

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ПРОГРАМУВАННЯ ТА АЛГОРИТМІЧНІ МОВИ

ЧАСТИНА 1

АЛГОРИТМІЗАЦІЯ ТА ОСНОВИ ПРОГРАМУВАННЯ

Лабораторний практикум

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для здобувачів ступеня бакалавра
за освітньою програмою «Системний аналіз і управління»
спеціальності 124 «Системний аналіз»

Укладач: І. В. Назарчук

Електронне мережеве навчальне видання

Київ
КПІ ім. Ігоря Сікорського
2023

УДК 004.42/.43(076)
П78

Укладач: *Назарчук Ірина Василівна*

Рецензент *Терентьев О. М., д. т. н., доцент, провідний науковий співробітник Інституту телекомунікацій і глобального інформаційного простору НАН України*

Відповідальний редактор *Тимошук О.Л., к.т.н., доцент*

*Гриф надано Методичною радою КПІ ім. Ігоря Сікорського
(протокол № 4 від 01.02.2024р.)
за поданням вченої ради факультету/навчально-наукового інституту
(протокол № 11 від 18.12.2023р.)*

Програмування та алгоритмічні мови. Частина 1. Алгоритмізація та основи програмування [Електронний ресурс] : лаб. практикум : навч. посіб. для здобувачів ступеня бакалавра за освіт. програмою «Системний аналіз і управління» спец. 124 Системний аналіз / КПІ ім. Ігоря Сікорського ; уклад.: І. В. Назарчук. – Електрон. текст. дані (1 файл). – Київ : КПІ ім. Ігоря Сікорського, 2023. – 102 с..

Наведено методичні рекомендації до виконання циклу лабораторних робіт з програмування та алгоритмічних мов в першому семестрі. Представлені теоретичні відомості щодо стандартних типів даних, основних конструкцій мови, масивів, рядків, функцій, розглянуті приклади реалізації мовою C++. Запропоновано набори варіантів індивідуальних завдань різної складності до кожної теми.

Навчальний посібник призначений для здобувачів ступеня бакалавр за спеціальністю 124 «Системний аналіз». Буде також корисним для студентів інших спеціальностей галузі знань 12 Інформаційні технології.

УДК 004.42/.43(076)

Реєстр. № НП 23/24-249. Обсяг 0,9 авт. арк.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
проспект Берестейський, 37, м. Київ, 03056
<https://kpi.ua>

Свідectво про внесення до Державного реєстру видавців, виготовлювачів і розповсюджувачів видавничої продукції ДК № 5354 від 25.05.2017 р.

© КПІ ім. Ігоря Сікорського, 2023

ЗМІСТ

Вступ.....	6
1. Мета та основні завдання робіт Лабораторного практикуму	7
2. Порядок виконання робіт	9
2.1 Загальні вимоги	9
2.2 Вимоги до етапів виконання кожної роботи	9
2.3 Загальні зауваження до оформлення програм	9
3. Оформлення звіту та порядок його подання	10
4. Заходи безпеки в комп'ютерному класі	11
5. Роботи лабораторного практикуму	13
5.1. Лабораторна робота № 1.	13
5.1.1. Тема	13
5.1.2. Мета роботи.	13
5.1.3. Основні теоретичні відомості.....	13
5.1.4. Приклад програми, що реалізує обчислення за лінійним алгоритмом з використанням математичних функцій	21
5.1.5. Порядок виконання роботи.....	23
5.1.6. Варіанти завдань	25
5.1.7. Запитання до самоконтролю.....	28
5.2. Лабораторна робота № 2.	29
5.2.1. Тема	29
5.2.2. Мета роботи.	29
5.2.3. Основні теоретичні відомості.....	29
5.2.4. Приклад програми з розгалуженням.....	33
5.2.5. Порядок виконання роботи.....	34
5.2.6. Варіанти завдань	35
5.2.7. Варіанти завдань для захисту в аудиторії	40
5.2.8. Запитання до самоконтролю.....	42
5.3. Лабораторна робота № 3.	43
5.3.1. Тема	43

5.3.2.	Мета роботи.....	43
5.3.3.	Основні теоретичні відомості.....	43
5.3.4.	Приклад програми з циклами	45
5.3.5.	Порядок виконання роботи.....	46
5.3.6.	Варіанти завдань	46
5.3.7.	Запитання до самоконтролю.....	54
5.4.	Лабораторна робота № 4.	55
5.4.1.	Тема	55
5.4.2.	Мета роботи.....	55
5.4.3.	Основні теоретичні відомості.....	55
5.4.4.	Приклад програми роботи з масивом	56
5.4.5.	Порядок виконання роботи.....	58
5.4.6.	Варіанти завдань	59
5.4.7.	Запитання до самоконтролю.....	67
5.5.	Лабораторна робота № 5.	69
5.5.1.	Тема	69
5.5.2.	Мета роботи.....	69
5.5.3.	Основні теоретичні відомості.....	69
5.5.4.	Порядок виконання роботи.....	70
5.5.5.	Варіанти завдань	71
5.5.6.	Запитання до самоконтролю.....	74
5.6.	Лабораторна робота № 6.	75
5.6.1.	Тема	75
5.6.2.	Мета роботи.....	75
5.6.3.	Основні теоретичні відомості.....	75
5.6.4.	Приклади програм опрацювання символічної інформації	81
5.6.5.	Порядок виконання роботи.....	82
5.6.6.	Варіанти завдань	83
5.6.7.	Запитання до самоконтролю.....	87
5.7.	Лабораторна робота № 7.	88

5.7.1. Тема	88
5.7.2. Мета роботи.....	88
5.7.3. Основні теоретичні відомості.....	88
5.7.4. Приклад з використанням вказівника на функцію.....	90
5.7.5. Порядок виконання роботи.....	91
5.7.6. Варіанти завдань	91
5.7.7. Запитання до самоконтролю.....	98
6. Список рекомендованої літератури.....	99
6.1. Базова	99
6.2. Допоміжна	99
6.3. Інформаційні ресурси	100
Додаток А. Титульна сторінка до звіту.....	101

ВСТУП

Роботи лабораторного практикуму до кредитного модуля «Алгоритмізація та основи програмування» дисципліни «Програмування та алгоритмічні мови» призначені для виконання здобувачами 1 курсу спеціальності 124 «Системний аналіз» галузі знань 12 «Інформаційні технології» денної форми навчання.

Основна мета виконання робіт лабораторного практикуму – це закріплення знань, отриманих на лекційних заняттях і в результаті самостійної роботи здобувача, та отримання навичку програмування.

Практикум складений із семи лабораторних робіт мовою C++, які можуть бути виконані самостійно, розглянуті онлайн або виконані в аудиторії. Включає теоретичні відомості щодо стандартних типів даних, основних конструкцій мови, масивів, рядків і функцій, проілюстровані прикладами реалізації відповідних програм. До кожної лабораторної роботи запропоновано набір варіантів індивідуальних завдань різної складності.

1. МЕТА ТА ОСНОВНІ ЗАВДАННЯ РОБІТ ЛАБОРАТОРНОГО ПРАКТИКУМУ

Метою проведення робіт є закріплення отриманих на лекціях та в результаті самостійної підготовки теоретичних відомостей, формування у здобувачів здатностей розробки програмного забезпечення з використанням стандартних алгоритмів у рамках підходів структурного програмування на базі мови програмування C++ у стандартному середовищі програмування; практичного досвіду роботи з базовими статичними структурами даних; застосування основних прийомів перевірки працездатності розробленого програмного продукту та аналізу отриманих результатів.

Згідно з вимогами силабусу кредитного модуля студенти після виконання даного циклу робіт мають продемонструвати такі результати навчання.

Знання загальної структури програми мовою програмування C++, загальних вимог до програмних продуктів, особливостей представлення мовою C++ даних простих та структурованих типів, операцій над ними, можливості перетворень типів; принципів використання директив препроцесора та засобів стандартної бібліотеки; основних конструкцій мови C++, та їх використання у базових алгоритмах; принципів побудови та використання функцій, вказівників на функції; загальні правила роботи із сучасним стандартним програмним забезпеченням; основні етапи розробки закінченого програмного продукту та технології оформлення результатів роботи.

Уміння програмно реалізувати розроблений алгоритм мовою програмування C++ з використанням структурного підходу; користуватись стандартним середовищем розробки; коректно інтерпретувати одержані результати; правильно оформити супровідну документацію відповідно до вимог чинного законодавства України зі стандартизації у вигляді звіту.

Набути досвід застосування стандартних алгоритмів для розробки закінченого програмного продукту мовою програмування C++ з використанням стандартних середовищ розробки.

2. ПОРЯДОК ВИКОНАННЯ РОБІТ

2.1 Загальні вимоги

При виконанні лабораторних робіт (ЛР) необхідно дотримуватись наступних вимог:

1. При роботі в лабораторії чи комп'ютерному класі слід дотримуватись правил техніки безпеки.
2. Ознайомитись з основами роботи у середовищі Dev C++, освоїти використання системи довідки. При роботі в режимі online, встановити на своєму робочому комп'ютері необхідне програмне забезпечення. В залежності від операційної системи це може бути Visual Studio від Microsoft, Code::Blocks, Xcode, інше середовище для створення консольних додатків.

2.2 Вимоги до етапів виконання кожної роботи

Етапи виконання кожної ЛР можна звести до наступного алгоритму:

1. проаналізувати умову задачі згідно з номером варіанту;
2. побудувати математичну модель та розробити алгоритм;
3. відповідно до розробленого алгоритму створити програму;
4. для доведення працездатності програми продумати тестові приклади;
5. для зручності використання програми користувачем забезпечити дружній інтерфейс;
6. результати роботи оформити у вигляді звіту.

2.3 Загальні зауваження до оформлення програм

Щоб програму було легко читати та тестувати, потрібно:

1. широко використовувати коментарі;
2. обирати ідентифікатори так, щоб вони відображували зміст констант, змінних, функцій;
3. використовувати пропуски для виділення структури програми;
4. кожному вводу даних повинно передувати повідомлення, які данні треба ввести і у якому вигляді.

3. ОФОРМЛЕННЯ ЗВІТУ ТА ПОРЯДОК ЙОГО ПОДАННЯ

Для кожної лабораторної роботи має бути підготовлений звіт. Без звіту студент не допускається до здачі відповідної ЛР.

Звіт до ЛР має містити:

1. формулювання теми роботи;
2. постановку завдання, яке необхідно виконати;
3. текст програми;
4. приклади тестів для перевірки працездатності програми;
5. висновки щодо роботи програми.

Шаблон титульної сторінки до звіту наведено у додатку 1.

Текст звіту виконується у відповідності до вимог стандартів машинним (на пристроях ЕОМ) способом. Формат аркуша – А4. Параметри сторінки (поля): верхнє – 25 мм, нижнє – 15 мм, лівє – 25 мм, правє – 10 мм. Основний текст набирається шрифтом Times New Roman, розмір 14, міжрядковий інтервал – 1,5.

Для формулювання тексту умови може бути використаний засіб Microsoft Equation, для виконання діаграм дій може бути використаний засіб «Автофігури». Вставлення результатів роботи програми забезпечити вставкою із сеансу DOS.

У висновках до ЛР необхідно зазначити, чи вирішує програма поставлене завдання, вказати труднощі, які виникли під час його виконання.

4. ЗАХОДИ БЕЗПЕКИ В КОМП'ЮТЕРНОМУ КЛАСІ

До виконання лабораторних робіт допускаються студенти, які пройшли інструктаж з техніки безпеки з наступним записом про це в спеціальному журналі.

Загальні положення:

1. До виконання лабораторних робіт допускаються студенти, які пройшли інструктаж з техніки безпеки і правил поведінки в комп'ютерному класі.

2. Робота студентів в комп'ютерному класі дозволяється лише у присутності викладача.

Перед початком роботи необхідно:

1. Переконавшись у відсутності видимих пошкоджень на робочому місці;

2. Розмістити на столі зошити, навчальні посібники так, щоб вони не заважали роботі на комп'ютері;

При роботі в комп'ютерному класі категорично забороняється:

1. Знаходитися в класі у верхньому одязі;
2. Класти одяг, сумки та інші речі на пристрої;
3. Знаходитися в класі з напоями та їжею;
4. Розташовуватися збоку або ззаду від включеного монітора;
5. Приєднувати або від'єднувати кабелі, торкатися роз'ємів, оголених дротів і розеток, пересувати комп'ютери і монітори, відкривати системний блок;
6. Включати і вимикати комп'ютери самостійно;
7. Намагатися самостійно усувати несправності в роботі обладнання;
8. Перекривати вентиляційні отвори на системному блоці та моніторі;
9. Видаляти і переміщати чужі файли;
10. Приносити і запускати комп'ютерні ігри.

Знаходячись в комп'ютерному класі, здобувач зобов'язаний:

1. Дотримуватись тиші і порядку;

2. Виконувати вимоги викладача;
3. Знаходячись в мережі працювати лише під своїм ім'ям і паролем;
4. Дотримуватись режиму роботи (згідно Санітарних правил і норм);
5. При появі різі в очах, різкому погіршенні видимості, неможливості сфокусувати погляд або навести його на різкість, появи болю в пальцях і кистях рук, посилення серцебиття негайно покинути робоче місце, повідомити про це, викладача і звернутися до лікаря;
6. Після закінчення роботи завершити всі активні програми і коректно вимкнути комп'ютер;
7. Залишити робоче місце чистим.

Вимоги безпеки в аварійних ситуаціях:

1. При появі програмних помилок або збоїв обладнання здобувач повинен негайно звернутися до викладача.
2. При появі запаху гару, незвичайного звуку негайно припинити роботу, і повідомити викладача.

5. РОБОТИ ЛАБОРАТОРНОГО ПРАКТИКУМУ

5.1. Лабораторна робота № 1.

5.1.1. Тема

Структура програми мовою C++. Засвоєння прийомів роботи в інтегрованому середовищі розробки. Проста обчислювальна програма.

5.1.2. Мета роботи.

Навчитись синтаксично правильно будувати програму мовою програмування C++, коректно користуватися можливостями середовища програмування для її запуску та отримання результату роботи; навчитися складати та вміти перевірити семантику простої обчислювальної програми, що реалізує лінійний алгоритм.

5.1.3. Основні теоретичні відомості

Складові мови програмування C++. Основні складові будь-якої алгоритмічної мови – це символи, лексеми, вирази і оператори, які складають власне програму.

Символи – це основні неподільні елементи мови. Представляють собою знаки, які складають усі конструкції мови. Символи утворюють **алфавіт** мови. До алфавіту мови C++ входять символи, визначені в таблиці кодування **ASCII**, а саме:

- великі та малі латинські літери (розрізняються компілятором);
- цифри 0-9;
- знаки пунктуації;
- деякі інші символи.

Таблиця 5.1. Знаки пунктуації

Символ	Назва	Символ	Назва
,	кома	)	права кругла дужка
.	крапка	(	ліва кругла дужка
;	крапка з комою	{	права фігурна дужка
:	двокрапка	}	ліва фігурна дужка
?	знак питання	<	менше

'	апостроф	>	більше
!	знак оклику	]	права квадратна дужка
	вертикальна риса	[	ліва квадратна дужка
/	дробова риса (слеш)	#	решітка
\	зворотна риса (зворотний слеш)	%	процент
~	тільда	&	амперсанд
*	зірочка	^	виключне «або»
+	плюс	=	дорівнює
-	мінус	"	лапки
_	знак підкреслення		

Лексема – мінімальна одиниця мови, яка має власний зміст та обробляється транслятором. Лексеми формують **вирази**. **Програма** розглядається як послідовність виразів і лексем.

У мові C++ кожна лексема належить одній з наступних категорій:

- зарезервовані (ключові) слова;
- ідентифікатори;
- літерали;
- знаки операцій;
- розділювачі.

Зарезервовані слова (всього 84 слова, включно з C++17 і C++11)[1]:

alignas	decltype	namespace	struct
alignof	default	new	switch
and	delete	noexcept	template
and_eq	do	not	this
asm	double	not_eq	thread_local
auto	dynamic_cast	nullptr	throw
bitand	else	operator	true
bitor	enum	or	try
bool	explicit	or_eq	typedef
break	export	private	typeid
case	extern	protected	typename
catch	false	public	union
char	float	register	unsigned
char16_t	for	reinterpret_cast	using
char32_t	friend	return	virtual
class	goto	short	void
compl	if	signed	volatile
const	inline	sizeof	wchar_t
constexpr	int	static	while
const_cast	long	static_assert	xor
continue	mutable	static_cast	xor_eq
auto	double	int	struct

break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

Ідентифікатор – це лексема, яка складена із літер латинського алфавіту, цифр, символів «_» (підкреслення) та «\$» (знак долару), і яку можна присвоїти програмним об'єктам, тобто змінним, іменованим константам, типам та функціям.

Інші символи, крім перелічених, в ідентифікаторах недопустимі.

Великі і малі букви в ідентифікаторах відрізняються (мова C++ чутлива до регістру).

Ідентифікатор не може співпадати з зарезервованим словом.

Ідентифікатор не може містити пропуск.

Літерали – це значення констант різних типів, які зустрічаються у тексті програми. Зовнішній вигляд літералів залежить від типу, до якого вони належать. Наприклад, 23 – ціле int, 45.6 або 0.5e-3 – дійсні double, 'g' – символ char, "some information" – рядковий літерал.

Знаки операцій побудовані на основі набору спеціальних символів і букв алфавіту.

Таблиця 5.2. Знаки операцій

,	!	!=		=	%	%=	&	&&
&=	()	*	*=	+	++	+=	-	--
--=	->	->*	.	.*	/	/=	<	<<
<=	<<=	>	>>	>=	>>=	==	?:	[]
^	^=	~		#	##	sizeof		

Розділювачі – це ';' крапка з комою, '{}' фігурні дужки, '...' трикрапка.

Множина операцій та розділювачів фіксована. Лексеми відокремлюються так званими пробільними символами, кількість яких необмежена. Це власне пропуск, ENTER, знаки табуляції.

Інші символи можуть зустрічатись лише у коментарях чи текстових виразах. Багаторядковий коментар починається з символів /* і закінчується символами */, однорядковий коментар починається з символів // і закінчується в кінці рядка. Наприклад:

```
/* це багаторядковий
коментар
*/
// це однорядковий коментар
```

Однорядковий коментар може знаходитись всередині багаторядкового.

Вираз - це сукупність операндів, об'єднаних знаками операцій. Операндом можуть бути: константи, змінні, елементи масивів, функції.

Порядок виконання операцій залежить від пріоритету операцій і може бути змінений за допомогою дужок ().

Загальна структура програми. Програма мовою C++ складається з директив препроцесора, зовнішніх описів і функцій і має таку структуру:

```
<директиви препроцесора>
<описи>
<функції>
```

У будь-якому місці програма може мати коментарі.

Директиви препроцесора вказують на перетворення, які потрібно зробити з текстом програми. За їх допомогою також здійснюється оголошення іменованих констант, що використовуються у всій програмі, підключення інформації про стандартну бібліотеку та ін.

