

Мета роботи: розбір та опанування практичних навичок в кодуванні різними шифрами.

7.1 Короткі теоретичні відомості

7.1.1 Шифр Цезаря

Шифр Цезаря полягає в тому, що кожна буква, котру потрібно зашифрувати, обирається з того ж самого алфавіту, але з зсувом на деяке значення Key [2]. Значення позиції шифрованої букви визначається за формулою:

$$Y = X + Key \qquad (7.1)$$

Наведемо приклад шифрування слова «PLAY» шифром Цезаря з ключом Key = 3.

Запишемо в табл. 7.1: в перший рядок – порядковий номер символу; у другий рядок – стандартний англійський алфавіт; у третій рядок – стандартний англійський алфавіт зсунутий на 3 літери вліво.

Таблиця 7.1 – Шифр Цезаря

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

З табл. 7.1 видно, що зашифроване значення букви під номером X (в стандартном алфавіті) є буква, що стоїть на позиції X + 3 (також в стандартному алфавіті).

Використовуючи шифровальну табл. 7.1 можна легко заширувати слово: PLAY → SODB.

Зворотнє перетворення виконується таким самим чином, за виразом X – 3.

7.1.2 Шифр Альберті (1460р.)

Для шифрування якого небудь повідомлення Альберті запропонував чергувати два шифроалфавіта (дивіться табл. 7.2).

Таблиця 7.2 – Шифр Альберті

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

В табл. 7.2: перший рядок – стандартний алфавіт, другий рядок – перший шифроалфавіт, третій рядок – другий шифроалфавіт.

Розглянемо принцип застосування шифру Альберті при шифрованні слова «THYRISTOR», наведений на рис. 7.1, а. При шифруванні слова непарні за чергою слідування в оригінальному слові букви беруться з першого шифроалфавіту (другий рядок), а парні – з другого шифроалфавіту (третій рядок). Використовуючи шифровальну табл. 7.2 отримаємо:

Ісходне слово	T	H	Y	R	I	S	T	O	R
шифрований алфавіт (2)	W		B		L		W		U
шифрований алфавіт (3)		L		O		I		S	
Результат	W	L	B	O	L	I	W	S	U

а)

Ісходне слово	E	O	D	K	Q
шифровані алфавіти	(2)	(3)	(2)	(3)	(2)
Результат	B	R	A	I	N

б)

а – шифрування слова «THYRISTOR»; б – дешифрування слова «EODKQ»

Рисунок 7.1 – Приклади шифрування та дешифрування шифром Альберті

7.1.3 Шифр «Квадрат Віженера»

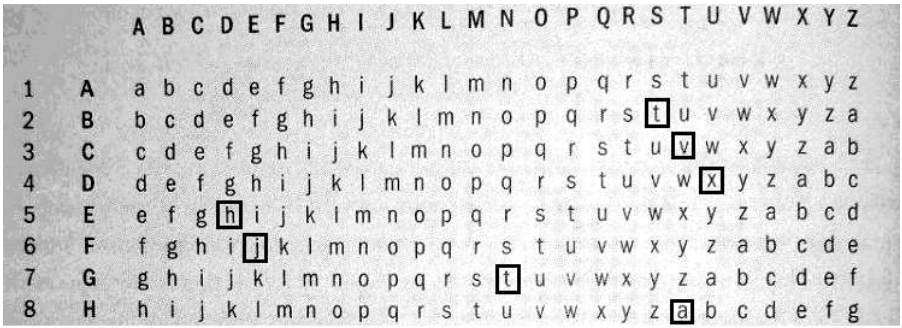
Шифр «Квадрат Віженера» відповідає класу поліалфавітних шифрів. Для шифрування тексту при переході від однієї букви повідомлення до іншої використовуються різні шифроалфавіти.

Таблиця квадрату Віженера складається із стандартного алфавіта з *n* букв, під котрим стоять *n* шифроалфавітів, зсунутих циклічно на одну букву вліво впорівнянні до вищезображеного алфавіту. Тобто, ця матриця складається с 26 рядків і 26 стовпців, та зображена на рис. 7.2.