В розділі **описів** роблять глобальні оголошення.

Функції – це основні складові програми. Представляють собою синтаксично та логічно завершений самостійний фрагмент програми, призначений для розв'язання певної задачі. Складаються із послідовності

операторів у фігурних дужках, кожний з яких відокремлений символом ';'.
Синтаксис оголошення функції:

```
тип ідентифікатор([список_параметрів])  
{  
    оператори_функції;  
};
```

Програма може містити декілька функцій, а може – лише одну, але обов’язково функцію з назвою `main()`.

Оператори бувають виконувані (задають дії над даними) і невиконувані (задають описи даних). Відокремлюються символом ';'.
Приклад. Проста програма мовою C++:

```
#include <stdio.h>          // директива препроцесора  
int main(){  
    printf("hello, world\n");//виклик функції виведення  
    return 0;  
}
```

Виконання програми завжди починається з функції `main()`. Із `main()` для виконання різних операцій можуть бути викликані інші функції, які або знаходяться у стандартних бібліотеках, або мають бути написані самим програмістом.

Перший рядок програми

```
#include <stdio.h>
```

– це директива препроцесора, яка вказує на необхідність додати до програми інформацію про стандартну бібліотеку. У даному випадку, `stdio.h` – заголовковий файл, що містить оголошення функцій введення-виведення, у тому числі і функцію `printf` із прикладу.

Один з методів передачі даних між функціями полягає в тому, що одна функція викликає іншу і надає їй список змінних, які називають аргументами. Цей список розташований у круглих дужках одразу після імені функції. У прикладі функція `main` викликає `printf` з аргументом `"hello, world\n"`.

Функція `printf` виводить свої аргументи на екран або на друк. Функція `main` не має аргументів, на що вказує порожній список в дужках: `int main()`.

У наведеному прикладі ніяких обчислень не було, тому змінні і константи не були використані.

Типи даних Типи даних визначають обсяг пам'яті, що виділяється під данні, спосіб представлення даних в комп'ютері та їх інтерпретацію, діапазон значень, перелік операцій та функцій, що можуть бути використані для обробки даних. Визначені у мові C цілі типи зведені у таблицю 5.3, дійсні – 5.4.

Таблиця 5.3 Цілі типи

Тип	Назва типу	Діапазон значень	Розмір	Знак
<code>short int</code>	коротке ціле	-32768(-8000)..32768(+7FFF)	2 байта	із знаком
<code>unsigned short int</code>	коротке ціле без знаку	0(0)... 65535(FFFF)	2 байта	без знаку
<code>int</code>	Ціле	-80000000..+7FFFFFFF	4 байта	із знаком
<code>unsigned int</code>	ціле без знаку	0..+FFFFFFF	4 байта	без знаку
<code>long int</code>	Довге ціле	[-2 147 483 647, +2 147 483 647]	4 байта	із знаком
<code>unsigned long int</code>	Довге ціле без знаку	[0, +4 294 967 295]	4 байта	без знаку
<code>long long int</code>	стандарт C99	$-(2^{63}-1) \dots (2^{63}-1)$	8 байт	із знаком
<code>unsigned long long int</code>	стандарт C99	$0-(2^{64}-1)$	8 байт	без знаку

Арифметичні операції над даними цілого типу:

+	додавання	
-	віднімання, також унарний мінус	
*	множення	
/	цілочисельне ділення:	$13/3 = 4$
%	остача від ділення:	$13\%3 = 1$
<code>++x;</code>	префіксний інкремент	<code>x=x+1;</code>
<code>x++;</code>	постфіксний інкремент	<code>x=x+1;</code>

--x; префіксний декремент x=x-1;

x--; постфіксний декремент x=x-1;

Функції цілого аргументу, стають доступні, якщо підключити заголовковий файл з математичними функціями <math.h>:

abs(x); |x|

pow(x, 2); x²

Порозрядні операції:

& – побітове І – бінарна операція, дія якої еквівалентно застосування логічного І до кожної пари бітів, які стоять на однакових позиціях у двійкових представленнях операндів.

Приклад:

& (І)	0011
	0101
	0001

| – побітове АБО – бінарна операція, дія якої еквівалентно застосування логічного АБО до кожної пари бітів, які стоять на однакових позиціях у двійкових представленнях операндів.

Приклад:

(АБО)	0011
	0101
	0111

^ – виключне АБО – бінарна операція, результат дії якої дорівнює 1, якщо сума бітів, які стоять на однакових позиціях у двійкових представленнях операндів, непарна, або 0, якщо парна.

Приклад:

^ (Викл.АБО)	0011
	0101
	0110

~ НЕ (заперечення, доповнення до 1)

Приклад:

~ (НЕ)	0110
	1001

>> зсув вправо, в напрямку від старшого біта до молодшого.

Приклад:

x>>2	0100
	0001

<< зсув вліво, в напрямку від молодшого біта до старшого.

Приклад:

x<<1	0011
	0110

Оператор надання:

<змінна>=<вираз>/<значення>

Таблиця 5.4 Типи даних з плаваючою крапкою

Тип	Точність (кількість знаків)	Розмір (байт)	Діапазон
float	6	4	1,175494351 E – 38 ... 3,402823466E+38
double	16	8	2,2250738585072014 E – 308 ... 1,7976931348623158 E + 308

Форма представлення називається експоненційною (або з плаваючою крапкою) $\pm mE\pm p$, де m - мантиса, p – порядок.

Приклад: $4.2E+02 = 4200 = 4.2 \cdot 10^2$

Арифметичні операції для даних з плаваючою крапкою

+	додавання
-	віднімання
*	множення
/	ділення

Функції бібліотеки <math.h>:

fabs (x) ;	x
pow (x, 2) ;	x^2
cos (x) ;	cos(x)
sin (x) ;	sin(x)
atan (x) ;	arctg(x)
exp (x) ;	e^x
log (x) ;	ln x
sqrt (x) ;	$\sqrt{x}$
modf (x) ;	дробова частина x

`round(x);` ціла частина x

Можна будувати складні функції:

`pow(sin(x),2)` – $\sin^2 x$;

Дані з плаваючою крапкою можна привести до цілого типу:

`long int lround(double x);` округлення x до найближчого цілого.

Складене надання або *стенографічне (shorthand) надання*. Складені оператори надання існують для всіх бінарних операцій (тобто операцій, що мають два операнда). Будь-який оператор виду:

<змінна> = <змінна><оператор><вираз>

можна записати, як

<змінна> <оператор>= <вираз>

Приклади:

`x += y;` те саме, що й: `x = x + y;`

`x -= y;` те саме, що й: `x = x - y;`

`x *= y;` те саме, що й: `x = x * y;`

`x /= y;` те саме, що й: `x = x / y;`

`x %= y;` те саме, що й: `x = x % y;`

`x &= y;` те саме, що й: `x = x & y;`

`x |= y;` те саме, що й: `x = x | y;`

`x ^= y;` те саме, що й: `x = x ^ y;`

`x >>= y;` те саме, що й: `x = x >> y;`

`x <<= y;` те саме, що й: `x = x << y;`

5.1.4. Приклад програми, що реалізує обчислення за лінійним алгоритмом з використанням математичних функцій

З клавіатури увести значення змінних x, y . Обчислити значення змінної a за формулою:

$$a = \frac{\sqrt{|x + 4,62|} - y}{1 + 2x^2 + y^2}$$

Перший варіант розв'язку, рекомендований для КП1, передбачає використання *функцій* введення-виведення:

```
#include <conio.h>
#include <math.h>
#include <stdio.h>
#define SQU 2

int main()
{
    float x,y,a;
    printf("введіть значення x і y через пропуск\n");
    scanf("%f%f",&x, &y);
    a=(sqrt(abs(x+4.62))-y)/(1+2*pow(x,SQU)+pow(y,SQU));
    printf("шукане значення a = %f",a);
    getch();
    return 0;
}
```

Підключення заголовкового файлу для роботи з консоллю `#include <conio.h>` потрібне тільки у випадку необхідності використання функції `getch()`; яка забезпечує призупинення виконання програми до натиснення будь-якої клавіші, щоб користувач встиг побачити результат. Використання цієї можливості зайве, якщо працювати, наприклад, в Visual Studio від Microsoft, де перед завершенням роботи програми за умовчанням уже передбачене очікування натиснення будь-якої клавіші.

У розглянутому прикладі було використано ще одну директиву – `#define`.

Тут – для визначення так званої іменованої константи `SQU`. У загальному випадку синтаксис цієї директиви наступний:

`#define ідентифікатор_константи значення`

Діятиме з моменту оголошення і до кінця коду або поки дія константи не буде скасована директивою `#undef`. Синтаксис:

`#undef ідентифікатор_константи`

Зверніть увагу: ПІСЛЯ ДИРЕКТИВИ крапка з комою як закінчення оператора, НЕ потрібна!

Другий варіант програми, рекомендований для наступних після КП1 робіт, передбачає іншу реалізацію введення-виведення інформації.

```
#include <math.h>
#include <iostream>
#define TWO 2

int main()
{
    double x,y,a;
    std::cout << "введіть значення x і y через пропуск"<<endl;
    std::cin >> x >> y;
    a=(sqrt(fabs(x+4.62))-y)/(1+2*pow(x,TWO)+pow(y,TWO));
    std::cout<<"шукане значення a ="<< a << std::endl;
    return 0;
}
```

Тут підключений заголовковий файл `<iostream>`, який містить об'єкти введення `cin` і виведення `cout` з визначеними для них операціями відповідно `>>` для введення та `<<` для виведення інформації. Оскільки `<iostream>` визначений в просторі імен `std`, то розширення `.h` в його назві в директиві `#include` писати не потрібно, а для використання визначених в ньому конструкцій, потрібно це явно вказувати у зверненні до них, як от `std::cin` та `std::cout`.

Якщо ж скористатися `using`-директивою і перед `main()` вставити `using namespace std;` то ставити `std::` перед кожним об'єктом цього простору імен в тексті програми не потрібно.

В обох варіантах реалізації програми введення даних необхідно здійснювати через пропуск або натисненням клавіші `enter`.

5.1.5. Порядок виконання роботи

Виконання ЛР1 розділено на два етапи. Виконання першого етапу – побудова працездатної програми відповідно варіанту із таблиці 5.5 безпосередньо в аудиторії під керівництвом викладача для ознайомлення з роботою в середовищі програмування.

Таблиця 5.5 Варіанти завдань для аудиторної роботи

1	$R = \ln(a^2 + b^2 + c^2) + e^{a+1}$	2	$R = \arcsin a^2 + \arccos(bx - a) + e^{bx}$
3	$R = (x + 2,45) + \sqrt{ x + y^2 }$	4	$R = 2\sin(0,214y^5 + 1)$
5	$R = \sqrt{ x^2 + 5y^2 + 0,12 } + 37,5$	6	$R = \ln(y + 0,95) + \sin x^4$

7	$R = \sin(a * \arcsin x + \ln y) - \cos(2ax)$	8	$R = \sin x + e^{2y}$
9	$R = \sqrt{(2+y)^2 + \sin^2(y+5)} + \ln(y+5) - y^3$	10	$R = \cos 4x^2 + \sqrt{y^2 + 28.61}$
11	$R = \operatorname{tg}^3 y + \sin^5 x \sqrt{b-c}$	12	$R = \cos^2 bx^5 - (\sin a^2 + \cos(x^3 + z^5 - a^2))$
13	$R = \operatorname{tg}(y^3 - h^4) + h^2 - \sin^3 h + y$	14	$R = \frac{e^{2y} + \sin x}{\ln(3.8y + x)}$
15	$R = \ln x + \frac{3.5x^2 + 1}{\cos 2y}$	16	$R = \frac{\ln(x-y) + y^4}{e^y + 2.33x^2}$
17	$R = \frac{x^2 + 2.8x + 0.35}{\cos 2y + 3.5}$	18	$R = \frac{4z^2 + \ln 2}{e^{x+z} + 2.7 \sin x}$
19	$R = \frac{x^2 - 0.1z + \sqrt{2.5y^2}}{3.7x^2 + \cos^2 y}$	20	$R = 4.35y^3 + \frac{1 + 2t \ln t}{\sqrt{\cos^2 2y + 4.35t^2}}$
21	$R = \frac{2t + y \cos t}{\sqrt{2.8t^2 + (y-1)^2}}$	22	$R = \frac{\sin(2t+1)^2 + 0.3}{\ln(t+y)}$
23	$R = \frac{\sin(p + 0.25)^2}{y^2 + 7.32p}$	24	$R = \frac{\sin(2x+y) + y^2}{e^y + x}$
25	$R = \frac{0.004a + e^{2b}}{e^{\frac{b}{2}}}$	26	$R = e^{y+2.5} + 7h^3 \frac{1}{\ln \sqrt{y^2 + 0.04h}}$
27	$R = \frac{0.33h^2 - 4.44}{e^{y+h} + \sqrt{2.7y}}$	28	$R = \frac{2 \sin(0.35y + 1)}{\ln(y + 2t)}$
29	$R = \frac{\cos^3 y + 2^x d}{e^y + \ln(\sin^2 x + 4.2)}$	30	$R = \frac{\ln(x^3 + y) - y^4}{e^y + 5.4k^3}$

Другий етап виконання РКП1 – підготовка і виконання другої програми відповідно варіанту п. 5.1.6. Для того, щоб здати цю роботу, необхідно підготувати звіт. Вимоги до звіту надані в розділі 3.

Доведення працездатності програми буде відбуватися на основі тестування роботи програми на підготовлених і перевірених здобувачем прикладах.

5.1.6. Варіанти завдань

1. Дано x, y, z , обчислити:

$$a = \frac{\sqrt{x-1} - \sqrt[3]{|y|}}{1 + \frac{x^2}{2} + \frac{y^2}{4}}; \quad b = x(\operatorname{arctg} z + e^{-(x+3)}).$$

2. Дано x, y, z обчислити:

$$a = \frac{3 + e^{y-1}}{1 + x^2 |y - \operatorname{tg} z|}; \quad b = 1 + |y - x| + \frac{(y - x)^2}{2} + \frac{|y - x|^2}{3}.$$

3. Дано x, y, z , обчислити:

$$a = (1 + y) \frac{x + y/(x^2 + 4)}{e^{-x-2} + 1/(x^2 + 4)}; \quad b = \frac{1 + \cos(y - 2)}{x^4/2 + \sin^2 z}.$$

4. Дано x, y, z обчислити:

$$a = y \frac{x}{y^2 + \left| \frac{x^2}{y + x^3/3} \right|}; \quad b = \left(1 + \operatorname{tg}^2 \frac{z}{2} \right).$$

5. Дано x, y, z обчислити:

$$a = \frac{2 \cos(x - \pi/6)}{1/2 + \sin^2 y}; \quad b = 1 + \frac{z^2}{3 + x^2/5}.$$

6. Дано x, y, z обчислити:

$$a = \frac{1 + \sin^2(x + y)}{2 + \left| x - 2x/(1 + x^2 y^2) \right|} + x; \quad b = \cos^2 \left(\operatorname{arctg} \frac{1}{z} \right).$$

7. Дано x, y, z обчислити:

$$a = \ln \left| \left(y - \sqrt{x} \right) \left(x - \frac{y}{z + x^2/4} \right) \right|; \quad b = x - \frac{x^2}{3!} + \frac{x^5}{5!}.$$

8. Дано c, d ; знайти:

$$\left| \frac{\sin^3 |cx_1^3 + dx_2^2 - cd|}{\sqrt{(cx_1^3 + dx_2^2 - x_1)^2 + 3.14}} \right|,$$

де x_1, x_2 – це більший і менший корені рівняння: $x^2 - 3x - cd = 0$

9. Дано c, d ; знайти:

$tg(cx_1^3 + dx_2^2 - x_1)$, де x_1, x_2 відповідно більший та менший дійсні

корені рівняння: $x^2 - 3x - cd = 0$

10. Знайти кути трикутника із сторонами a, b, c

11. За сторонами трикутника a, b, c . Знайти:

1) довжини висот;

2) довжини медіан.

12. Дано сторони трикутника a, b, c . Знайти:

1) довжини бісектрис;

2) радіуси вписаного та описаного кола.

13. Є гіпотенуза та катет прямокутного трикутника. Знайти:

1) другий катет;

2) радіус вписаного кола.

14. Трикутник задано координатами вершин $(x_1, y_1)(x_2, y_2)(x_3, y_3)$.

Знайти:

1) периметр трикутника

2) площу трикутника.

15. Надано дві сторони трикутника та кут між ними в градусах. Знайти довжину третьої сторони та площу трикутника.

16. Дано x, y, z . Знайти:

$$\max(x + y + z, xyz); \quad \min^2(x + y + z/2, xyz) + 1.$$

17. Дано h . Визначити, чи має рівняння: $ax^2 + bx + c = 0$ дійсні корені, якщо:

$$a = \left(\frac{|\sin 8h| + 17}{(1 - \sin 8h \cos(h^2 + 18))^2} \right)^{1/2}, \quad b = 1 - \left(\frac{3}{3 + |tg ah^2 - \sin ah|} \right)^{1/2},$$

$$c = ah^2 \sin 6h + 6h^3 \cos ah.$$

18. Вивести комплексне число у вигляді $RE + i \cdot IM$, дійсна та уявна частини якого визначаються за формулами:

$$RE = \frac{|\sin 8h| + 17}{(1 - \sin 8h \cos(h^2 + 18))^2},$$

$$IM = 1 - \left(\frac{3}{3 + |tg h^2 - \sin h|} \right)^{1/2}, \text{ де } h - \text{надане.}$$

19. Для заданих a, b обчислити:

$$(a + b) \sin a + (a - b) \cos b + \frac{a - b}{\sqrt{\sin a + \cos b}}.$$

20. Для наданого x знайти:

$$\frac{x+1}{x-1} + 3.6(x - (\sin x + 1)^2 + x^2(\sin x - 1)).$$

21. Для наданих x, y, z знайти:

$$a = \frac{xyz - 3.3|x + \sqrt[4]{y}|}{10^7 + \sqrt{\lg 4!}}; \quad b = e^{|x-y|} + \ln(1+e) - z.$$

22. Для 5 наданих елементів обчислити:

$$p = \frac{a_1 + \sin^2 a_2 - \arctg(1 + \ln a_3)}{\sqrt{a_4^2 + a_5^2}}.$$

23. Надано числа a_1, a_2 . Обчислити a_3, a_4 за такими виразами:

$$a_3 = \frac{\max^2(a_1, a_2) + e^{a_2} \sin a_1}{\sqrt{\min(a_1, a_2) + a_1^2 + a_2^2}}; \quad a_4 = \ln|a_1| - tg^2 a_2 \sin a_1.$$

24. Обчислити площу поверхні піраміди, гранню якої є рівносторонній трикутник із стороною A .

25. Знайти дробову частину середнього геометричного та цілу частину середнього арифметичного п'яти дійсних чисел, введених з клавіатури. масиву A (5).

26. Знайти усі цифри цілого числа a і вивести їх у стовпчик.

27. Знайти суму цифр введеного з клавіатури цілого числа.
28. Знайти число, що утворюється перестановкою у зворотному порядку цифр наданого чотиризначного числа.
29. Дано 5 дійсних чисел. Знайти цілу та дробову частини їх суми.
30. Трикутник задано у вигляді координат вершин. Знайти довжину його сторін та периметр.
31. Надано два кути трикутника та сторона між ними. Знайти інші сторони та кути.
32. Змішано v_1 літрів води температури t_1 з v_2 літрами води температури t_2 . Знайти об'єм та температуру суміші.
33. Визначити периметр правильного n -кутника, описаного навколо кола радіусом r .
34. Знайти час, через який зустрінуться два тіла, що рухаються з постійним прискоренням, якщо відомі їх початкові швидкості та прискорення і відстань між тілами.
35. Перевести координати точки із декартової до полярної системи координат.

5.1.7. Запитання до самоконтролю

1. Основні складові мови програмування. Склад алфавіту.
2. Що таке лексеми? Які різновиди лексем ви знаєте? Надати характеристику.
3. Правила формування ідентифікаторів.
4. Що таке зарезервовані слова? Для чого потрібно їх знати?
5. Перерахуйте можливі складові виразів?
6. Опишіть загальну структуру програми.
7. Що таке тип даних? Які відміни між цілими типами та типами з плаваючою крапкою?
8. Що таке базовий тип? Модифікатор типу?

5.2. Лабораторна робота № 2.

5.2.1. Тема

Основні конструкції мови програмування C++. Реалізація розгалужень

5.2.2. Мета роботи.

Навчитися правильно реалізовувати алгоритми з розгалуженнями засобами мови C++.

5.2.3. Основні теоретичні відомості

Логічний тип даних. В мові C++ для визначення істинності події введений тип даних `bool` (від англійського `Boolean`). Змінна такого типу може приймати тільки два значення: `true` (істинне значення, правда, так, 1) та `false` (хибне значення, неправда, ні, 0).

В мові C такого типу ще не було і для позначення істинності використовували цілий тип. Якщо змінна, яка за своїм положенням у виразі, повинна була мати логічне значення, приймала значення нуль (0), то це відповідало `false`, тобто було хибним. Якщо ж ця змінна не дорівнювала нулю, то це відповідало `true`, тобто істині.

Таке трактування цілого числа у виразі, який повинен мати логічне значення, зберіглося й у C++.

Операції порівняння Це операції, в яких значення двох змінних порівнюються один з одним.

<code>></code>	більше ніж
<code>>=</code>	більше або дорівнює
<code><</code>	менше ніж
<code><=</code>	менше або дорівнює
<code>==</code>	дорівнює
<code>!=</code>	не дорівнює

Наприклад:

```
bool a=5>2, b=15<=10;           //a - true, b - false
int c=35==45, g=6<7;           //c - 0, g - 1
```

Логічні операції засобами мови програмування C++ реалізують операції формальної логіки. Операндом логічної операції є логічний вираз.

! логічне НІ – унарна операція заперечення.

Таблиця істинності:

x	!x
0	1
1	0

Бінарні операції:

&& логічне І – кон’юнкція ($\wedge$)

|| логічне АБО – диз’юнкція ($\vee$)

Таблиця істинності виглядає так:

x	y	x&& y	x y
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

За допомогою логічних операцій можна утворити складений логічний вираз, наприклад:

```
bool a=5>2&&15<=10;           //a - false
int c=! (35==45)||0;           //c - 1
```

Умовні оператори. До умовних операторів відносять оператор умовного переходу `if` та оператор-перемикач `switch`.

Оператор if. Оператор умовного переходу `if` використовують для розгалуження процесу обчислень на два напрямки. Синтаксис оператора:

```
if (вираз) оператор_1;
else
    оператор_2;;
```

де вираз — це вираз, який має логічне значення (`true` — «істина» або `false` — «неправда»). Виконання оператору `if` реалізоване так: спочатку компілятор обчислить вираз і, якщо його значення не дорівнює нулю («істинне»), виконується оператор_1, у протилежному випадку —

оператор_2, після чого управління буде передано наступному за умовним оператору, наприклад:

```
if ( i < j ) i++;
else{
    j = i-3; //оператор_2 - складений
    i++;
};
cout<<"i= "<<i<<"", j= "<<j<<endl;
```

У загальному випадку вираз може не використовувати операцій порівняння, наприклад:

```
if(n++)m++;
else k++;
```

Скорочена форма if. Оператор if може мати скорочену форму запису, тоді у записі оператора друга частина (тобто else) відсутня. Для такого оператора, якщо вираз приймає хибне значення (або 0), управління одразу буде передане на наступний після if оператор програми, наприклад :

```
cin >> n;
if (n) n++; //якщо введено 0, то оператор_2 відсутній
cout<< n <<endl; // n++ не виконався, виведено 0
```

Якщо у будь-якій гілці розгалуження необхідно виконати декілька операторів, їх слід розташовувати у блоці, як у першому прикладі.

Синтаксис C++ припускає, що у випадку застосування вкладених умовних операторів кожне else відповідає найближчому до нього попередньому if. Наприклад, розглянуті фрагменти тотожні:

<pre>if (a > b){ if (a > c) max = a; else max = c; }</pre>	<pre>if (a > b) if (a > c) max = a; else max = c;</pre>
--	---

Тернарна операція. У деяких випадках замість оператора if зручніше використати тернарний оператор "?". Тобто оператор

if (вираз) оператор_1 [else оператор_2];

можна записати так:

вираз? оператор_1: оператор_2;

Приклад застосування тернарного оператора замість оператора if:

$y = (x < 9) ? 100 : 200;$ замість `if (x < 9) y=100;else y=200;`

У наведеному прикладі результати роботи обох фрагментів будуть тотожними.

Оператор-перемикач switch називають оператором вибору або множинного розгалуження і використовують для вибору одного із багатьох варіантів рішення. Синтаксис оператора:

```
switch (вираз_цілого_типу) {  
    case константний_вираз_1: оператор 1; [break];  
    case константний_вираз_2: оператор 2; [break];  
    ...  
    case константний_вираз_n: оператор n; [break];  
    [default: оператор n+1;]  
}
```

де **switch**, **case**, **default** — службові слова; **break** — оператор (необов'язковий) здійснює вихід із оператора **switch**; константні вирази повинні мати різні значення, не можуть містити змінних і викликів функцій; як правило, це ціла або символьна константа; оператор 1; ... – будь-які оператори мови.

Порядок виконання оператора: обчислене значення виразу_цілого_типу порівнюється (послідовно зверху донизу) зі значеннями константних виразів 1, ..., n. Якщо воно співпаде зі значенням константного виразу_K, наприклад, а наприкінці гілки розгалуження немає оператора **break**, то буде виконаний оператор_K та усі наступні або до завершення оператора **switch**, або доки не зустрінеться оператор **break** в одній із гілок. За наявності оператора **break** у вибраній гілці K, буде виконаний тільки оператор_K, і керування буде передане оператору, розташованому за межами оператора **switch**.

Якщо значення виразу_цілого_типу не збігається з жодним із значень константних виразів, то виконується гілка **default** і здійснюється вихід з оператора **switch**. Якщо ж у такому випадку обов'язкова гілка

default відсутня, одразу відбувається вихід із оператора switch і управління передається на наступний оператор програми.

Приклад використання оператора у повній формі :

```
switch (number)
{
    case 1:      cout<<number <<" - one \n";
                break;
    case 2:      cout<< number <<" - two \n";
                break;
    case 3:      cout<< number <<" - three\n";
                break;
    default:     cout<< "good bye \n";
}
```

Якщо значення number дорівнює, наприклад, 2, то буде виконана тільки гілка case 2 з виведенням «2 - two» на екран, після чого виконання оператора switch закінчиться.

5.2.4. Приклад програми з розгалуженням

Приклад 1. Обчислити функцію

$$F(x) = \begin{cases} \sqrt{x}, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

Тут обидва оператори оператор_1 і оператор_2 – прості.

```
#include <iostream>
#include <math.h>
using namespace std;

int main()
{
    double x, f;
    cout<<"введіть значення x = "<<endl;
    cin>>x;

    if (x>0) f=sqrt(x);
    else f=0;

    cout<<"функція дорівнює = "<<f<<endl;
    return 0;
}
```

Приклад 2. Обчислити функцію.

$$F(x) = \begin{cases} \sqrt{x}, & x \geq 0 \\ \text{немає розв'язку}, & x < 0 \end{cases}$$

Розв'язок:

```
#include <iostream>
#include <math.h>
using namespace std;

int main(){
    double x, f;
    cout<<"введіть значення x = "<<endl;
    cin>>x;

    if (x>=0){
        // обчислення функції f для x>=0, складений оператор
        f= sqrt(x);
        cout<<"x>0, функція дорівнює = "<<f<<endl;
    }
    else
        cout<<"x<0, немає розв'язку \n";
    }
    return 0;
}
```

У розглянутій реалізації оператор_1 і оператор_2 - складені.

5.2.5. Порядок виконання роботи

Порядок підготовки звіту та виконання ЛР2 відповідає загальним вимогам, викладеним в розділі 3.

Доведення працездатності програми буде відбуватися на основі тестування роботи програми на підготовлених і перевірених здобувачем прикладах.

Для захисту роботи необхідно підготуватися і виконати в аудиторії один із варіантів завдань з літерою **а**, вказаний викладачем.

5.2.6. Варіанти завдань

1. Обчислити функцію $y=f(x)$, для довільного значення x :

$$f(x) = \begin{cases} \sqrt{x+1} + e^x & \text{при } x \geq 0.4 \\ \operatorname{tg} x & \text{при } x < 0.4 \end{cases}$$

2. Для довільного x обчислити:

$$y = \begin{cases} 1, & \text{при } x < 0 \\ \cos x, & \text{при } 0 \leq x \leq \pi \\ -1, & \text{при } x > \pi \end{cases}$$

3. Точка задана координатами (x, y) , визначити, чи належить вона квадрату зі стороною 2:
4. Точка задана координатами (x, y) , визначити, чи належить вона одиничному колу.
5. Надано числа a, b, c, d . Якщо $a \leq b \leq c \leq d$, то кожне число замінити найбільшим з них, у протилежному випадку всі числа замінити їх квадратами.
6. Надано числа a, b, c . Подвоїти ці числа, якщо $a \geq b \geq c$, та замінити абсолютними значеннями у протилежному випадку.
7. Відомо, що з чотирьох чисел a_1, a_2, a_3, a_4 тільки три є рівними між собою. Знайти номер числа, що не дорівнює іншим, та вивести значення.
8. Для довільного a обчислити :

$$f(a) = \begin{cases} -\frac{1}{a^2}, & \text{при } a \leq -1 \\ a^2, & \text{при } -1 < a \leq 2 \\ 4, & \text{при } a > 2 \end{cases}$$

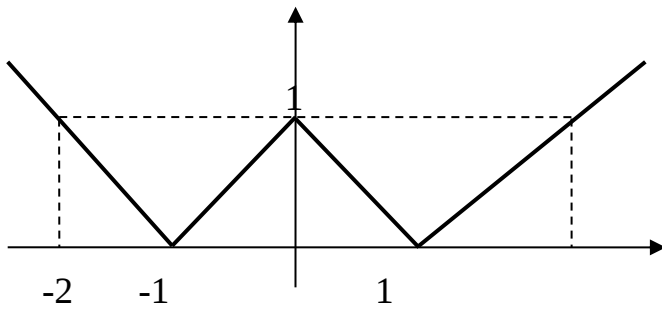
9. Для довільного a обчислити $f(a)$, де:

$$f(x) = \begin{cases} 0, & \text{при } x \leq 0 \\ x, & \text{при } 0 < x \leq 1 \\ x^4, & \text{при } x > 1 \end{cases}$$

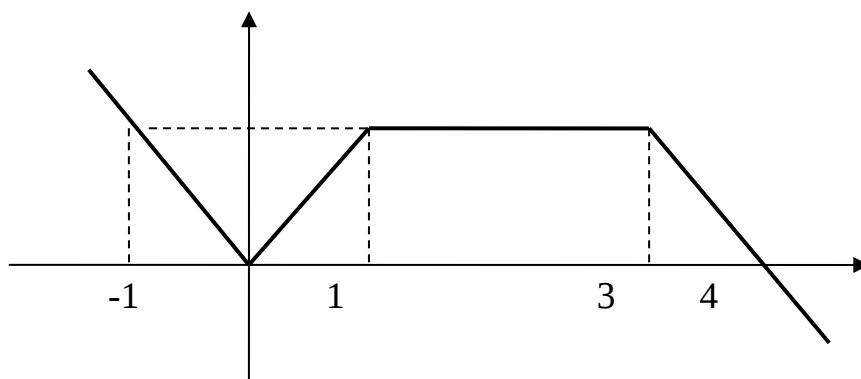
10. Надано число x . Вивести у порядку зростання значення:

$$\operatorname{ch} x; \quad 1+|x|; \quad (1+x^2)^x.$$

11. Для довільного a обчислити $f(a)$, де $f(x)$ задана графіком:



12. Для довільного a обчислити $f(a)$, де $y=f(x)$ відповідає графіку:

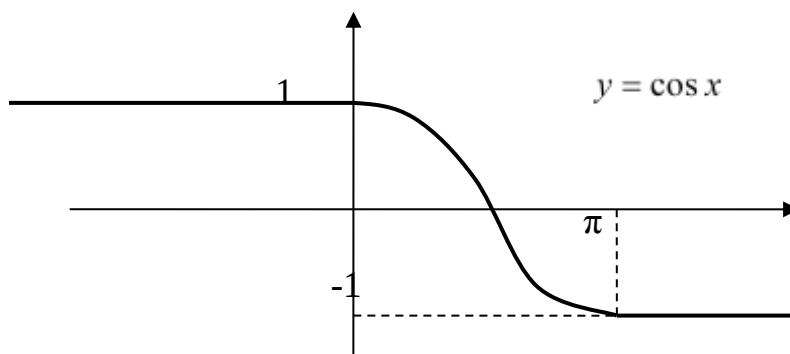


13. Без використання спеціальних функцій знайти:

$$\max \left(x + y + z, xyz, \frac{x^2 + y^2 + z^2}{4} \right), \text{ де } x, y, z - \text{в інтервалі } (-100, 100).$$

14. Змінній k надати номер квадранта, в якій знаходиться точка з координатами x та y ($xy \neq 0$).

15. Обчислити $y=f(x)$, де $f(x)$ задане графіком:



16. Обчислити:
$$y = \begin{cases} 3.85x^2/(b-c), & x > 0, b \neq c, \\ x/(2+3x), & x > 0, b = c, \\ (2b+c)x^3, & x \leq 0 \end{cases}$$

17. Обчислити:

$$y = \begin{cases} 5\sqrt{x} + 8.2, & x \geq 5, \\ 1/x, & 1 < x < 5, \\ -x^2, & -2 \leq x \leq 1, \\ x^3, & x < -2 \end{cases}$$

18. Обчислити:

$$f = \begin{cases} 4 - x^2, & 0 < x < 4, \\ 0, & x = 0, \\ x^3, & x < 0, \\ 1, & \text{в інших випадках} \end{cases}$$

19. Обчислити:

$$f = \begin{cases} (x + y)/xy + x^3, & x > 0, y > 0, \\ 3.14y, & x = 0, \\ x^2, & y = 0, \\ 1, & \text{в інших випадках} \end{cases}$$

20. Обчислити:

$$f = \begin{cases} (x + y)/(1 - xy), & x > 0, xy > 1, \\ (x^2 - y)/(1 - xy), & x = 0, \\ 1, & \text{в інших випадках} \end{cases}$$

21. Обчислити:

$$f = \begin{cases} 4 - x^2, & 0 < x < 4, \\ 0, & x = 0, \\ x^3, & x < 0, \\ \text{не існує,} & \text{в інших випадках} \end{cases}$$

22. Обчислити:

$$f = \begin{cases} (x + y)/xy + x^3, & x > 0, y > 0, \\ 3.14y, & x = 0, \\ x^2, & y = 0, \\ \text{не існує,} & \text{в інших випадках} \end{cases}$$

23. Обчислити:
$$f = \begin{cases} (x+y)/(1-xy), & x > 0, xy > 1, \\ (x^2 - y)/(1-xy), & x = 0, \\ \text{не існує,} & \text{в інших випадках} \end{cases}$$
24. Скласти програму виведення на екран усіх тризначних десяткових чисел, сума цифр яких дорівнює наданому цілому числу.
25. Якщо рівняння $ax^2 + bx + c = 0$ має дійсні корені, то символьній змінній *rez* надати значення y , а змінним x_1, x_2 - значення самих коренів, у протилежному випадку надати *rez* значення n , корені вивести у вигляді комплексних чисел.
26. Для даного числа a знайти корінь рівняння $f(x)=0$, де:
- $$f(x) = \begin{cases} 2ax + |a-1| & \text{при } a > 0 \\ \frac{e^x}{\sqrt{1+a^2}} - 1 & \text{при } a \leq 0 \end{cases}$$
27. Надані числа a_1, b_1, c_1 ; a_2, b_2, c_2 Надрукувати координати точки перетину прямих:
- $$a_1x + b_1y + c_1 = 0,$$
- $$a_2x + b_2y + c_2 = 0,$$
- якщо вона існує, та видати повідомлення, що прямі не перетинаються у протилежному випадку.
28. Надано три сторони трикутника a, b, c . Вивести повідомлення для таких можливих варіантів:
- «Трикутник побудувати неможливо»;
 - «Трикутник рівнобічний»;
 - «Трикутник рівнобедрений»;
 - «Трикутник звичайного вигляду».
29. Надано три сторони трикутника a, b, c . Вивести повідомлення для таких можливих варіантів:
- «Трикутник побудувати неможливо»;
 - «Трикутник гострокутний»;

- в) «Трикутник тупокутний»;
г) «Трикутник прямокутний».

30. Чи належить початок координат трикутнику з координатами $(x_1, y_1), (x_2, y_2), (x_3, y_3)$, де відповідні значення x_i, y_i є заданими?
31. Два прямокутники надано своїми сторонами a, b та c, d . З'ясувати, чи можна вписати один прямокутник в інший. Вивести відповідне повідомлення.
32. Надано числа a, b, c . Подвоїти ці числа, якщо $a \geq b \geq c$, та замінити абсолютними значеннями у протилежному випадку.
33. Надано числа a, b, c . Повністю дослідити біквдратне рівняння $ax^4 + bx^2 + c = 0$ на дійсні корені. Вивести у відповідності з дослідженням кількість дійсних коренів.
34. Чи можна задане натуральне число представити у вигляді суми простих чисел, які не повинні повторюватись? Скласти програму, що виводить один або декілька таких варіантів.
35. Чи можна цеглину із сторонами a, b, c втиснути в прямокутний отвір із сторонами e, d . Якщо ні, вивести відповідне повідомлення. Якщо так, вкажіть як саме.
- 36*. Для двох натуральних чисел M та N надрукувати період дробу M/N . Для кінцевого дробу період 0.
37. Перевірити, чи перетинаються трикутники, надані координатами вершин.
38. Перевірити, чи належить точка прямій. Пряма надана у канонічному вигляді.
39. Перевірити, чи лежать довільна точка та початок координат по одну сторону від прямої.
40. Знайти відстань від довільної точки до прямої, наданої у канонічному вигляді.
41. Знайти відстань від початку координат до прямої, наданої у канонічному вигляді.