		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	A	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
2	B	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a
3	C	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b
4	D	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c
5	E	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d
6	F	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e
7	G	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f
8	H	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g
9	I	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h
10	J	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i
11	K	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j
12	L	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k
13	M	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l
14	N	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m
15	O	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n
16	P	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
17	Q	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
18	R	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q
19	S	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r
20	T	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s
21	U	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t
22	V	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u
23	W	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
24	X	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w
25	Y	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x
26	Z	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y

Рисунок 7.2 – Таблиця шифру «Квадрат Віженера»

Розглянемо принцип застосування шифру «Квадрат Віженера» при шифруванні слова «STUDENT» та зворотного перетворення, наведені на рис. 7.3.



		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	A	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
2	B	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a
3	C	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b
4	D	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c
5	E	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d
6	F	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e
7	G	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f
8	H	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g

а)

Шифроване слово	T	V	X	H	J	T	A
Номер рядку	2	3	4	5	6	7	8
Результат	S	T	U	D	E	N	T

б)

а – шифрування слова «STUDENT»; б – дешифрування слова «TVXHJTA»

Рисунок 7.3 – Приклади шифрування та дешифрування шифром «Квадрат Віженера»

7.2 Порядок виконання лабораторної роботи № 7

1) Написати функції шифратора та дешифратора для кожної з трьох розглянутих систем шифрування. Для шифрів Цезаря та Альберті ключ Key потрібно надавати в функцію в якості аргумента.

2) Зашифрувати своє прізвище кожним з трьох способів шифрування.

3) Отримати дешифроване значення шифрованого слова відповідно до номера варіанту з табл. 7.3.

Таблиця 7.3 – Індивідуальні завдання

№ вар.	Слово	№ вар.	Слово	№ вар.	Слово
1	zspxeki	11	bzivanwzumz	21	kypcl
2	rqygt	12	utbjw	22	burzgmk
3	mebboxd	13	jxobk	23	cbjre
4	xverwjsvqiv	14	cajwbrbcxa	24	uwbwz
5	lzqdm	15	atqyflj	25	cvsahnl
6	cejeh	16	hszsdvcbs	26	octgp
7	iaxxktz	17	fdmzeuefad	27	ldaanwc
8	zkrkvnutk	18	xwemz	28	jiqyl
9	yafad	19	zxgtyluxskx	29	thylu
10	ywfsxnxytw	20	jbyylua	30	fyvdkqo

4) Оформляємо звіт.

Лабораторна робота №8

ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

Мета роботи: розбір та опанування практичних навичок в об'єктно-орієнтованому програмуванні.

8.1 Короткі теоретичні відомості.

В основі об'єктно-орієнтованого програмування (ООП) лежать три основних принципа [2 – 4]:

інкапсуляція – об`єднання даних і програм і передачу даних через вхідні і вихідні параметри функцій. В результаті з`являється новий елемент програмування – **об'єкт**;

наслідування – можливість створення батьківських об'єктів та нових дочірніх об'єктів, наслідуючих властивості батьківських об'єктів. Можливо також множинні наслідування, при яких клас наслідує властивості декількох батьківських об'єктів. На наслідуванні основані системи задання типів даних, дескрипторна графіка і багато інших прийомів програмування;

поліморфізм – присвоєння деякій дії одного імені, яке в подальшому використовується по всьому ланцюгу створюваних об'єктів зверху до низу, причому кожний об'єкт виконує цю дію звичним йому способом.

ООП в MATLAB дозволяє агрегування об'єктів, тобто об`єднання частин об'єктів або ряду об'єктів в єдине ціле.

Об`єкти можна визначати як деяку структуру, яка належить деякому класу. В MATLAB визначаються сім класів об'єктів:

double – числові масиви з елементами – числами подвійної точності;

sparse – двовимірні числові або комплексні розріджені матриці;

char – масиви символів;

struct – масиви структур (записів);

cell – масиви комірок;

function_handle – дескрипторні функції.

Для створення класа об'єктів або об'єктів, а також для їх ідентифікації використовується функція **class**.

class(OBJ) – повертає клас вказаного об'єкта **OBJ**;

OBJ=class(S,'class_name',PARENT1,PARENT2,…) – створює об'єкт класа **'class_name'** на базі структури **S** і батьківських об'єктів **PARENT1**, **PARENT2**… при цьому створюваний об'єкт наслідує структуру і поля батьківських об'єктів. Об'єкт **OBJ** в даному випадку має множинне наслідування;

OBJ=class(struct[ ],'class_name', PARENT1,PARENT2,…) – не може мати полів, крім унаслідуваних від батьківського об'єкта.