42. Перевірити, чи перетинаються пряма та трикутник.
43. Задано N чисел ($N \leq 5$). Знайти мінімальне число, якщо N – парне та максимальне, якщо N – непарне.
44. Без використання оператора циклу та спеціальних функцій знайти мінімальне число в послідовності із N чисел, $N \leq 5$.
45. Без використання оператора циклу та спеціальних функцій знайти максимальне число в послідовності із N чисел, $N \leq 5$.

5.2.7. Варіанти завдань для захисту в аудиторії

Завдання наступних варіантів виконати в аудиторії з використанням оператора вибору **switch**. Протокол оформити після виконання завдання як додаток до основного протоколу.

- 1а. Розробити «калькулятор», що виконує додавання, віднімання, множення та ділення.
- 2а. Розробити програму, що переводить шістнадцяткову цифру $0 \leq n \leq 15$ в десяткову систему числення.
- 3а. Розробити програму, що переводить десяткове число $0 \leq n \leq 15$ в шістнадцяткову систему числення.
- 4а. За номером дня тижня вивести його назву.
- 5а. За номером місяця вивести пору року.
- 6а. Надане число від 1 до 100 надрукувати латинськими цифрами, пам'ятаючи, що C-100, L -50, X – 10, V – 5, I -1.
- 7а. За датою народження вивести знак гороскопу, знаючи, що:

20.01-18.02 – Водолій	21.05-21.06 – Близнюки	23.09-22.10 – Терези
19.02-20.03 – Риби	22.06-22.07 – Рак	23.10-22.11 – Скорпіон
21.03-19.04 – Овен	23.07-22.08 – Лев	23.11-21.12 – Стрілець
20.04-20.05 – Телець	23.08-22.09 – Діва	22.12-19.01 – Козеріг

 Дату зручно аналізувати у вигляді mmdd (22.10 – 1022).

- 8а. За наданим номером місяця вивести кількість днів в ньому, вважаючи рік не високосним.
- 9а. Давній японський календар використовував 60-річний цикл, який був складений із п'яти 12-річних вкладених циклів. Під цикли визначали кольори: зелений, червоний, жовтий, білий та чорний. В межах 12-річного циклу роки мали назву тварини: пацюка, корови, тигра, зайця, дракона, змії, коня, вівці, мавпи, півня, собаки та свині. 1984 рік – рік зеленого пацюка – був роком початку чергового циклу. За наданим роком визначити його назву за давнім японським календарем.
- 2–10а. За номером y ($y > 0$) деякого року визначити s — номер його сторіччя (врахувати, що, наприклад, початком XX сторіччя був 1901, а не 1900 рік).
- 2–11а. Визначити, чи є наданий рік високосним (рік високосний, якщо його номер кратний 4, однак із кратних 100 високосними є лише кратні 400; наприклад, 1700, 1800 і 1900 — не високосні роки, а 2000 — високосний).
- 2–12а. З використанням оператора **switch** скласти програму: перевести 12-бальну оцінку в п'ятибальну систему.
- 2–13а. З використанням оператора **switch** скласти програму: за віком вивести повідомлення «дитина», «школяр», «студент», «працівник», «пенсіонер».
- 2–14а. Скласти програму перевірки, якому з наданих трьох інтервалів, $[-3,5]$, $[10,12]$, $[80,100]$ належить ціла змінна.
- 2–15а. Для цілого числа k від 1 до 99 надрукувати фразу «мені k років», з огляду на при цьому, що при деяких значеннях k слово «років» треба замінити на слово «рік» чи «роки».
- 16а. Розробити програму роботи поштового автомата, що пропонує вітальні листівки: 1 – різдвяні, 2 – з днем народження, 3 – з днем захисника Вітчизни, кожна представлена у трьох варіантах (а, б, с), кожна вартістю 2 гривні. Ввести з клавіатури номер теми, варіант і купюру оплати

(номінал 1, 2, 5, 10, 20, 50, 100). Як результат вивести тему, варіант і купюри решти. Наприклад, «різдвяні, варіант с, решта 5, 2, 1»). Передбачити обробку помилок.

- 17а. Розробити програму замовлення квитків до кінотеатру, яка за вказаним сеансом і залом, рахує вартість. Врахувати, що при замовленні 5 квитків і більше – скидка 5%, 10 і більше – 10%. Червоний зал – сеанси 12-00 вартість 25, 16-00 – 45, 20-00 – 50 грн., Синій – 10-00 – 25, 14-00 – 45грн, 17 – 50 грн., Зелений – 10-00 – 25, 13-00 – 45, 21-00 – 50грн. Передбачити обробку помилок.
- 18а. Розробити програму замовлення квитків на авіарейси. З Києва – 500грн. в одному напрямку, 750 – разом із зворотнім, з Харкова – відповідно 400 і 700, зі Львова – 600 і 800. Врахувати знижку на 20%, якщо замовлення за 45 діб і 10%, якщо за 20 діб.

5.2.8. Запитання до самоконтролю

1. Які операції порівняння реалізовані в C++?
2. Які логічні операції реалізовані в C++?
3. Що ви розумієте під термінами «простий оператор», «структурований оператор» та «складений оператор»?
4. Які умовні оператори ви знаєте? Загальний вигляд. Послідовність виконання. Різновиди.
5. Що таке оператор вибору? Загальний вигляд. Послідовність виконання. Різновиди.

5.3. Лабораторна робота № 3.

5.3.1. Тема

Основні конструкції мови програмування C++. Реалізація циклів

5.3.2. Мета роботи.

Навчитися правильно реалізовувати циклічні алгоритми засобами мови C++ на прикладі розв'язання задач підрахунку суми ряду ітераційними методами з перевіркою збіжності.

5.3.3. Основні теоретичні відомості

Оператори циклу використовують для здійснення багаторазового повторення деякої послідовності дій. Кожен цикл складається з умови та тіла циклу, що виконується декілька разів. Один прохід циклу називається ітерацією. У мові C+ існують три циклічні конструкції: **while**, **do while**, **for**.

Цикл while. Називається циклом з передумовою тому, що умова в ньому перевіряється до початку циклу. В програмі ця конструкція виглядає так:

```
while (вираз-умова) оператор;
```

Тіло циклу представлено **оператором**, який може бути простим або складеним. **Складеним** називається послідовність операторів, розділених знаком «;», які сформовані у **блок**, обмежений фігурними дужками.

Цикл буде повторюватися, тобто тіло циклу буде виконуватися до тих пір, поки значення **виразу-умови** буде істинним, тобто не дорівнюватиме нулю. Як тільки значення виразу стане хибним, тобто рівним нулю, цикл припинить роботу і управління буде передане наступному за ним оператору.

Якщо значення **виразу-умови** одразу дорівнюватиме нулю, цикл з передумовою не виконається жодного разу, управління одразу буде передане до наступного за цим циклом оператора. Приклад використання:

```
int i=0, sum=0;  
while (i < n)  
    sum += ++i * i; //простий оператор
```

Оператор **while** зручно застосовувати у випадках, коли кількість ітерацій заздалегідь невідома.

Цикл з постумовою do – while. Циклічну конструкцію з постумовою застосовують, якщо тіло циклу потрібно виконати хоча б один раз. Тіло циклу представлено простим або складеним **оператором**, умова перевіряється після його виконання. Синтаксис:

do оператор while (вираз-умова);

Приклад використання:

```
int numb;
do{                               //складений оператор
    cout<<"enter the number of students \n";
    cin>>numb;
}
while (numb<0);
cout<<"ok, number of students = "<< numb << endl;
```

Як і для циклу з передумовою, для циклу з постумовою, виконання циклу буде продовжене за значенням **true**. У розглянутому прикладі – поки значення виразу `numb<0` дорівнюватиме **true**. Вихід з циклу буде здійснений по **false**.

Цикл з параметрами for. Синтаксис оголошення наступний:

for ([вираз1]; [вираз2]; [вираз3]) оператор;

де **вираз1** — вираз ініціювання, що використовується для встановлення початкового значення параметрів, не може бути блоком;

вираз2 — вираз умови продовження циклу, не містить операцію кома;

вираз3 — вираз модифікації, який визначає крок зміни параметрів, що керують циклом, після кожної ітерації;

оператор — тіло циклу, представлений простим або складеним оператором.

Вирази **вираз1**, **вираз2** та **вираз3** — необов'язкові параметри, які розділені символом «;».

Порядок роботи оператора **for** наступний:

Крок 1. Виконується тільки один раз, перед входом у цикл, обчислення **виразу1** – ініціювання.

Крок 2. Обчислення **виразу2** – умови продовження циклу.

Крок 3. Якщо умовний вираз істинний (не дорівнює нулю), виконується **оператор**, якщо умова хибна (дорівнює нулю), керування передається наступному за циклом **for** оператору.

Крок 4. Обчислення **виразу3**, модифікації.

Крок 5. Перехід до кроку 2.

Цикл **for** – різновид циклу з передумовою, тому якщо **вираз2** одразу дорівнює 0, цикл буде пропущено і управління буде передане одразу наступному оператору.

Оператор **for** може використовувати декілька параметрів – змінних, що керують циклом з використанням операції «кома», наприклад:

```
int n, y, k;  
for (k = 0, n = 20; k <= n; k++, n--)  
    y = k * n;
```

Будь-який з виразів може бути відсутнім, наприклад:

```
int n=20, y, k=0;  
for (; k <= n; k++, n--) y = k * n;
```

Цикли можуть бути **вкладеними**. Глибина вкладеності мовою не регламентована. Програміст повинен сам стежити, щоб границі зовнішнього і внутрішнього циклів не перетиналися.

5.3.4. Приклад програми з циклами

Обчислити суму ряду $1/2+1/4+1/8+1/16+ \dots$. Коректна умова виходу з циклу – обчислення з наданою точністю, тобто черговий доданок не повинен перевищувати надану точність:

```
#include <iostream>  
using namespace std;  
int main(){  
    double eps=0.0, s=0.0, a=0.5;  
    //цикл з постумовою, введення коректного значення
```

```

do{
    cout<<"eps="<<endl;
    cin>>eps;
}while(eps>0.1||eps<=0);
//цикл з передумовою, обчислення з заданою точністю
while(a>eps){
    s+=a;
    a/=2;
}
cout<<"eps = "<< eps <<"\ns = "<<s<<endl;
return 0;
}

```

У наведеному прикладі послідовно реалізовані два різновиди циклів – з постумовою і передумовою.

5.3.5. Порядок виконання роботи

Порядок підготовки звіту та виконання ЛРЗ відповідає загальним вимогам, викладеним в розділі 3.

Доведення працездатності програми буде відбуватися на основі тестування роботи програми на підготовлених і перевірених здобувачем прикладах, а також на запропонованих викладачем.

Увага! У більшості варіантів значення x , n та ε уводяться з клавіатури.

У тих варіантах, де потрібно перевірити збіжність ряду шляхом виведення кожного p -го доданку, необхідно вводити також і значення p .

Для варіантів з вкладеними сумами, виводити окремо кожену внутрішню суму, а потім уже загальну.

Для захисту роботи необхідно виконати в аудиторії один із варіантів аудиторних завдань, вказаний викладачем. Звернути увагу на форматування значень, що виводяться.

5.3.6. Варіанти завдань

1. Обчислити суму ряду $s = \sum_{i=1}^n (-1)^{i+1} x_i$, значення x_i послідовно вводити з клавіатури, масив не утворювати.
2. Обчислити добуток:

$$p = x_n (x_n + x_{n-1}) (x_n + x_{n-1} + x_{n-2}) \dots (x_n + x_{n-1} + \dots x_1),$$

значення x_i послідовно вводити з клавіатури, масив не утворювати.

3. Обчислити суму ряду:
$$s = \sum_{i=1}^n x^{2i+1} / (i+1)$$

4. Обчислити суму ряду:
$$s = \sum_{i=1,2}^n x^i / i!$$

5. Обчислити суму ряду:
$$s = \sum_{i=2,2}^n 1 / x^i$$

6. Обчислити суму ряду:
$$s = \sum_{i=1}^n x^i / i!$$

7. З наданою точністю ε , ($0 < \varepsilon < 1$), знайти:

$$e^x = 1 + x/1! + x^2/2! + \dots + x^n/n! + \dots, \text{ перевірити знайдене значення.}$$

8. З наданою точністю ε , ($0 < \varepsilon < 1$) знайти:

$$\operatorname{sh} x = x + x^3/3! + x^5/5! + \dots + x^{2n+1}/(2n+1)! + \dots,$$

перевірити знайдене значення.

9. З наданою точністю ε , ($0 < \varepsilon < 1$) знайти:

$$\cos x = 1 - x^2/2! + x^4/4! + \dots + (-1)^n x^{2n}/(2n)! + \dots,$$

перевірити знайдене значення.

10. З наданою точністю ε , ($0 < \varepsilon < 1$) знайти:

$$\ln(1+x) = x - x^2/2 + x^3/3 + \dots + (-1)^{n-1} x^n/n + \dots,$$

перевірити знайдене значення.

11. З наданою точністю ε , ($0 < \varepsilon < 1$) знайти:

$$\operatorname{arctg} x = x - x^3/3 + x^5/5 + \dots + (-1)^n x^{2n+1}/(2n+1) + \dots,$$

перевірити знайдене значення.

12. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , виводячи кожний p -й доданок та відповідне значення суми:
$$\sum_{k=0}^{\infty} \frac{(-1)^k (x/2)^{2(k-1)}}{((k+1)!)^2}.$$
13. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , виведенням кожного p -го доданку та відповідне значення суми:
$$\sum_{k=0}^{\infty} \frac{(-1)^{k+1} x^{2k+1}}{k!(2k+1)}.$$
14. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x виведенням кожного p -го доданку та відповідного значення суми:
$$\sum_{k=0}^{\infty} \frac{(-1)^k x^{4k+3}}{(k+1)!}.$$
15. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , виводом кожного p -го доданку та відповідного значення суми:
$$\sum_{k=0}^{\infty} \frac{(-1)^{k+1} (x/3)^{4k}}{2k!}.$$
16. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , виводом кожного p -го доданку та відповідного значення суми:
$$\sum_{k=0}^{\infty} \frac{(-1)^k}{((k+1)!)^2} \left(\frac{x}{2}\right)^{2(k+1)}.$$
17. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , виводом кожного p -го доданку та відповідного значення суми:
$$\sum_{k=0}^{\infty} \frac{(-1)^k x^{4k+3}}{(2k+1)!(4k+3)}.$$

18. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , в виводом кожного p -го доданку та відповідного значення суми: $\sum_{k=0}^{\infty} \frac{(-1)^{k+1} x^{2k-1}}{(2k-1)(2k+1)!}$.

19. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , виведенням кожний p -й доданок та відповідне значення суми: $\sum_{k=0}^{\infty} \frac{(-1)^k}{k!(k+1)!} \left(\frac{x}{2}\right)^{2k+1}$

20. Дано дійсне число x . Послідовність $a_1, a_2, \dots$ утворена за законом: $a_n = \frac{(-1)^n x^{2n}}{n(n+1)(n+2)}$. Отримати $\sum_{n=1}^k a_n$, де k -найменше число, що задовольняє умовам: $k > 10; |a_{k+1}| < 10^{-5}$.

21. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , виводячи кожний p -й доданок та відповідне значення суми: $\sum_{k=0}^{\infty} \frac{x^{2k}}{2^k k!}$.

22. Дано дійсне число x . Послідовність $a_1, a_2, \dots$ утворена за таким законом: $a_n = \frac{x^n}{(2n)!}$. Отримати $S = \sum_{n=1}^k a_n$, де k -найменше число, що задовольняє умовам: $k > 10; |a_{k+1}| < 10^{-5}$

23. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , виводом кожного p -го доданку та відповідне значення суми: $\sum_{k=0}^{\infty} \frac{(-1)^{k+1}}{(2k)!} \left(\frac{x}{3}\right)^{4k}$.

24. Обчислити з точністю до ε ($0 < \varepsilon < 1$) кожен внутрішню суму, визначити кількість доданків, вивести проміжні результати:

$$\sum_{x=1}^5 \sum_{k=1}^{\infty} x! / k(x)^k.$$

25. Обчислити з точністю до ε ($0 < \varepsilon < 1$) кожну внутрішню суму, визначити кількість доданків, вивести проміжні результати:

$$\sum_{x=1}^5 \sum_{k=1}^{\infty} k^2 x^k / (k+1)!$$

26. Обчислити з точністю до ε ($0 < \varepsilon < 1$) кожну внутрішню суму, визначити кількість доданків, вивести проміжні результати:

$$\sum_{x=1}^5 \sum_{k=1}^{\infty} (-1)^k x^k / k!$$

27. Обчислити з точністю до ε ($0 < \varepsilon < 1$) кожну внутрішню суму, визначити кількість доданків, вивести проміжні результати:

$$\sum_{x=1}^5 \sum_{k=1}^{\infty} (-1)^k x^k / (k+1)!$$

28. Обчислити з точністю до ε ($0 < \varepsilon < 1$) кожну внутрішню суму, визначити кількість доданків, вивести проміжні результати:

$$\sum_{x=1}^5 \sum_{k=1}^{\infty} k * x^k / (k+1)!$$

29. Обчислити $\sum_{i=1}^{100} \sum_{j=1}^{50} \frac{1}{i+j^2}$, вивести окремо внутрішні суми.

30. Обчислити $\sum_{i=1}^{100} \sum_{j=1}^{50} \sin(i^3 + j^4)$, вивести окремо внутрішні суми.

31. Обчислити $\sum_{i=1}^{100} \sum_{j=i}^{100} \frac{j-i+1}{i+j}$, вивести окремо внутрішні суми.

32. Обчислити $\sum_{i=1}^{100} \sum_{j=1}^i \frac{1}{i+2j}$, вивести окремо внутрішні суми.

33. Для наданих x, k обчислити $\sum_{k=1}^n \sum_{m=k}^n \frac{x+k}{m}$, вивести окремо внутрішні суми.

34. Знайти 100 перших простих чисел.

35. Знайти натуральне число від 1 до 10000 з максимальною сумою дільників.
36. Знайти всі досконалі числа, менші, ніж надане натуральне число. Число є досконалим, коли дорівнює сумі своїх дільників, крім самого себе.
37. Знайти всі прості дільники наданого числа.
38. Надані два натуральних числа, m, n . Знайти всі числа, менші за n , сума цифр яких дорівнює m .
39. З клавіатури вводиться значення кута в радіанах x . Обчислити y_i – перше із чисел послідовності $\sin x, \sin \sin x, \sin \sin \sin x, \dots$ таке, що $|y_i| \leq 10^{-4}$
40. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , виводячи кожний p -й доданок та відповідне значення суми:
$$\sum_{k=1}^{\infty} (-1)^k \frac{k \sin kx}{k}.$$
41. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , виводячи кожний p -й доданок та відповідне значення суми:
$$\sum_{k=1}^{\infty} \frac{x^k \cos(k \frac{\pi}{3})}{k}.$$
42. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , виводячи кожний p -й доданок та відповідне значення суми:
$$\sum_{k=1,2}^{\infty} \frac{\cos kx}{k^2}.$$
43. Дано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε та дослідити збігання послідовності для обраного значення x , виводячи кожний p -й доданок та відповідне значення суми:
$$\sum_{k=0}^{\infty} \frac{\cos(k \frac{\pi}{4})}{k!} x^k.$$

Завдання для аудиторної роботи

1. На інтервалі $[0.1, 2.1]$ з кроком 0.2 вивести таблицю значень для побудови графіка функції

$$y = x^2 + \sin 5x$$

2. На інтервалі $[0.1, 2.2]$ з кроком 0.2 вивести таблицю значень для побудови графіка функції

$$y = x^2 - \cos^2(x+1)$$

3. На інтервалі $[1, 3]$ з кроком 0.2 вивести таблицю значень для побудови графіка функції

$$y = x^3 - 4x^2 + 2$$

4. На інтервалі $[0.1, 2.7]$ з кроком 0.2 вивести таблицю значень для побудови графіка функції

$$y = |x^2 - 4| + 0.25x - 2$$

5. На інтервалі $[1, 29]$ з кроком 0.2 вивести таблицю значень для побудови графіка функції

$$y = \ln x^2 - x + 4$$

6. На заданому інтервалі $[-1.5, 1.5]$ з кроком 0.2 вивести таблицю значень для побудови графіка функції

$$y = \begin{cases} 1 + \sqrt{|\cos x|}, & x > 1 \\ x + 1, & -0.5 \leq x \leq 1 \\ 1 - x^2, & x < -0.5 \end{cases}$$

7. На інтервалі $[-2, 3]$ з кроком 0.2 вивести таблицю значень для побудови графіка функції

$$y = \begin{cases} 2.5x^3 + 6x^2 - 30, & x > 1.5 \\ x + 1, & 0 \leq x \leq 1.5 \\ x, & x < 0 \end{cases}$$

8. На інтервалі $[-\pi/2, 2\pi]$ з кроком 0.2 вивести таблицю значень для побудови графіка функції

$$y = \begin{cases} 3x^2, & x > 4.5 \\ e^{-x}, & 1 \leq x \leq 4.5 \\ -\cos^2(2x), & x < 1 \end{cases}$$

9. Вивести у три колонки на екран ASCII-коди усіх маленьких латинських букв
10. Побудувати на екрані табличку множення 10 на 10
11. Вивести на екран двійкове зображення введеного з клавіатури беззнакового цілого числа <256
12. Вивести на екран табличку з ASCII-кодами символів десяткових цифр, їх зображеннями у вісімковій і шіснадцятковій системах числення
13. На інтервалі [-1,5] з кроком 0.5 вивести таблицю значень для побудови графіка функції

$$f = \begin{cases} 4 - x^2, & 0 < x < 4, \\ 0, & x = 0, \\ x^3, & x < 0, \\ 1, & \text{в інших випадках} \end{cases}$$

14. На інтервалі [-3,6] з кроком 0.5 вивести таблицю значень для побудови графіка функції

$$y = \begin{cases} 5\sqrt{x} + 8.2, & x \geq 5, \\ 1/x, & 1 < x < 5, \\ -x^2, & -2 \leq x \leq 1, \\ x^3, & x < -2 \end{cases}$$

15. На інтервалі [-2,3] з кроком 0.2 вивести таблицю значень для побудови графіка функції

$$f(a) = \begin{cases} -\frac{1}{a^2}, & \text{при } a \leq -1 \\ a^2, & \text{при } -1 < a \leq 2 \\ 4, & \text{при } a > 2 \end{cases}$$

16. На інтервалі [-2,3] з кроком 0.2 вивести таблицю значень для побудови графіка функції

$$f(x) = \begin{cases} \sin x, & \text{при } x > 0 \\ x^2, & \text{при } x \leq 0 \end{cases}$$

5.3.7. Запитання до самоконтролю

1. Які різновиди циклічних конструкцій реалізовані в мові C++?
2. Які варіації циклу **for** Ви знаєте?
3. Що таке цикл з передумовою, який його загальний вигляд та порядок виконання?
4. Що таке цикл з постумовою? Загальний вигляд та порядок виконання?
5. Чим відрізняється цикл з передумовою від циклу з постумовою? Який із них виконується хоча б один раз у будь-якому випадку?
6. Чи може бути ситуація, коли цикл не буде виконаний жодного разу? Чи приведе це до переривання виконання програми?

5.4. Лабораторна робота № 4.

5.4.1. Тема

Структуровані типи даних. Масиви

5.4.2. Мета роботи.

Навчитися правильно реалізовувати алгоритми роботи з одновимірними та двовимірними масивами з використанням індексів.

5.4.3. Основні теоретичні відомості

Масив – іменована послідовність областей пам'яті, що містять однотипні елементи. Кожна така область пам'яті називається **елементом** масиву. Для масивів визначений **вимір**, який дозволяє описувати одновимірні, двовимірні, багатовимірні масиви та **розмірність** (більша або дорівнює одиниці), яка задає кількість елементів у одному вимірі масиву. Загальна кількість елементів масиву називається **розміром** масиву і дорівнює **вимір*розмірність**.

Одновимірний масив або **вектор** має єдиний вимір. Синтаксис оголошення одновимірного масиву наступний:

тип ідентифікатор[розмір] ;

тип – базовий тип, тобто тип елемента масиву;

ідентифікатор - визначає назву масиву

розмір – константний вираз, обов'язкова частина оголошення, визначає кількість елементів масиву. Приклад оголошення:

```
double balance[5];
```

Елементи масиву в пам'яті комп'ютера розташовані послідовно без пропусків. До кожного з елементів можна звернутися за його порядковим номером – **індексом**, який записують у прямокутних дужках безпосередньо за ідентифікатором масиву. Індекс може бути тільки цілого типу.

Нумерацію елементів масиву починають з нуля.

	0	1	2	3	4	індекс
balance	36.6	15.8	-33.6	44.1	-20.3	вміст

Приклади звертання до елементів масиву:

```
double b=balance[3];  
balance[0]=balance[j+1]+0.5;
```

У результаті такої послідовності дій вміст третього елемента масиву буде надано змінній `b`, а вміст нульового елемента зміниться і дорівнюватиме вмісту `j+1`-го елемента цього ж масиву, збільшеному на 0.5.

Двовимірний масив має два виміри, його можна представити у вигляді матриці (або таблиці), що має `розмір1` рядків та `розмір2` стовпчиків. Тоді синтаксис оголошення двовимірного масиву наступний:

```
тип ідентифікатор[розмір1][розмір2];
```

Доступ до окремого елемента масиву можна здійснити визначенням двох його координат: номеру рядка `i=0..розмір1-1` та стовпця `j=0..розмір2-1`. У наведеному прикладі оголошений масив `a` із трьох рядків і семи стовпчиків:

```
int a[3][7];  
i = 2; j = 4;  
a[i][j]=12;    // звертання до a[2][4];
```

Аналогічно двовимірному можна оголосити масив з більшою вимірністю, додавши розмірності по вимірах у прямокутних дужках. Для доступу до елемента такого масиву необхідно вказати відповідну кількість індексів. Наприклад, у **тривимірному масиві** необхідно вказати три індекси: `a[i][j][k]`. Потрібно пам'ятати, що робота з масивами, вимірність яких перевищує 3, істотно ускладнює алгоритм, тому, по можливості, необхідно уникати організації подібних структур даних.

5.4.4. Приклад програми роботи з масивом

Задача: Знайти найбільший елемент серед найменших кожного рядка для двовимірного масиву `arr`, максимальний розмір якого `10*10`. Відомо, що для даної матриці такий елемент точно існує і він єдиний.

```
#include <iostream>  
using namespace std;  
#define MAX 10
```

```

// пошук сідлової точки масиву (saddle point)
int main(){
    double n, m, arr[MAX][MAX];
    int imin, imax, jmin, jmax;
    cout<<"Enter dimention of the array as m rows and n
columns:\n";
    do{
        cout<<"Enter m: 0<m<10;"<<endl;
        cin>>m;
    } while(m<1||m>9);
    do{
        cout<<"Enter n: 0<n<10;"<<endl;
        cin>>n;
    } while(n<1||n>9);
    cout<<"Enter the values of the array elements row by
row\n";
    for(int i = 0; i < m; i++)
        for(int j = 0; j < n; j++)
            cin>>arr[i][j];
    cout<<"You have entered such an array:\n";
    for(int i = 0; i < m; i++){
        for(int j = 0; j < n; j++)
            cout<<arr[i][j]<<' ';
        cout<<endl;
    }
    // пошук мінімального елемента рядка 0
    imin=jmin=0;
    double min=arr[imin][jmin];
    for(int j=1;j<n;j++)
        if(arr[0][j]<min) {
            min=arr[0][j];
            jmin=j;
        }
    cout<<"current min in 0 row ="<<min<<endl;
    // прийемо найменшу точку 0-го рядка за шуканий максимум
    double max=min; imax=imin;jmax=jmin;
    for(int i = 1; i < m; i++) {
        // пошук min у i-му рядку
        int min=arr[i][0],imin=i,jmin=0;
        for(int j=1;j<n;j++)
            if(arr[i][j]<min) {
                min=arr[i][j];

```

```

        jmin=j;
    }
    cout<<"current min in "<<i<<" row = "<<min<<endl;
    if (min>max) {
        max=min; imax=imin; jmax=jmin;
    }
}
cout<<"the saddle point is arr["<<imax<<"]["<<jmax<<"]="
"<<max<<endl;
return 0;
}

```

Результати роботи програми:

```

Enter dimention of the array as m rows and n columns:
Enter m: 0<m<10;
4
Enter n: 0<n<10;
4
Enter the values ??of the array elements row by row
1.5 6.7 8.9 4.0
7.5 8.2 5.5 6.2
4.4 5.5 6.6 3.3
5.1 4.1 3.1 4.2
You have entered such an array:
1.5 6.7 8.9 4
7.5 8.2 5.5 6.2
4.4 5.5 6.6 3.3
5.1 4.1 3.1 4.2
current min in 0 row =1.5
current min in 1 row = 5
current min in 2 row = 3
current min in 3 row = 3
the saddle point is arr[1][2] = 5

```

У наведеному прикладі показано використання можливості виведення проміжних результатів для полегшення процесу відлагодження програми та таку, що дозволяє зрозуміти коректність отриманого результату. Після остаточного відлагодження програми, виведення проміжних результатів можна прибрати.

5.4.5. Порядок виконання роботи

Порядок підготовки звіту та виконання ЛР4 відповідає загальним вимогам, викладеним в розділі 3.

Для перевірки коректної роботи програми передбачити можливість вибору при початковому заповненні масиву користувачем: або випадковими числами, або з клавіатури.

Звернути увагу, що у деяких варіантах масив має бути заповнений у певній послідовності числами натурального ряду.

Для захисту роботи необхідно виконати в аудиторії один із варіантів завдань з літерою **а**, вказаний викладачем.

5.4.6. Варіанти завдань

Одновимірні масиви

- 1а. Надано масив $A(N)$. Якщо N – парне, знайти максимальний елемент, інакше – мінімальний.
- 2а. Надано масив $A(N)$. Поміняти мінімальний та максимальний елементи місцями.
- 3а. Надано масив $A(N)$. Вилучити елемент з наданим індексом (решту зсунути).
- 4а. Надано масив $A(N)$. Вставити елемент у надане місце.
- 5а. Надано масив $A(N)$. Вилучити мінімальний елемент (решту зсунути)..
- 6а. В одновимірному масиві знайти найдовшу послідовність нулів, якщо така існує.
- 7а. Надано одновимірний масив. Всі його елементи, що не дорівнюють нулю, переписати у початок масиву, не порушуючи порядку. Нульові – в кінець масиву. Новий масив не створювати.
- 8а. Надано два вже впорядкованих масиви
$$a_1 \leq a_2 \leq \dots \leq a_n, b_1 \leq b_2 \leq \dots \leq b_m$$
 Створити новий впорядкований масив $c_1 \leq c_2 \leq \dots \leq c_{n+m}$. Сортування жодного масиву не виконувати.
- 9а. Дано два одновимірних масиви, $A(n)$, $B(m)$, $n > m$. Перевірити, чи існує послідовність елементів масиву A , яка цілком співпадає з масивом B .

- 10a. Дано два одновимірних масиви, $A(n)$, $B(m)$, $n > m$. Перевірити, чи існує послідовність елементів масиву A , яка цілком співпадає з масивом B , і, якщо так, вилучити її.
- 11a. Дано два одновимірних масиви, $A(n)$, $B(m)$, $n > m$. Перевірити, чи є в масиві A елементи, що співпадають з елементами масиву B , і, якщо так, вилучити їх.
- 12a. Задано масив $X(M)$. Знайти довжину k самої довгої послідовності, що задовольняє умові: $x_p < x_{p+1} > x_{p+2} < x_{p+3} > \dots < x_{p+k-1} > x_{p+k}$
Вивести на друк цю послідовність.
- 13a. Задано масив $X(M)$. Визначити, чи впорядковані елементи масиву і, якщо так, то за спаданням чи за зростанням.
- 14a. Дано два одновимірних масиви, $A(n)$, $B(n)$. Якщо ці масиви не співпадають за значеннями елементів, порахувати їх скалярний добуток.
- 15a. Задано масив $X(M)$. Відомо, що елементи масиву відсортовані за модулем за зростанням (це перевіряти не потрібно). Переставити елементи масиву так, щоби на початку масиву знаходилися усі додатні, а потім – усі від’ємні елементи так, щоби початковий порядок слідування окремо додатних, окремо від’ємних елементів зберігся.
- 16a. Дано два дійсних одновимірних масиви, $A(n)$, $B(n)$. Усі елементи масиву A , абсолютне значення яких менше деякого наперед заданого значення $0 < \epsilon < 1.0$, замінити відповідними значеннями масиву B (тобто $A[i]$ замінити $B[i]$), або нулем.
- 17a. Відомо, що усі елементи символьного масиву із 10 елементів можуть приймати одне з трьох фіксованих значень: або ‘а’, або ‘b’, або ‘с’. Відсортуйте масив за не спаданням.
- 18a. Задано символьний масив $X(M)$. Вилучити з нього усі елементи, у що містять символи цифр 0..9.
- 19a. Задано дійсний масив $X(M)$. Вилучити з нього усі елементи, що містять дробову частину, яка починається десятковою п’ятіркою.

Двовимірні масиви

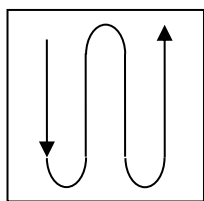
1. Дана дійсна квадратна матриця A розміром $n \times n$. Утворити вектор B , елементами якого є мінімальні елементи рядків матриці A .
2. Дана дійсна квадратна матриця A розміром $n \times n$. Утворити вектор B , елементами якого є суми елементів по стовпчиках матриці A .
3. Надано цілі числа $a_1, \dots, a_{14}$, та квадратна матриця цілих чисел розміром $n \times n$. Замінити нулями у матриці ті елементи із парною сумою індексів, значення яких співпадають зі значеннями $a_1, \dots, a_n$.
4. Дано дві дійсні квадратні матриці розміром $n \times n$. Утворити нову матрицю множенням елементів кожного рядка першої матриці на найбільше із значень елементів відповідного рядка другої матриці.
5. Дана дійсна квадратна матриця розміром $n \times n$. Шляхом перестановки рядків та стовпчиків помістити один з елементів матриці, що має найменше значення, у правому нижньому куті.
6. Дана дійсна квадратна матриця розміром $n \times n$. Шляхом переставлення рядків та стовпчиків помістити один з елементів матриці, що має найбільше за модулем значення, у лівий верхній кут.
7. Дана дійсна квадратна матриця розміром $n \times n$, усі елементи якої різні. Знайти максимальні елементи на головній та на побічній діагоналі та поміняти їх місцями.
8. Дано двовимірний масив $A(M \times N)$ цілих чисел. Відомо, що серед його елементів тільки два рівних між собою. Знайти їх та вивести на друк їх індекси.
9. У наданій дійсній квадратній матриці розміром $n \times n$ знайти найбільший по модулю елемент. Знайти матрицю розміром $n-1 \times n-1$ шляхом викреслювання рядка та стовпчика, на перетину яких знаходиться знайдений елемент.
10. Дано дві дійсні квадратні матриці розміром $n \times n$. Знайти нову матрицю, додаючи до елементів кожного рядка першої матриці добуток елементів відповідних рядків другої матриці.

11. Дано двовимірний масив $A(M \times N)$. Деякий його елемент називається «сідловою точкою», якщо він одночасно найменший в рядку та найбільший в стовпці. Знайти такий елемент, якщо він існує, а якщо ні - вивести на друк відповідне повідомлення.
12. Дана дійсна квадратна матриця розміром $2n \times 2n$. Отримати нову матрицю перестановкою її блоків розміром $n \times n$ за годинниковою стрілкою.
13. З дійсної квадратної матриці розміром $2n \times 2n$ отримати нову матрицю розміром $n \times n$ перестановкою її блоків розміром $n \times n$ по діагоналям.
14. Дана дійсна квадратна матриця розміром $n \times n$. Знайти найбільший з елементів, розташованих під побічною діагоналлю.
15. Дана дійсна квадратна матриця розміром $n \times n$. Знайти найменший з елементів, розташованих над головною діагоналлю.
16. Дана дійсна квадратна матриця розміром $n \times n$. Знайти суму елементів, розташованих під головною діагоналлю.
17. Дана дійсна квадратна матриця розміром $n \times n$. Знайти суму від'ємних елементів, розташованих над побічною діагоналлю.
18. Дана дійсна квадратна матриця A розміром $n \times n$. Знайти дійсну квадратну матрицю B розміром $n \times n$, кожний елемент якої дорівнює сумі всіх елементів матриці A за виключенням елементів, розташованих у рядку i і стовпчику, на перетині яких знаходиться цей елемент. Тобто для елемента b_{ij} виключені будуть елементи i -того рядка і j -того стовпчика.
19. Дана дійсна квадратна матриця A розміром $n \times n$. Знайти дійсну квадратну матрицю B розміром $n \times n$, кожний елемент b_{ij} якої дорівнює сумі елементів першої матриці у верхній правій прямокутній області, обмеженій i -м рядком і j -м стовпчиком: (включно).
20. Дана дійсна квадратна матриця A розміром $n \times n$. Знайти дійсну квадратну матрицю B розміром $n \times n$, кожний елемент b_{ij} якої

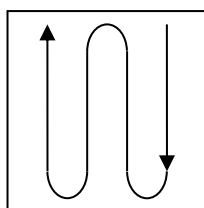
дорівнює сумі елементів першої матриці у верхній лівій прямокутній області, обмеженій i -м рядком і j -м стовпчиком (включно).