Для контроля належності об'єкта до деякого класу використовується функція **isa**:

isa(OBJ,'class_name') – повертає логічну одиницю, якщо **OBJ** належить класу з вказаним іменем.

Приклад:

```
>>X=[1 2 3];
>>isa(X,'char')
Ans =
        0

>>isa(X,'double')
Ans =
1
```

Для отримання списку методів даного класу об'єктів використовують **methodsview name** та **methods**.

methodsview name_class або **methods name_class – full** – в окремому вікні повертають повний опис методів класа, включаючи інформацію про наслідування.

M=methods('class_name') – повертає масив комірок з перерахунком методів, що відносяться до заданого класа об'єктів;

Methods class_name – повертає перелік методів в окреме вікно.

Приклад:

```
>> methods char

Methods for class char:

abs          full          mrdivide      reshape
…….
```

Для створення нового макету класу потрібно натиснути на відповідну вкладку, на котру вказує красна стрілка на рис. 8.1. При натисненні на кнопку у вікні m-файлу з’явиться макет класу, як це зображено на рис. 8.2.

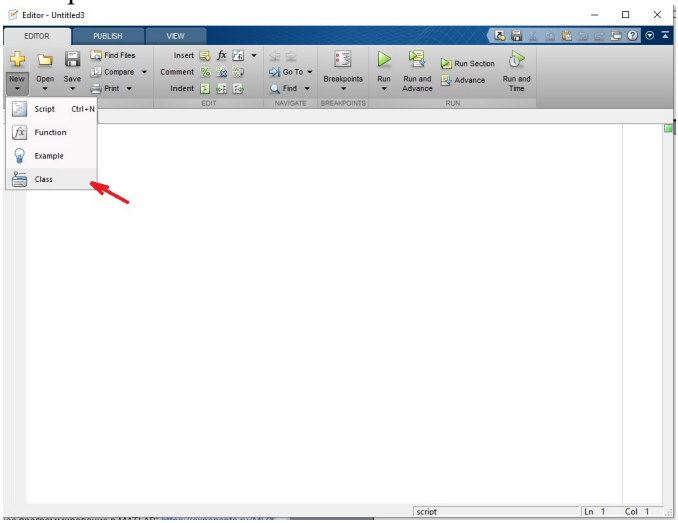


Рисунок 8.1 – Кнопка створення нового макету класу

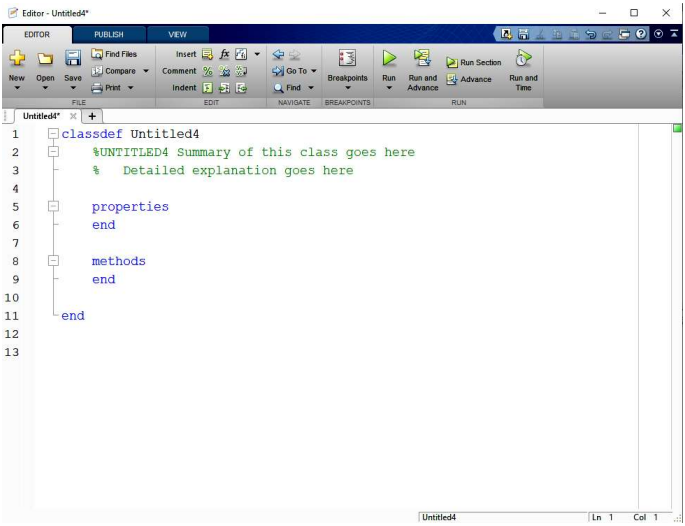


Рисунок 8.2 – Макет нового порожнього класу

Розглянемо створення класу на прикладі класу myDate, текст коду наведено на рис. 8.3.