21. Дана дійсна квадратна матриця A розміром $n \times n$. Знайти дійсну квадратну матрицю B розміром $n \times n$, кожний елемент b_{ij} якої дорівнює сумі елементів першої матриці у нижній лівій прямокутній області, обмеженій i -м рядком і j -м стовпчиком (включно).
22. Дана дійсна квадратна матриця A розміром $n \times n$. Знайти дійсну квадратну матрицю B розміром $n \times n$, кожний елемент b_{ij} якої дорівнює сумі елементів першої матриці у нижній правій прямокутній області, обмеженій i -м рядком і j -м стовпчиком (включно).
23. Дана дійсна квадратна матриця A розміром $n \times n$. Знайти дійсну квадратну матрицю B розміром $n \times n$, кожний елемент b_{ij} якої є сумою елементів матриці A у квадратній області, обмеженій i -тим і j -тим стовпчиками та i -тим і j -тим рядками для кожного.
24. Утворити квадратну матрицю, елементами якої є натуральні числа, що розташовані по спіралі за годинниковою стрілкою, починаючи з правого верхнього кута.
25. Утворити квадратну матрицю, елементами якої є натуральні числа, що розташовані по спіралі проти годинникової стрілки, починаючи з лівого нижнього кута.
26. Дана дійсна квадратна матриця порядку $n \times n$. Знайти максимальний елемент, розташований в області, що утворилася перетином головної і побічної діагоналі і складається з двох рівнобічних трикутників справа і зліва від точки перетину.
27. Дана дійсна квадратна матриця $n \times n$. Знайти мінімальний елемент, розташований в області, що утворилася перетином головної і побічної діагоналі і складається з двох рівнобічних трикутників зверху і знизу від точки перетину.

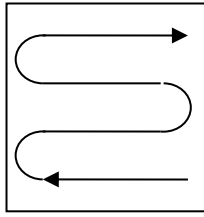
28. Дана дійсна квадратна матриця порядку $n \times n$. Знайти найбільший з елементів, розташованих в області, що утворилася перетином головної і побічної діагоналі і складається з одного рівнобічного трикутника справа від точки перетину.
29. Дана дійсна квадратна матриця порядку $n \times n$. Знайти найбільший з елементів, розташованих в області, що утворилася перетином головної і побічної діагоналі і складається з одного рівнобічного трикутника знизу від точки перетину.
30. Дана дійсна квадратна матриця порядку $n \times n$. Знайти найбільший з елементів, розташованих в області, що утворилася перетином головної і побічної діагоналі і складається з одного рівнобічного трикутника зверху від точки перетину.
31. Дана дійсна квадратна матриця порядку $n \times n$. Знайти найбільший з елементів, розташованих в області, що утворилася перетином головної і побічної діагоналі і складається з одного рівнобічного трикутника зліва від точки перетину.
32. Утворити матрицю $n \times n$, елементами якої є натуральні числа, значення яких збільшується згори вниз у першому стовпчику, знизу вгору у другому розташовані наступним чином:



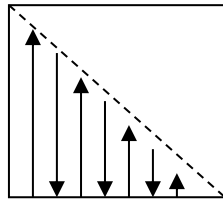
33. Утворити квадратну матрицю розміром $n \times n$, елементами якої є натуральні числа, розташовані наступним чином:



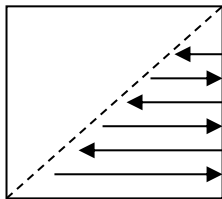
34. Утворити квадратну матрицю розміром $n \times n$, елементами якої є натуральні числа, розташовані наступним чином:



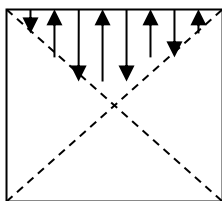
35. Утворити квадратну матрицю розміром $n \times n$, розташувавши числа натурального ряду за напрямом стрілки. На інших позиціях елементи дорівнюють нулю. Починати з лівого нижнього кута:



36. Утворити квадратну матрицю $n \times n$, розташувавши числа натурального ряду за напрямом стрілки. На інших позиціях елементи дорівнюють нулю. Починати з лівого нижнього кута:

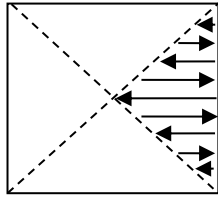


37. Утворити квадратну матрицю $n \times n$, розташувавши числа натурального ряду за напрямом стрілки. На інших позиціях елементи дорівнюють нулю. Починати з лівого верхнього кута:

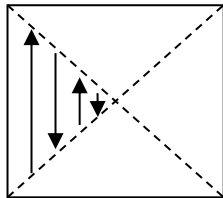


38. Утворити квадратну матрицю $n \times n$, розташувавши числа натурального ряду за напрямом стрілки. На інших позиціях елементи дорівнюють нулю.

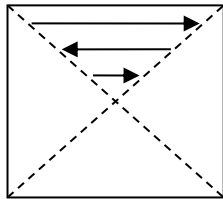
нулю. Починати з правого нижнього кута:



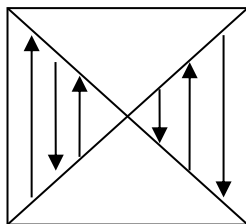
39. Утворити квадратну матрицю $n \times n$, розташувавши числа натурального ряду за напрямом стрілки. На інших позиціях – нулі:



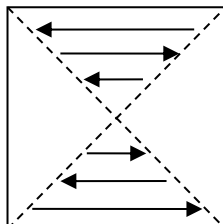
40. Утворити квадратну матрицю $n \times n$, розташувавши числа натурального ряду за напрямом стрілки. На інших позиціях – нулі:



41. Утворити квадратну матрицю $n \times n$, розташувавши числа натурального ряду за напрямом стрілки. На інших позиціях елементи дорівнюють нулю. Починати з лівого нижнього кута:



42. Утворити квадратну матрицю $n \times n$, розташувавши числа натурального ряду за напрямом стрілки. На інших позиціях елементи дорівнюють нулю. Починати з правого верхнього кута:



43. Утворити квадратну матрицю $n \times n$, розташувавши числа натурального ряду за напрямом стрілки. На інших позиціях елементи дорівнюють нулю:

7. При проходженні матриці по рядках/стовбцях який цикл буде зовнішнім, а який вкладеним ?
8. Як організувати проходження матриці «змійкою»? (ідея)
9. Яка умова дозволяє обрати елементи на головній діагоналі квадратної матриці? побічної діагоналі? нижче головної? вище побічної?
10. В чому різниця між двома наведеними алгоритмами проходження матриці по спіралі – симетричного та несиметричного?

5.5. Лабораторна робота № 5.

5.5.1. Тема

Структуровані типи даних. Реалізація алгоритмів сортування

5.5.2. Мета роботи.

Навчитися правильно реалізовувати базові алгоритми сортування одновимірних масивів з використанням індексів.

5.5.3. Основні теоретичні відомості

Загальна характеристика методів сортування. Методи сортування "на тому ж місці" можна розбити у відповідності до принципів, що їх визначають, на три основні групи:

1. Сортування за допомогою вставки (by Insertion) або за допомогою включення.
2. Сортування за допомогою вибору (by Selection) або за допомогою виділення.
3. Сортування за допомогою обміну (by Exchange) або бульбашкове. Кожна група має прямий метод (самий простий) і покращені (ускладнені) методи сортування.

Принцип сортування за допомогою вставки. Масив розподіляється на відсортовану та не відсортовану частини. На першому кроці за відсортовану частину (послідовність) приймається перший елемент масиву. Кожний наступний елемент з невідсортованої частини вставляємо в заздалегідь відсортовану послідовність так, щоб ця послідовність залишалась відсортованою.

При цьому треба:

1. Знайти місце, куди потрібно вставити цей елемент.
2. Зсунути елементи, що стоять справа у відсортованій частині, на одну позицію вправо.
3. На звільнене місце поставити елемент, який аналізують (вставляють).

Можливі два варіанти, як це можна зробити.

Перший. Кожний наступний елемент $a[i]$ із невідсортованої частини вилучити і запам'ятати у робочій змінній. Потім порівнювати його послідовно з елементами відсортованої частини і знайти місце вставки. Усі наступні відсортовані елементи зсунути на одну позицію вправо до звільненого місця, після чого вставити елемент із робочої змінної на знайдене місце вставки.

Другий. Кожний наступний елемент $a[i]$ із невідсортованої частини так само послідовно, але справа наліво, порівнювати з кожним із елементів у відсортованій частині. Після кожного порівняння, якщо потрібно, елемент у відсортованій частині одразу зсунути на одну позицію вправо. Як тільки знайдене потрібне місце вставки, елемент $a[i]$ вставити на звільнену позицію.

Принцип сортування за допомогою прямого вибору. Масив також поділити на відсортовану та не відсортовану частини. На першому кроці весь масив вважати не відсортованим. У не відсортованій частині знайти мінімальний (або максимальний) елемент та поміняти місцями з першим елементом не відсортованої частини. Тепер цей елемент вважати першим елементом відсортованої частини. Таку саму процедуру повторити з новою невідсортованою частиною $n-1$ раз. Після кожного разу межа, що розділяє відсортовану і не відсортовану частини, буде зсунута на одну позицію.

Принцип сортування за допомогою прямого обміну (бульбашкове). Зліва направо по чергові йде порівняння двох сусідніх елементів. Якщо вони не впорядковані між собою, то їх міняють місцями. В базовому алгоритмі проходження масиву та чергове впорядкування повторюють $n-1$ раз.

5.5.4. Порядок виконання роботи

Порядок підготовки звіту та виконання ЛР5 відповідає загальним вимогам, викладеним в розділі 3.

Для перевірки коректної роботи програми передбачити можливість вибору при початковому заповненні масиву користувачем: або випадковими числами, або з клавіатури.

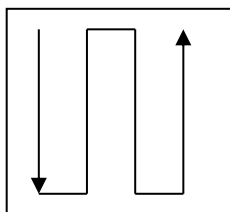
Для захисту роботи необхідно підготуватися і виконати в аудиторії один із варіантів завдань, вказаний викладачем.

5.5.5. Варіанти завдань

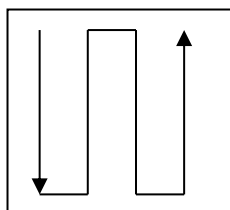
1. Задано матрицю $m \times n$. Відсортувати k -й рядок за зростанням методом вставки.
2. Задано матрицю $m \times n$. Відсортувати k -й рядок за спаданням методом вставки.
3. Задано матрицю $m \times n$. Відсортувати i -й стовпчик за зростанням методом обміну з прапорцем.
4. Задано матрицю $m \times n$. Відсортувати i -й стовпчик за зростанням методом вибору.
5. Задано матрицю $m \times n$. Відсортувати i -й стовпчик за зростанням методом обміну з запам'ятовуванням місця останньої перестановки.
6. Задано матрицю $m \times n$. Відсортувати i -й стовпчик за спаданням шейкерним методом.
7. Відсортувати головну діагональ матриці методом вставки.
8. Відсортувати головну діагональ матриці методом обміну.
9. Відсортувати головну діагональ матриці шейкерним методом.
10. Впорядкувати за зростанням побічну діагональ матриці шейкерним сортуванням.
11. Впорядкувати за спаданням побічну діагональ матриці методом обміну з прапорцем.
12. Задано матрицю, відсортувати кожний із стовпчиків за спаданням методом обміну.
13. Задано матрицю, відсортувати кожний із стовпчиків за зростанням методом вставки.
14. Задано матрицю, відсортувати кожний із рядків за спаданням методом обміну з прапорцем.

15. Задано матрицю, відсортувати кожний із рядків за зростанням методом вибору.
16. Задано матрицю, відсортувати її рядки (за сумою елементів, з використанням масиву сум) за зростанням методом вставки.
17. Задано матрицю, відсортувати її рядки (за сумою елементів, з використанням масиву сум) за зростанням методом обміну.
18. Задано матрицю, відсортувати стовпчики (за сумою елементів, з використанням масиву сум) за спаданням методом вставки.
19. Задано матрицю, відсортувати стовпчики (за сумою елементів, з використанням масиву сум) за спаданням методом вибору.
20. Переставити рядки матриці за спаданням сум елементів рядків методом вибору. Можна використовувати масив сум.
21. Переставити стовпчики матриці за спаданням сум елементів стовпчиків методом обміну з прапорцем. Можна використовувати масив сум.
22. Задано матрицю, відсортувати кожний із стовпчиків почергово за зростанням та за спаданням методом обміну (парні – за зростанням, не парні – за спаданням).
23. Задано матрицю, відсортувати кожний із рядків почергово за зростанням та за спаданням методом вибору. (парні – за зростанням, не парні – за спаданням).
24. Впорядкувати за спаданням методом вибору всі елементи матриці $m \times n$, по рядках так, ніби кожний наступний рядок є продовженням попереднього.
25. Впорядкувати за спаданням методом вибору всі елементи матриці $m \times n$, по стовпчиках так, ніби кожний наступний стовпчик є продовженням попереднього.
26. Відсортувати масив за зростанням методом обміну, виконуючи сортування елементів, які перевищують надане число.

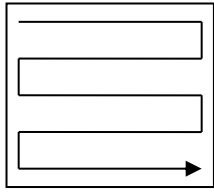
27. Задана послідовність чисел довжиною n . Впорядкувати елементи, що перевищують задане значення p , за спаданням методом вибору. Від'ємні залишаються на своїх місцях.
28. Відсортувати масив за зростанням методом обміну з прапорцем, виконуючи сортування додатних елементів, нульові та від'ємні залишити на своїх місцях.
29. Відсортувати за зростанням елементи одновимірного масиву методом обміну із запам'ятовуванням місця останньої перестановки, впорядковуючи тільки додатні елементи, залишивши на своїх місцях від'ємні елементи та нульові.
30. Впорядкувати за спаданням методом вставки ті елементи кожного рядка матриці, що розташовані між мінімальним та максимальним елементами.
31. Впорядкувати за зростанням методом вибору ті елементи кожного рядка матриці, що розташовані між мінімальним та максимальним елементами.
32. Впорядкувати за спаданням методом обміну з фіксацією місця останньої перестановки ті елементи кожного стовпчика матриці, що розташовані між мінімальним та максимальним елементами.
33. Задана матриця розміром $m \times n$. Впорядкувати за зростанням методом обміну з прапорцем елементи матриці, обходячи їх змійкою:



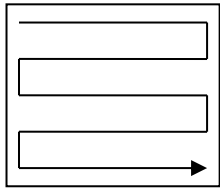
34. Задана матриця розміром $m \times n$. Впорядкувати за зростанням методом вибору елементи матриці, обходячи їх змійкою:



35. Задана матриця розміром $m \times n$. Впорядкувати за зростанням методом обміну з прапорцем елементи матриці, обходячи їх змійкою:



36. Задана матриця розміром $m \times n$. Впорядкувати за зростанням методом вибору елементи матриці, обходячи їх змійкою:



5.5.6. Запитання до самоконтролю

1. На які групи розподіляються методи сортування (на підставі реалізованих принципів)?
2. У чому полягають принципи основних прямих методів сортування?
3. Які різновиди прямих методів сортування ви можете навести? Яка різниця між базовим методом та наведеними вами варіантами?
4. Яка ідея реалізована у базовому методі прямого вибору?
5. В чому полягає ідея базового методу обміну?
6. Які ви можете навести варіанти методу обміну та в чому полягають покращення?
7. Поясніть базовий метод вставки та можливі модифікації.

5.6. Лабораторна робота № 6.

5.6.1. Тема

Структуровані типи даних. Символьні рядки

5.6.2. Мета роботи.

Навчитися працювати з символьною інформацією, використовувати особливості побудови символьних рядків і функції стандартної бібліотеки C++ для роботи з ними.

5.6.3. Основні теоретичні відомості

Представлення символьної інформації в C++. Тип **char** (скорочення від **charakter** – літера) віднесено до цілих базових типів. У більшості реалізацій C і C++ займає один байт і містить **ASCII**-код символу.

Символьна константа – це один або декілька охоплених апострофами символів. Константа може складатися з одного символу або послідовності символів, які починаються з символу **** (обернений слеш) і називаються керуючими або **escape-послідовностями**. Код символу у вісімковій, десятковій або шіснадцятковій системах числення також може бути заданий у вигляді escape-послідовності.

Приклади оголошення і ініціалізації символьних змінних

```
char a='a', b='5', sym='*', s;  
char a='\n';           // escape-послідовність  
char a='\235';         // десятковий код символу  
char b='\023';         // вісімковий код символу  
char c='\x23';         // шіснадцятковий код символу
```

Тип string. Для оголошення символьних рядків вбудованого типу немає, але можна підключити заголовковий файл **#include <string>** із стандартної бібліотеки **STL**, в якому визначений тип **string**. Цей тип визначений як клас, структуру якого визначає масив, складений із елементів

типу `char`. Для класу `string` визначені декілька конструкторів, низка операцій і методів.

Отже, об'єкт типу `string` можна оголосити і потім йому присвоїти значення, а можна одразу ініціалізувати значенням рядкового літерала або іншого рядка, наприклад:

```
string str1, str2("some info"), str4="words", str3(str2);  
str1="sentence";
```

Для змінних типу `string` визначені операції присвоєння `=`, виведення за допомогою `cout<<`, введення `cin>>`, конкатенації `+`, лексикографічного порівняння `<`, `>`, `<=`, `>=`, `==`, `!=`. Наприклад:

```
string name, surname, fullname;  
name="John";           //присвоєння  
cout << "What is your surname? "<<endl;  
cin>>surname;          //введення  
fullname=name+" "+surname; //конкатенація  
//виведення:  
cout << "Your full name is " << fullname<<endl;
```

Можливий результат роботи такої програми:

```
What is your surname?  
Smith  
Your full name is John Smith
```

Якщо рядок складений з декількох слів і навіть містить перехід на наступний рядок `\n`, він може бути виведений одним оператором `<<`. У наведеному прикладі рядок `fullname` складається з двох слів.

За допомогою `cin>>` рядок можна ввести тільки до першого пробільного символу. Тобто із речення буде введено тільки перше слово, решта залишиться у потоці до наступного застосування `cin>>`.

Для введення рядка із декількох слів використовують функцію `getline()`. Вона приймає два параметри: перший — `cin`, другий — змінна типу `string`. Наприклад:

```
string name;
```

```
cout << "Enter your full name: ";  
getline(cin, name);  
cout << "Your full name is " << name<<endl;
```

Використання `getline()` дає можливість введення цілого речення.

Можливий результат:

```
Enter your full name:  
Martin Luther King  
Your full name is Martin Luther King
```

Використання функції `getline()` після введення числової інформації може привести до неправильного результату із-за того, що в потоці введення залишається символ кінця рядка, який буде сприйнятий як порожній очікуваний рядок. Щоб цьому запобігти, потрібно цей символ видалити із потоку перед читанням потрібного рядка. Наприклад, за допомогою функції `cin.ignore(32767, '\n');` після введення останнього числа.