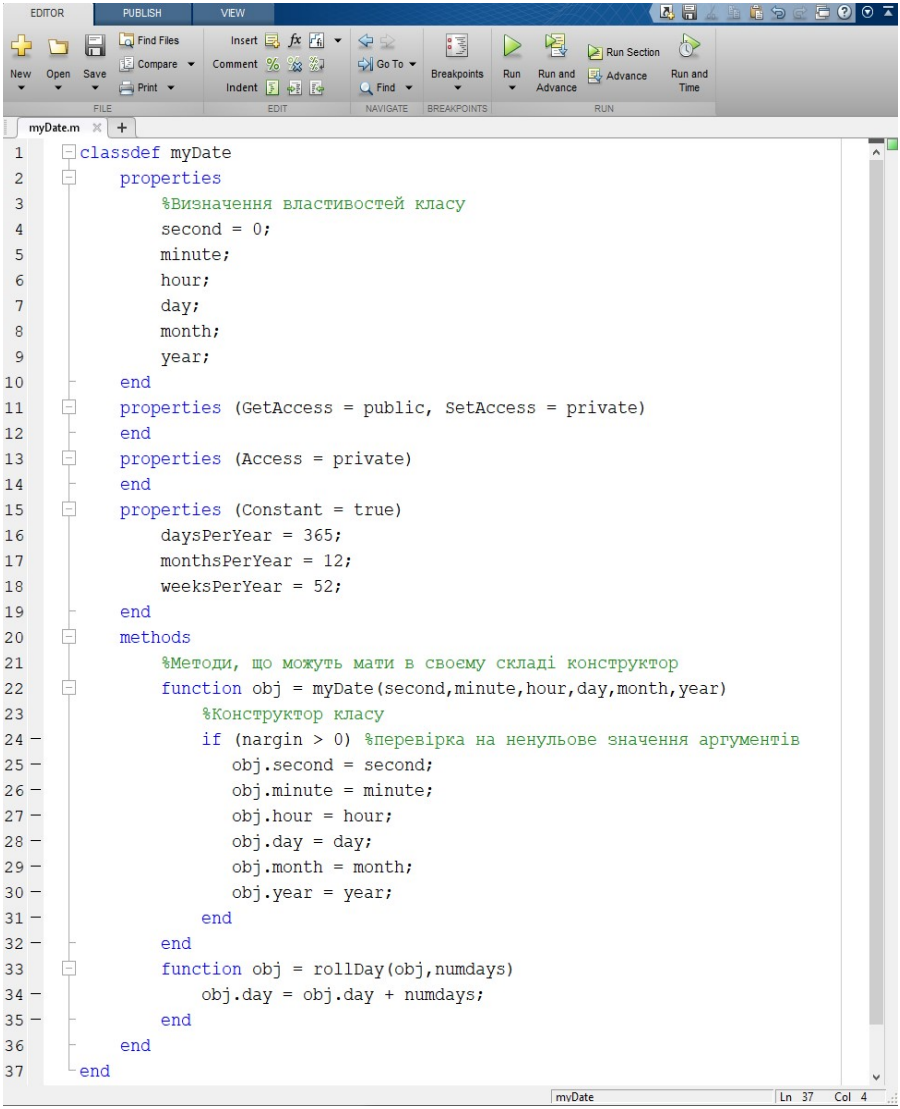


Рисунок 8.3 – Код класу myDate

У 11 рядку виконується налаштування доступу змінних. Всього можливі три варіанти (**public** – доступність зміни із зовні, **private** – доступність на зміну із середини, **protected** – захищений на змінення із зовні) доступу для двох режимів (**GetAccess** – читання, **SetAccess** – запис). Якщо налаштування доступу у полей співпадають між собою можна скористатися короткою формою запису: **Access = private**.

У рядках 15 – 19 задаються вбудовані у клас константи, тобто ті величини значення котрих не будуть змінюватись при роботі класу.

У 22 рядку створюється функція-конструктор, котра необхідна для створення нового еземпляру класу з необхідними його полями. Він повинен мати теж ім'я, що і ім'я класу та необхідну кількість полей.

У 33 рядку створюється і додається новий метод rollDate класу додавання кількості днів до класу.

Створимо новий об'єкт класу myDate:

```
>> date1 = myDate(0,0,3,27,4,2023)
date1 =
  myDate with properties:

              second:  0
              minute:  0
                hour:  3
                 day:  27
               month:  4
                year:  2023
         daysPerYear:  365
        monthsPerYear:  12
         weeksPerYear:  52
```

Отримаємо доступ до поля значення поля днів:

```
>> day = date1.day
day =
          27
```

Змінемо значення року в об'єкті:

```
>> date1.year = 2002
```

```
date1 =
  myDate with properties:

              second:         0
              minute:         0
                hour:         3
                 day:         27
               month:         4
                year:         2002
     daysPerYear:             365
  monthsPerYear:              12
   weeksPerYear:              52
```

Переглянемо всі властивості класу:

```
>> properties(myDate)
Properties for class myDate:
   second
   minute
   hour
   day
   month
   year
   daysPerYear
   monthsPerYear
   weeksPerYear
```

Скористаємось вбудованим методом для збільшення кількості днів:

```
>> date2 = date1.rollDay(3)
date2 =
  myDate with properties:

              second:         0
              minute:         0
                 hour:        3
                day:          30
              month:          4
```

```
          year:     2002
    daysPerYear:    365
  monthsPerYear:    12
   weeksPerYear:    52
```

В класі можна створювати статичні методи класу, котрі можна визивати лише в середині класу, а не у зовнішнього екземляру класу:

```
methods (Static = true)
    function printCurrentDate()
       display(datestr(now));
    end
end
```

Створення скритих методів:

```
methods (Hidden = true)
end
```

Класи підтримують успадкування:

classdef className < superClass – className буде успатковувати всі методи та властивості superClass;

classdef className < superClass & class2 & class3 – className буде успатковувати всі методи та властивості superClass, class2 та class3.

Створення запечатаного класу (заборона на успадкування):

```
classdef(Sealed = true) myclass
method(Sealed = true)
properties(Sealed = true)
```

Створення абстрактного класу – клас методи котрого будуть визначені в ітоговом класі, в котрому буде виконуватись робота:

```
methods (Abstract = true)
      function vol = calculateVolume(obj, units);
      function area = calculateSurfaceArea(obj, units);
      function obj = doubleSize(obj);
end
```

В суперкласі створюються лише заголовки методів, а саме їх реалізація виконується в успадкованих класах.

Створимо масив об’єктів:

```
>> d1 = myDate(0,4,3,27,2,2000);
d2 = myDate(0,4,3,28,4,2023);
dates = [d1 d2] % об’єднання об’єктів
dates =
        1x2 myDate array with properties:
            second
            minute
            hour
            day
            month
            year
            daysPerYear
            monthsPerYear
            weeksPerYear
```

Визов першого елементу масиву dates:

```
>> d1 = dates(1,1)
d1 =
        myDate with properties:
            second: 0
            minute: 4
            hour: 3
            day: 27
            month: 2
            year: 2000
            daysPerYear: 365
            monthsPerYear: 12
            weeksPerYear: 52
```

8.2 Порядок виконання лабораторної роботи № 8

Виконайти наступні завдання:

1) Створіть клас, котрому можна передати значення: радіусу, висоти, ширини, довжини, щільність. За запитом клас може повертати назву фігури, значення її об‘єму та ваги.

2) Реалізуйте клас Worker (Працівник), який матиме такі властивості: name (ім'я), surname (прізвище), rate (ставка за день роботи), days (кількість відпрацьованих днів). Також клас повинен мати метод getSalary(), який виводитиме зарплату працівника. Зарплата – це добуток ставки rate на кількість відпрацьованих днів days.

3) Створіть клас групи студентів, що містить: номер групи, кількість студентів, ПІБ, курсу, семестр, кількість та назви дисциплін, оцінки, кількість пропусків на протязі місяця; та методи: виводу загальної кількості пропусків, ПІБ чотирьох успішних студентів, середній бал групи, графік успішності студентів за кожним семестром окремо та на протязі всього навчання.

Умови: кількість студентів не меньше 5; кількість предметів – не менше 4; кількість курсів – 2 (семестрів – 4).

Перелік джерел посилання

1. Hant B.R., Lipsman R.L., Rosenberg J.M. A Guide to Matlab. Cambridge university press. 2001. 347p. https://engineering.futureuniversity.com/BOOKS%20FOR%20IT/A%20Guide%20to%20MATLAB%5BBrian_R._Hunt__Ronald_L._Lipsman__J._Rosenberg__K(BookFi.org).pdf

2. Chapman S.J. Matlab programming for engineers. Cengage Learning. 2018. 865p. https://www.mohsin-sies.com/scsj2273/books/Stephen%20J.%20Chapman%20-%20MATLAB%20Programming%20for%20Engineers-Cengage%20Learning%20(2019).pdf

3. Fitzpatrik J.M., Ledeczi A. Computer programming with MATLAB. 2015. 366p. https://repo.darmajaya.ac.id/5660/1/Computer%20Programming%20with%20MATLAB.pdf

4. Офіційний сайт MATLAB [Електронний ресурс] https://www.mathworks.com/help/?s_tid=user_nav_help (16.04.2026)