Деякі інші методи, визначені для типу **string**:

append — дописування у кінець рядка;

insert — вставка у задану позицію рядка інший рядок або його частину;

replace — заміна символів одного рядка на інший або його частин;

clear — видалення всіх символів рядка;

erase — видалення *k* символів з заданої позиції *pos*;

find (rfind) — пошук позиції першого входження підрядка-аргумента у даний рядок при проході з початку (з кінця);

substr — повертає підрядок даного рядка, з заданої позиції до кінця або задану кількість символів;

c_str — конвертує рядок `string` в рядок в стилі C;

length (size) — повертають довжину рядка без нуль-символу.

Наприклад:

```
string s1="The C++ language is the best",  
        s2="programming ";  
cout<<s1.size()<<endl;
```

```
s1.insert(8, s2);  
cout<<s1<<' '<<s1.length()<<endl;
```

У наведеному прикладі варіанти викликів деяких функцій.

Робота з символьним рядком як з масивом. З рядком можна також працювати і як із звичайним символьним масивом (говорять: в стилі C): оголошують масив типу **char** відповідного розміру з врахуванням нуля-символу як останнього символу рядка. Тоді можна користуватися функціями C.

Функції C при роботі з рядками дуже ефективні, але достатньо незручні і небезпечні. Вони вимагають глибокого розуміння внутрішніх механізмів реалізації і не мають розвинутих засобів відстеження помилок.

Рядковий літерал – константа типу символьного рядка в стилі C – послідовність довільних символів в подвійних лапках, обов'язково містить останній `'\0'`.

Приклади рядкових літералів: `"some information"`, `"words"`.

Приклади оголошення і ініціалізації символьних рядків безпосередньо як масивів:

```
char str1[20];  
str2[] = "1234", str3[]={ 'w', 'o', 'r', 'd', '\0' };  
char *strp="next sentence";
```

І в стилі C, і в стилі C++ для роботи з вмістом рядка можна застосовувати індексну форму, як для звичайного масиву, наприклад:

```
int i;  
for(i=0;str2[i]!='\0';i++) str1[i]=str2[i];  
str1[i]='\0';
```

Назва рядка, як і назва будь-якого масиву – константний вказівник на його перший символ, отже для роботи з його вмістом можна застосувати і форму доступу з вказівниками. Той самий приклад з вказівниками:

```
char *p, *q;  
for(p=str2, q=str1;*p!=0;p++, q++) *q=*p;  
*q=0;
```

Бібліотечні функції для роботи з рядками. Розглянемо декілька найуживаніших функцій по роботі з символами та рядками. Прототипи цих функцій розташовані у різних заголовкових файлах.

Заголовковий файл `<ctype.h>` (`<cctype>`) містить групу функцій, призначених для класифікації символів типу `char` та їх перетворення.

Таблиця 5.6 Деякі функції класифікації і перетворення `<ctype.h>`

Прототип	Призначення
<code>int isalnum(int c);</code>	символ <code>c</code> є ASCII-кодом латинської букви або цифри ('a'-'z', 'A'-'Z', '0'-'9')
<code>int isalpha(int c);</code>	символ <code>c</code> є ASCII-кодом латинської букви ('a'-'z', 'A'-'Z')
<code>int isascii(int c);</code>	символ <code>c</code> є ASCII-кодом (0-127)
<code>int isdigit(int c);</code>	символ <code>c</code> є ASCII-кодом десяткової цифри ('0'-'9')
<code>int iscntrl(int c);</code>	символ <code>c</code> є ASCII-кодом керуючого знаку (0x7f або 0x00-0x1f)
<code>int islower(int c);</code>	символ <code>c</code> є ASCII-кодом малої букви ('a'-'z')
<code>int isprint(int c);</code> <code>int isgraph(int c);</code>	символ <code>c</code> є ASCII-кодом видимого знаку (0x20-0x7e), (крім пропуску 0x20 для <code>isprint</code>)
<code>int ispunct(int c);</code>	символ <code>c</code> є ASCII-кодом розділового знаку
<code>int isspace(int c);</code>	символ <code>c</code> є ASCII-кодом пропуску
<code>int isupper(int c);</code>	символ <code>c</code> є ASCII-кодом великої букви ('A'-'Z')
<code>int isxdigit(int c);</code>	символ <code>c</code> є ASCII-кодом шіснадцяткової цифри ('0'-'9', 'a'-'f', 'A'-'F')
<code>int toascii(int c);</code>	перетворює цілі числа в ASCII-коди
<code>int tolower(int c);</code>	перетворює великі латинські букви у маленькі
<code>int toupper(int c);</code>	перетворює маленькі латинські букви у великі

Заголовковий файл `<stdlib.h>` (`<cstdlib>`) містить групу функцій, призначених для перетворення символьних рядків у числа, працює з символьними рядками `C`.

Таблиця 5.7 Деякі функції перетворення у числа `<stdlib.h>`

Прототип	Призначення
<code>double atof(const char *s);</code>	перетворює початкову частину рядка у число типу <code>double</code> ; очікуваний формат

	[+ -]digits[.][digits]([E e)[+ -]digits] повертає відповідне число або 0.0
float atoff(const char *s);	перетворює початкову частину рядка у число типу float
int atoi(const char *s);	Виділяє у рядку s перше ціле десяткове і перетворює його у змінну типу int, кінець – перший нецифровий символ
long atol(const char *s);	Виділяє у рядку s перше ціле десяткове і перетворює його у змінну типу long, кінець – перший нецифровий символ
double strtod(const char *str, char **tail);	Розширений варіант atof, другий параметр – адреса першого символу за числом
float strtodf(const char *str, char **tail);	Розширений варіант atoff, другий параметр – адреса першого символу за числом
long strtol(const char *s, char **ptr, int base);	Розширений варіант atol, другий параметр – адреса першого символу за числом, третій – система числення від 2 до 36, якщо 0, то розшифровка відповідно загальних правил 10/8/16

Заголовковий файл <string.h> містить групу функцій, призначених для операцій над символьними рядками. Ці функції починаються з **str...** Використання цих функцій може передбачати перетворення рядка або використання декількох рядків, тому важливо враховувати те, що сумарний рядок має бути достатнього розміру, що рядки не повинні перетинатися у пам'яті і що кожний рядок повинен закінчуватись нулем, інакше функція не працюватиме коректно.

Таблиця 5.8 Деякі функції роботи з символьними рядками <string.h>

Прототип	Призначення
char * strchr(const char *string, int c);	знаходить перше входження символу c до рядка string і повертає вказівник на нього або NULL, якщо такий не знайдено
char * strrchr(const char *string, int c);	знаходить останнє входження символу c до рядка string і повертає вказівник на нього або NULL, якщо такий не знайдено
char * strcat(char *dst, const char *src);	долучає копію рядка src разом з /0 до рядка dst так, що перший символ src заміщує останній /0 в dst
char * strncat(char *dst, const char *src, size_t length);	аналог strcat для перших length символів src (або до /0, якщо рядок коротший за length)
int strcmp(const char *a, const char *b);	виконує лексикографічне порівняння двох рядків (-1,0,1)
int strncmp(const char *a,	аналог strcmp для перших length символів

<code>const char * b, size_t length);</code>	
<code>char *strcpy(char *dst, const char *src);</code>	виконує копіювання рядка <code>src</code> до масиву <code>dst</code>
<code>char *strncpy(char *dst, const char *src, size_t length);</code>	аналог <code>strcpy</code> для перших <code>length</code> символів
<code>size_t strlen(const char *str);</code>	рачує довжину рядка (без <code>/0</code>)
<code>char *strstr(const char *s1, const char *s2);</code>	знаходить перше входження символів рядка <code>s2</code> (без <code>/0</code>) до рядка <code>s1</code> і повертає вказівник на знайдений підрядок або <code>NULL</code> , якщо такий не знайдено
<code>size_t strspn(const char *s1, const char *s2);</code>	рачує кількість букв із <code>s2</code> , які зустрічаються на початку <code>s1</code> , навіть якщо порядок їх слідування не співпадає
<code>char *strtok(char *source, const char *delimiters)</code>	у результаті серії викликів розбиває рядок <code>source</code> на лексеми, розділені знаками із <code>delimiters</code>

5.6.4. Приклади програм опрацювання символної інформації

Приклад 1. Програма з використанням рядка як масиву символів в стилі C. З клавіатури вводиться речення, серед слів якого можуть зустрічатися дійсні додатні числа. Написати програму, яка знаходить значення найбільшого з цих чисел або виводить -1. Відомо, що кількість символів речення не перевищує 100, слова речення розділені пропусками і, можливо, комами.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX 100
int main(){
    char sent[MAX], *pw, delim[]=" ,"; //розділювачі -пропуск і кома
    double temp, max=-1; //оскільки усі можливі числа - додатні
    puts("enter sentence:\n");
    gets(sent);
    pw=strtok(sent, delim); //поділити на лексеми
    do{
        if (isdigit(*pw)){ //якщо перший символ слова - цифра
            temp= atof(pw); //перетворити його на дійсне число
            if(temp>max) max=temp;
        }
    }
```

```

        pw=strtok(NULL,delim);
    }while (pw!=NULL);
printf("%lf\n", max);
return 0;
}

```

Приклад 2. Приклад програми, виконаної з використанням типу **string**.

З клавіатури вводиться речення. Сформувати масив із окремих слів цього речення і вивести їх на екран у стовпчик.

```

#include <string>
#include <iostream>
using namespace std;
int main() {
    const int MAX=120;
    string str, a[10];
        cout<<"enter sentence:\n";
        getline(cin,str);
int len=str.length(), j=0, i=1, mas[10]={0}; //індекси пропусків
    for(int j=0; j<len; j++)
        if(str[j]==' ') mas[i++]=j;
    a[0]=str.substr(0,mas[1]); i=1;
    cout<<a[0]<<endl;
    while(mas[i]!=0){
        int b=mas[i]+1, e=mas[i+1]-mas[i];
        a[i]=str.substr(b,e);
        cout<<a[i]<<endl; i++;
    }
return 0;
}

```

Для коректної роботи наведеного прикладу потрібно, щоб у введеному реченні не було розділових знаків і слова були розділені одним пропуском.

5.6.5. Порядок виконання роботи

Порядок підготовки звіту та виконання ЛР6 відповідає загальним вимогам, викладеним в розділі 3.

Доведення працездатності програми буде відбуватися на основі тестування роботи програми на підготовлених і перевірених здобувачем прикладах. Обмеження на максимальну кількість символів або максимальну кількість слів для уведеної з клавіатури фрази визначити самостійно.

Для захисту роботи необхідно виконати в аудиторії один із варіантів завдань, вказаний викладачем.

5.6.6. Варіанти завдань

1. З клавіатури увести речення, видалити в ньому усі подвоєні слова. Подвоєними вважати розташовані поряд однакові слова.
2. З клавіатури увести текст, який може складатися із декількох речень, розділених крапками, знаками оклику або запитання. Перевірте, чи починаються речення з великої букви і, якщо ні, виправте помилки.
3. З клавіатури увести послідовність слів, організованих як речення. Відсортувати слова цього речення у лексикографічному порядку.
4. З клавіатури увести фразу. Вивести її слова у зворотному порядку (реверсувати).
5. З клавіатури увести фразу. Перевірити, чи зміниться вона, якщо слова прочитати у зворотному порядку.
6. З клавіатури увести текст, який може складатися із декількох речень, розділених крапками, знаками оклику або запитання. Виведіть кожне речення з нового рядочка.
7. З клавіатури увести фразу. Перевірити, чи утворюють символи цієї фрази паліндром (текст, симетричний відносно середини). Пропуски, розділові знаки і різницю в регістрах – ігнорувати.
8. З клавіатури увести фразу. Викинути із усіх речень фрази деяке слово (також уводиться з клавіатури). Різницю в регістрах – ігнорувати.
9. З клавіатури увести фразу. Перевірити, чи містить ця фраза слова-паліндроми (симетричні відносно середини слова).

10. З клавіатури увести фразу. Замінити в усіх реченнях фрази деяке слово іншим (обидва також уводяться з клавіатури). Різницю в регістрах – враховувати.
11. З клавіатури увести текст деякої програми. Виділити в ньому великими буквами ключові слова. Список ключових слів надано іншим реченням, заданим у вигляді рядкового літерала.
12. З клавіатури послідовно уводяться речення англійською мовою. Визначити, з якою частотою у тексті зустрічається кожна з букв мови (частотний аналіз всього тексту). Кількість речень визначає користувач інтерактивно.
13. З клавіатури увести фразу. Найдовше і найкоротше слова в ньому записати великими буквами, а зайві пропуски між словами видалити.
14. З клавіатури увести фразу, яка представляє собою вираз, тобто складається із чисел і знаків чотирьох арифметичних операцій без дужок. Про наявність пропусків інформації немає. Порахуйте і виведіть результат.
15. Зашифрувати уведений з клавіатури текст відповідно кодової таблиці, яка задана як рядковий літерал із послідовності символів, кожний з яких відповідає певній букві алфавіту. Вхідний текст складено тільки із малих букв латинського алфавіту і десяти цифр.
16. Розшифрувати уведений з клавіатури закодований текст відповідно кодової таблиці, яка задана як рядковий літерал із послідовності символів, кожний з яких відповідає певній букві алфавіту (малі англійські, відомо, що цифр і розділових знаків немає).
17. З клавіатури ввести фразу, у якій замість пропущених слів стоять символи '\$' . Пропущені слова у правильному порядку задані у другому реченні, також уведеному з клавіатури. Розставити пропущені слова правильно і вивести фразу, що утворилась.
18. У введеній з клавіатури фразі зустрічаються дійсні числа. Замінити їх цілими, врахувати правила округлення чисел. Вивести модифікований текст.

19. У введеному з клавіатури тексті знайти слова, які можуть бути складені із двох інших слів із цього ж тексту.
20. У введеному з клавіатури тексті знайти слово, яке може бути складене із букв одного з інших слів із цього ж тексту, можливо, складених інакшим способом, регістр ігнорувати.
21. Введений з клавіатури текст містить записаний через коми список, кожний елемент якого складений з ідентифікатора деякої змінної, символу '=' і значення відповідного типу (записане як цілий, дійсний або символьний літерал). Сформуванати відповідний рядок формату, яким можна скористатися у функції `printf()`.
22. У введеному з клавіатури тексті знайти слово, яке може бути складене із букв одного з інших слів із цього ж тексту, просто складених у іншій послідовності, регістр враховувати.
23. У введеному з клавіатури тексті знайти слова, які починаються і закінчуються на однакові букви. Регістри не враховувати.
24. У введеному з клавіатури тексті видалити усі подвоєні букви – окрім заданих у деякому рядковому літералі, подвоєні розділові знаки і подвоєні пропуски.
25. З клавіатури заповнити масив цін товарів у копійках. За значеннями елементів цього масиву заповнити масив речень, кожне з яких представляє собою текст з ціною відповідного товару у гривнях і копійках. Наприклад 325 перетворити на 3 гривні 25 копійок.
26. Кожний символ тексту можна замінити на наступний символ алфавіту, (наприклад, 'a' замінити на 'b') або з деяким зсувом n (наприклад, якщо $n=3$, то 'a' замінити на 'd').
 - a) Зашифрувати уведений з клавіатури текст шляхом заміни кожного символу на зсунутий на деяке n інший символ.
 - b) Розшифрувати уведений з клавіатури текст, якщо він був зашифрований шляхом заміни кожного символу на зсунутий на деяке n інший символ.

27. Введений з клавіатури англійський текст без розділових знаків і зайвих пропусків зашифрувати шляхом заміни кожної букви її порядковим номером у англійському алфавіті, записаному як набір із двох символів. Наприклад, для 'а' це буде 01, для 'j' – 10. Код пропуску – 30. Регістр ігнорувати.
28. Для гарантування надійності під час передачі каналами зв'язку послідовностей з нулів і одиниць, замість одного символу передають підряд три однакових: 1 – як 111, 0 – як 000. Скоріше всього, що якщо збій трапиться, він буде одиночний, тож із трьох однакових символів однаковими залишаться два. З клавіатури уводиться деяка послідовність з нулів і одиниць, її потрібно перетворити на послідовність з потроєними символами, внести туди випадкові «збої», а потім відновити початкову послідовність.
29. З клавіатури увести фразу, яка представляє собою арифметичний вираз, тобто складається із чисел, знаків бінарних арифметичних операцій і дужок. Визначити, чи коректно побудований цей вираз: перевірити розстановку дужок, порядок розташування літералів і знаків операцій.
30. Написати програму, яка формує текст вітальної листівки. З клавіатури увести три речення. В першому – послідовність імен і по батькові співробітників, до кого листівка буде адресована, у другому – назви свят, у третьому – шаблон тексту листівки, куди буде вставлятися кого вітають і з яким святом (порядок слідування у першому і другому реченнях співпадають).
31. З клавіатури увести фразу, у якій потрібно зробити контекстну заміну. Потрібні послідовності символів уводити з клавіатури.
32. З клавіатури увести фразу, у якій потрібно перенести усі розділові знаки в кінець речення.
33. З клавіатури увести фразу, у якій всі власні імена замінити групами зірочок. Власними іменами вважати слова у середині речення, які починаються з великої букви. Наприклад `Standing on the River`

Thames in the south east of the island of Great Britain, London has been a major settlement for two millennia
перетвориться на Standing on the *** *** in the south east of the island of *** ***, *** has been a major settlement for two millennia.

34. З клавіатури увести текст, який може складатися із декількох речень, розділених крапками, знаками оклику або запитання. Перетворіть цей текст так, щоб довжина кожного з речень не перевищувала 20 символів (відкиданням кінцівки речення, якщо воно довше за 20).
35. З клавіатури увести текст, складений із двох речень, що закінчуються крапками. Визначити, чи входить друге речення цілком у перше (без останньої крапки).
36. З клавіатури увести речення. Визначити, яке словосполучення із чотирьох букв зустрічається в ньому більше одного разу. Регістри ігнорувати.

5.6.7. Запитання до самоконтролю

1. Що таке рядок і як він оголошений в C?
2. Як виглядає символьний літерал? Рядковий літерал?
3. Як рядок можна ініціалізувати?
4. Як ввести значення одного символу?
5. Як з клавіатури ввести рядок?
6. Які функції класифікації і перетворення ви знаєте?
7. Які склеїти два рядка? Що при цьому потрібно врахувати?
8. Що таке лексикографічне порівняння? Як порівняти два рядка?
9. Які функції перетворення символьної послідовності в число і навпаки ви знаєте?

5.7. Лабораторна робота № 7.

5.7.1. Тема

Створення і використання функцій

5.7.2. Мета роботи.

Навчитися створювати власні функції і правильно користуватися механізмом передачі параметрів.

5.7.3. Основні теоретичні відомості

Функція – логічно завершений іменований фрагмент програми, призначений для виконання деякої задачі. Унікальний ідентифікатор функції використовують для її виклику із іншої функції.

Використання функцій дозволяє один і той же фрагмент програми використовувати повторно, забезпечує кращу структурованість програми, що полегшує читання коду та його модифікацію.

Функції – основні складові програми мовою C++ і формують окремий тип даних.

Будь-яка функція повинна бути оголошена, визначена і викликана.

Оголошення (заголовок, прототип) повинно знаходитися до її виклику. Синтаксис оголошення:

```
тип ідентифікатор ([список_формальних_параметрів]);
```

Визначення функції включає її оголошення і опис (тіло) функції. Визначення функції завжди одне. В тексті програми може знаходитись як до, так і після опису функції, що її викликає. Синтаксис:

```
тип ідентифікатор ([список_формальних_параметрів]) {  
    /* тіло функції */ };
```

Наприклад

```
int func(unsigned char c)
{
    if (c>='A' && c<='S') return 1;
    else
        return 0;
}
```

Функція повертає значення, якщо її виконання закінчується оператором **return**, що містить деякий вираз. Оператор **return** перериває виконання функції. Функція може містити декілька операторів **return**, а може не містити жодного і тоді повертати тип **void**.

Порядок і типи формальних параметрів повинні бути однаковими у визначенні функції і у всіх її оголошеннях. Тип формального параметра може бути будь-яким основним типом, структурою, об'єднанням, перерахуванням, вказівником або масивом.

Виконання функції починається з першого виконаного оператора в момент виклику із іншої функції. Синтаксис виклику:

ідентифікатор ([список_фактичних_параметрів]);

Типи фактичних параметрів при виклику функції повинні бути сумісні з типами відповідних формальних параметрів.

Наприклад:

```
int add(int x, int y){return x+y;};//опис
. . .
int i,var;
var=add(i,24); //виклик
```

Передача масивів у функцію може бути виконана або з використанням їх ідентифікаторів, або вказівника на нульовий елемент. Наприклад, для одновимірного масиву це може виглядати так:

```
void gen (int n,int x[max]);
void out (int n,int *p);
void sort (int n,int x[]);
```

Виклик для усіх трьох випадків – однаковий, наприклад:

```
gen(10,a);
```

Передача двовимірного масиву у функцію, приклади прототипів:

```
void gen (int x[max1][max2], int n, int m);
void out (int x[][max2], int n, int m);
void sort (int *p, int n, int m);
```

Виклики у перших двох випадках виглядатимуть так:

```
gen(x, n, m);
```

В останньому наприклад так:

```
sort (&a[0][0], int n, int m);
```

Із будь-якої функції можна викликати будь-яку іншу функцію, або саму себе – так званий рекурсивний виклик.

Оскільки в C++ визначений функціональний тип, функція може бути передана до іншої функції як параметр за допомогою вказівника на функцію.

Синтаксис оголошення вказівника на функцію:

```
<тип>(*ідентифікатор_вказівника) ([список_типів_
    формальних_параметрів]);
```

Наприклад, вказівник на функцію з двома параметрами `int` та `int*`, яка повертає `int`, матиме оголошення

```
int(*pf)(int, int*)
```

яке може бути використане для будь-якої функції з таким прототипом. Ідентифікатор функції, як і ідентифікатор масиву сам по собі є вказівником, отже допустимим буде таке присвоєння:

```
int func(int a, int* arr); //прототип функції
...
pf=func;
```

5.7.4. Приклад з використанням вказівника на функцію

Написати функцію визначення максимального або мінімального елемента масиву, у яку як параметри передаються: сам масив, його розмірність і вказівник на функцію для порівняння.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define n 10

double less(double a, double b) { if(a<b) return a;return b;}
double grt(double a, double b) { if(a>b) return a;return b;}

void gen(int n, double a[]){int i=0;
    for(;i<n;i++)a[i]=0.01*(rand()%100);};

double extremum(int n, double *a, double(*pf)(double, double)){
    double max=a[0]; int i=1;
    for(; i<n; i++) max=pf(max,a[i]);
    return max;
}

void out(int n, double*p){
```

```

while(n-1){
    cout<<*p<<' ';p++;n--;
};
cout<<endl;
};

int main(){
    srand(time(0));
    double a[10], m;
    gen(n,a); out(n,a);
    m=extremum(n, a, less); cout<< "min="<<m<<endl;
    m=extremum(n, a, grt); cout<< "max="<<m<<endl;
    return 0;
}

```

5.7.5. Порядок виконання роботи

Порядок підготовки звіту та виконання ЛР7 відповідає загальним вимогам, викладеним в розділі 3.

Звернути увагу, що у кожному завданні необхідно передбачити використання декількох функцій, для однієї з яких обов'язковим є використання вказівника на функцію.

Доведення працездатності програми буде відбуватися на основі тестування роботи програми на підготовлених і перевірених здобувачем прикладах. Для захисту текст програми може бути модифікований.

5.7.6. Варіанти завдань

1. Реалізувати функцію, у яку передається деякий масив дійсних чисел та його розмірність і яка перетворює кожний елемент цього масиву, крім останнього, відповідно певного правила. Це правило передавати як вказівник на функцію, яка буде застосована до поточного і наступного елемента масиву. Передбачити дві функції з двома параметрами. Перша – повернути середнє арифметичне цих двох чисел. Друга – повернути корінь квадратний із квадратів цих двох чисел. Продумати тестовий приклад для демонстрації роботи програми.

2. Потрібно побудувати таблицю, кожний рядок якої містить значення аргументу і відповідне значення функції обчислення арктангенса для даного значення аргументу. Значення аргументу задається на заданому інтервалі і з заданим кроком. Оформити у вигляді функції, в яку як параметри передаються границі інтервалу, крок і вказівник на потрібну функцію обчислення арктангенса – або наближеного з точністю 0.001 за формулою

$$\arctg x = x - x^3/3 + x^5/5 - \dots + (-1)^n x^{2n+1}/(2n+1) + \dots$$

або з використанням функції із стандартної бібліотеки.

3. Потрібно побудувати таблицю, кожний рядок якої містить значення аргументу і відповідне значення функції обчислення експоненти для даного значення аргументу. Значення аргументу задається на заданому інтервалі і з заданим кроком. Оформити у вигляді функції, в яку як параметри передаються границі інтервалу, крок і вказівник на потрібну функцію обчислення експоненти – або наближеного з точністю 0.001 за формулою

$$e^x = 1 + x/1! + x^2/2! + \dots + x^n/n! + \dots$$

або з використанням функції із стандартної бібліотеки.

4. Потрібно побудувати таблицю, кожний рядок якої містить значення аргументу і відповідне значення функції обчислення натурального логарифму для даного значення аргументу. Значення аргументу задається на заданому інтервалі і з заданим кроком. Оформити у вигляді функції, у яку як параметри передаються границі інтервалу, крок і вказівник на потрібну функцію обчислення натурального логарифму – або наближеного з точністю 0.01 за формулою

$$\ln(1+x) = x - x^2/2 + x^3/3 - \dots + (-1)^{n-1} x^n/n + \dots$$

або з використанням функції із стандартної бібліотеки.

5. Потрібно побудувати таблицю, кожний рядок якої містить значення аргументу і відповідне значення функції обчислення косинусу для даного

значення аргументу. Значення аргументу задається на заданому інтервалі і з заданим кроком. Оформити у вигляді функції, у яку як параметри передаються границі інтервалу, крок і вказівник на потрібну функцію обчислення косинусу – або наближеного з точністю 0.01 за формулою

$$\cos x = 1 - x^2/2! + x^4/4! - \dots + (-1)^n x^{2n}/(2n)! + \dots$$

або з використанням функції із стандартної бібліотеки.

6. Надано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε :

$$\sum_{k=0}^{\infty} \frac{(-1)^k x^{4k+1}}{(2k)!(4k+1)}$$

Реалізувати у вигляді функції обчислення суми з наданою точністю, в яку як параметр передається функція визначення чергового доданку.

7. Надано дійсні числа x, ε ($x \neq 0, 0 < \varepsilon < 1$). Обчислити з точністю до ε :

$$\sum_{k=0}^{\infty} \frac{(-1)^k x^{2k+1}}{k!(2k+1)}$$

Реалізувати у вигляді функції обчислення суми з наданою точністю, в яку як параметр передається функція визначення чергового доданку.

8. Реалізувати функцію, у яку передається деякий масив цілих чисел та його розмір і яка підраховує кількість елементів цього масиву, які задовольняють певній умові. Цю умову передавати як вказівник на функцію. Передбачити дві функції – визначення чи є дане число простим, і визначення чи є дане число парним. Продумати тестовий приклад для демонстрації роботи програми.

9. Реалізувати функцію, у яку передається деякий масив цілих чисел та його розмірність і яка підраховує кількість елементів цього масиву, які задовольняють певній умові. Цю умову передавати як вказівник на функцію. Передбачити дві функції – визначення, чи є дане число квадратом іншого

натурального числа, і визначення, чи містить дане число п'ятірку у будь-якій позиції свого десяткового запису. Продумати тестовий приклад для демонстрації роботи програми.

10. Методом половинного ділення з точністю до ε знайти приблизне значення кореня $f(x)=0$, де $f(x) = x + \ln(x + 0.5) - 0.5$ на інтервалі $(0;2)$. Реалізувати метод пошуку кореня у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

11. Методом половинного ділення на інтервалі $[1;1.1]$ з точністю до ε знайти приблизне значення кореня $f(x)=0$, де $f(x) = x^5 - x - 0.2 = 0$. Реалізувати метод пошуку кореня у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

12. Методом половинного ділення на інтервалі $[0, \pi/2]$ з точністю до ε знайти приблизне значення кореня $f(x)=0$, де $f(x) = \frac{2 \sin^2 x}{3} - \frac{3 \cos^2 x}{4}$; Реалізувати метод пошуку кореня у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

13. Методом половинного ділення на інтервалі $[-1.2 ; -0.5]$ з точністю до ε знайти приблизне значення кореня $f(x)=0$, де $f(x) = x^4 + 0.2x^3 - 0.4x^2 - 1.4x - 1.2 = 0$ Реалізувати метод пошуку кореня у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

14. Методом половинного ділення на інтервалі $[3.7 ; 5]$ з точністю до ε знайти приблизне значення кореня $f(x)=0$, де $f(x) = x^4 - 4.1x^3 + x^2 - 5.1x + 4.1 = 0$ Реалізувати метод пошуку кореня у

вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

15. Методом трапецій обчислити визначений інтеграл: $\int_1^2 \frac{1}{x^2} dx$. Реалізувати метод пошуку інтеграла у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

16. Методом трапецій обчислити інтеграл: $\int_1^9 \frac{1}{x} dx$. Реалізувати метод пошуку інтеграла у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

17. Методом трапецій обчислити інтеграл: $\int_0^{\pi/3} \sin x dx$. Реалізувати метод пошуку інтеграла у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

18. Методом трапецій обчислити інтеграл: $\int_0^1 \frac{1}{1+x^2} dx$. Реалізувати метод пошуку інтеграла у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

19. Методом трапецій обчислити інтеграл: $\int_0^{\pi/3} \cos x dx$. Реалізувати метод пошуку інтеграла у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

20. Методом прямокутників обчислити інтеграл: $\int_0^{1.2} \frac{1}{\sqrt[3]{x+1}} dx$. Реалізувати метод пошуку інтеграла у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

21. Методом прямокутників обчислити інтеграл: $\int_{\pi/2}^{\pi} \frac{\cos x}{x} dx$. Реалізувати

метод пошуку інтеграла у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

22. Методом прямокутників обчислити інтеграл: $\int_0^3 \sqrt{4+x^2} dx$. Реалізувати

метод пошуку інтеграла у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

23. Методом прямокутників обчислити інтеграл: $\int_0^6 \frac{1}{\sqrt{1+x^4}} dx$. Реалізувати

метод пошуку інтеграла у вигляді функції, в яку як параметр передається функція, для якої розв'язується задача.

24. Реалізувати функцію, у яку передається символьний рядок і яка підраховує кількість слів цього рядка, які задовольняють певній умові. Цю умову передавати як вказівник на функцію. Передбачити дві функції – визначення чи утворюють букви даного слова паліндром, і визначення чи починається дане слово з великої букви. Продумати тестовий приклад для демонстрації роботи програми.

25. Реалізувати функцію, у яку передається символьний рядок і яка підраховує кількість слів цього рядка, які задовольняють певній умові. Цю умову передавати як вказівник на функцію. Передбачити дві функції – визначення чи містить дане слово цифри, і визначення чи містить дане слово більше п'яти символів. Продумати тестовий приклад для демонстрації роботи програми.

26. Реалізувати функцію, у яку передається деякий масив цілих чисел та його розмірність і яка перетворює кожний елемент цього масиву відповідно певного

правила. Це правило передавати як вказівник на функцію. Передбачити дві функції – перша: заміна елемента на суму цифр його десяткового запису, і друга: обнуління елемента, абсолютне значення якого менше 5. Продумати тестовий приклад для демонстрації роботи програми.

27. Реалізувати функцію, у яку передається двовимірний масив дійсних чисел та його розмірності і яка перетворює кожний рядок цього масиву відповідно певного правила. Це правило передавати як вказівник на функцію. Передбачити дві функції – перша: визначити максимальний і мінімальний елементи одновимірного масиву і замінити усі розташовані між ними елементи нулями, друга: замінити нулями усі ті елементи одновимірного масиву, абсолютне значення яких менше 0.01. Продумати тестовий приклад для демонстрації роботи програми.

28. Реалізувати функцію, у яку передається двовимірний масив дійсних чисел та його розмірності і яка рахує ті рядки цього масиву, що задовольняють певній умові. Цю умову передавати як вказівник на функцію. Передбачити дві функції – перша: визначити, чи впорядкований одновимірний масив за зростанням; друга – чи впорядкований одновимірний масив за спаданням. Продумати тестовий приклад для демонстрації роботи програми.

29. Реалізувати функцію, у яку передається двовимірний масив цілих чисел та його розмірності і яка рахує ті рядки цього масиву, що задовольняють певній умові. Цю умову передавати як вказівник на функцію. Передбачити дві функції – перша: визначити, чи заповнений одновимірний масив нулями більше, ніж наполовину і друга: чи присутня у одновимірному масиві хоча б одна пара сусідніх однакових чисел. Продумати тестовий приклад для демонстрації роботи програми.

30. Реалізувати функцію, у яку передається двовимірний масив цілих чисел **arr** та його розмірності і яка будує одновимірний масив, кожний елемент якого обчислений на основі правила, застосованого до відповідного рядка **arr**. Це правило передавати як вказівник на функцію. Передбачити дві функції – перша: визначити максимальний елемент одновимірного масиву, друга: порахувати суму елементів одновимірного масиву. Продумати тестовий приклад для демонстрації роботи програми.

31. Реалізувати функцію, у яку передається символьний рядок і яка підраховує кількість слів цього рядка, які задовольняють певній умові. Цю умову передавати як вказівник на функцію. Передбачити дві функції: перша – визначити, чи містить дане слово подвоєння букв; друга – визначити, чи складається дане слово виключно з великих букв. Продумати тестовий приклад для демонстрації роботи програми.

32. Реалізувати функцію, у яку передається деякий масив цілих чисел та його розмірність і яка перетворює кожний елемент цього масиву відповідно певного правила. Це правило передавати як вказівник на функцію. Передбачити дві функції – перша: замінити значення цілого числа остачею від ділення його на 16, і друга: обнуління цілого числа, якщо віно від’ємне. Продумати тестовий приклад для демонстрації роботи програми.

5.7.7. Запитання до самоконтролю

1. Що таке функція і як її оголошують в C++?
2. Поясніть поняття прототип, опис, виклик. Скільки разів може бути оголошена одна функція? Описана? Викликана?
3. Що таке формальні параметри? Фактичні? Як вони між собою зв’язані?
4. Як можна передати у функцію масив?
5. Як можна передати у функцію рядок?
6. Що означає передача параметрів за значенням?
7. Що означає передача параметрів за посиланням?
8. Що таке глобальні, що таке локальні оголошення?

6. СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

6.1. Базова

1. Програмування та алгоритмічні мови 1. Алгоритмізація та основи програмування: Конспект лекцій [Електронний ресурс]: навч. посіб. для студ. спеціальності 124 «Системний аналіз», освітньо-професійні програми «Системний аналіз та управління», «Системний аналіз фінансового ринку» / Уклад.: І. В. Назарчук. Київ: КПІ ім. Ігоря Сікорського, 2020. 142 с. URL: <https://ela.kpi.ua/handle/123456789/31849>
2. Програмування та алгоритмічні мови. Алгоритмізація та основи програмування: Методичні вказівки до комп'ютерного практикуму для студентів напрямів підготовки 124 «Системний аналіз» [Електронний ресурс]: / Уклад.: І. В. Караюз, І.В. Назарчук, О.Л. Тимощук. Київ : НТУУ «КПІ», 2017. 94 с. URL: [http://mmsa.kpi.ua/disciplines/sac/1ii/Програмування та алгоритмічні мови/Методичні вказівки до комп'ютерного практикуму.pdf](http://mmsa.kpi.ua/disciplines/sac/1ii/Програмування_та_алгоритмічні_мови/Методичні_вказівки_до_комп'ютерного_практикуму.pdf)
3. Ковалюк Т.В. Основи програмування Київ: BVH, 2005. 384с.
4. С++. Основи програмування. Теорія та практика: підручник/ О.Г. Трофименко, Ю.В. Прокоп, І.Г. Швайко, Л.М. Буката та ін.; за ред. О.Г.Трофименка. Одеса: Фенікс, 2010. 544 с. URL: <http://dspace.onua.edu.ua/handle/11300/23501>

6.2. Допоміжна

5. Brian W. Kernighan, Dennis M. Ritchie. C Programming Language. 2nd edition/ Pearson; 2nd edition. – 272p. URL: <https://kremlin.cc/k&r.pdf>
6. Войтенко В. В., Морозов А. В. С та С++. Теорія та практика. Житомир: Видавництво ЖДТУ, 2004. 324с. Бібліотека.
7. Шпак З. Я. Програмування мовою С. Навч. посібн. Львів: Видавництво Львівської політехніки, 2011. 436с. Бібліотека.
8. Задачі з програмування /С.А. Абрамов, Г.Г.Гнезділова, О.Н. Капустіна, М.І. Селюн. М.: Наука, 1988. 224с.

6.3. Інформаційні ресурси

9. Електронний кампус НТУУ «КПІ ім. І. Сікорського» [сайт] / Єдине інформаційне середовище НТУУ «КПІ ім. І. Сікорського», URL: <http://campus.kpi.ua>
10. Библиотека MSDN /Microsoft: [сайт]/ MSDN Library Documentation, 2023. URL: <https://learn.microsoft.com/en-us/cpp/standard-library/cstdlib?view=msvc-170>

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НН ІІСА НТУУ «КПІ ІМ.ІГОРЯ СІКОРСЬКОГО»
КАФЕДРА ММСА**

**Лабораторна робота № N
Варіант № NN
з дисципліни:
Програмування та алгоритмічні мови**

Виконав:

Студент І курсу групи КА-NN

Прізвище Ім'я

Перевірив: Прізвище N.N.

Київ – 20XX р.