

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет "Запорізька політехніка"

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторних робіт із дисципліни
"Програмування" (І частина)

для студентів спеціальності 123 "Комп'ютерна інженерія"
денної форми навчання

Методичні вказівки до виконання лабораторних робіт із дисципліни "Програмування" (І частина) для студентів спеціальності 123 "Комп'ютерна інженерія" денної форми навчання / Укл.: Н.В. Луценко, О.В. Польська. – Запоріжжя: НУ "Запорізька політехніка", 2021. – 47 с.

Укладачі:

Н.В. Луценко, ст. викладач
О.В. Польська, ст. викладач

Рецензент:

Р.К. Кудерметов, к.т.н., доцент

Відповідальний за випуск

Н.В. Луценко, ст. викладач

Затверджено
на засіданні кафедри КСМ
Протокол № 6 від 05.02.2021

Рекомендовано до видання
на засіданні НМК факультету КНТ
Протокол № 7 від 26.02.2021

ЗМІСТ

Порядок виконання лабораторних робіт.....	4
Поради та рекомендації.....	5
Загальні та теоретичні відомості	6
Лабораторна робота №1. Базові типи змінних. Елементарне введення-виведення даних	16
Контрольні питання до лабораторної роботи № 1	21
Лабораторна робота №2. Оператори вибору.....	22
Контрольні питання до лабораторної роботи № 2.....	26
Лабораторна робота №3. Оператори циклу.....	27
Контрольні питання до лабораторної роботи № 3	31
Лабораторна робота №4. Масиви і показчики	32
Контрольні питання до лабораторної роботи № 4.....	38
Лабораторна робота №5. Двовимірні та динамічні масиви	39
Контрольні питання до лабораторної роботи № 5	45
Рекомендована література.....	46
Додаток. Таблиця кодів асції основних символів	47

ПОРЯДОК ВИКОНАННЯ ЛАБОРАТОРНИХ РОБІТ

Лабораторні роботи складаються з декількох програм, завдань до них і контрольних питань. Студент повинен розробити, відпрацювати і зберегти у робочому каталозі програму, що наведена, й програму, розроблену згідно заданого варіанта, та проаналізувати результати їхньої роботи.

Програми, що позначені зірочкою, призначені для самостійної роботи (не обов'язкові). За результатами виконання кожної лабораторної роботи студент складає й захищає письмовий звіт, вимоги до якого викладені нижче.

ЗМІСТ ЗВІТУ

1. Титульний лист, виконаний відповідно до СТП.
2. Мета роботи.
3. Відповіді на контрольні питання.

ПОРАДИ ТА РЕКОМЕНДАЦІЇ

Робота в середовищі програмування Visual C++

Запускаємо середовище програмування **Microsoft Visual C++**. При цьому відкривається початкова сторінка, в якій потрібно створити порожній проект. Для цього обирається вкладка **File (файл)** та пункт **New->Project (Создать->Проект)**. У вікні, що з'явилося, обирається **Visual C++**, а справа обирається піктограма **Empty project (Пустой проект)**. Нижче задається назва проекту та вказується папка, в якій буде зберігатися цей проект.

Для додавання в проект файлу з кодом програми використовується меню **Project (Проект)** та пункт **Add New Item (Добавить новый элемент)**. У вікні, що з'явилося, обирається піктограма **C++File.cpp (файл.cpp)**, а нижче задається ім'я нового файлу. Далі натискається кнопка **Add (Добавить)**. Відкривається сторінка редактора, в якому пишеться код програми. Для запуску програми обирається значок ► (**Start Debugging**).

Для відкриття створеної програми обирається вкладка **File (файл)** та пункт **Open->Project (Открыть->Проект)**.

Робота в середовищі програмування Dev-C++

Запускаємо середовище програмування **Dev c++ IDE**. При цьому відкривається початкова сторінка, в якій потрібно створити порожній файл. Для цього обирається вкладка **File (файл)** та пункт **New->File (Создать->Исходный файл)**. Відкривається сторінка редактора, в якому пишеться код програми. Далі обирається вкладка **File (файл)** та пункт **Save or Save As (Сохранить або Сохранить как)** і задається назва файлу та вказується папка, в якій буде зберігатися цей файл.

Для запуску програми обирається вкладка **Excute (Выполнить)** і пункт **Compile (Скомпилировать)**, потім пункт **Run (Выполнить)** або пункт **Compile and run (Скомпилировать и выполнить)**.

Для відкриття створеної програми обирається вкладка **File (файл)** та пункт **Open->File (Открыть-> файл)**.

Для налаштування середовища обирається вкладка **Tools (Сервис)**.

ЗАГАЛЬНІ ТА ТЕОРЕТИЧНІ ВІДОМОСТІ

Програма мовою Сі – це послідовність препроцесорних директив, описів та визначень глобальних змінних та функцій.

Базові типи змінних. Для визначення змінних цілого типу даних використовують ключові слова **char**, **int**, **short**, **long**. Слова **signed**, **unsigned** позначають знакове чи беззнакове ціле число, за умовчанням всі числа є знакові. Для визначення змінних дійсного типу даних використовують ключові слова **float**, **double**.

Базові цілі типи даних

Тип	Розмір	Діапазон
char	1 байт (8 біт)	$-2^7 \div 2^7 - 1$
unsigned char	1 байт (8 біт)	$0 \div 2^8 - 1$
short	2 байт (16 біт)	$-2^{15} \div 2^{15} - 1$
unsigned short	2 байт (16 біт)	$0 \div 2^{16} - 1$
int	4 байт (32 біт)	$-2^{31} \div 2^{31} - 1$
unsigned int	4 байт (32 біт)	$0 \div 2^{32} - 1$
long	4 байт (32 біт)	$-2^{31} \div 2^{31} - 1$
unsigned long	4 байт (32 біт)	$0 \div 2^{32} - 1$

Базові дійсні типи даних

Тип	Розмір	Діапазон
float	4 байт (32 біт)	$\pm 3.4 \cdot 10^{-38} \div \pm 3.4 \cdot 10^{+38}$
double	8 байт (64 біт)	$\pm 1.7 \cdot 10^{-308} \div \pm 1.7 \cdot 10^{+308}$

Константи цілого типу за умовчанням мають тип **int**. Для завдання константи типу **long** дописується символ **L|l**, типу **unsigned** – символ **U|u**: **234L**, **456u**, **23456Lu**. Для завдання константи в 8-й системі числення (СЧ) на початку константи дописується **0**, в 16-й СЧ – **0x**: **020**, **0x4a**.

Константи дійсного типу за умовчанням мають тип **double**. Для завдання константи типу **float** дописується символ **F|f**: **23.45E-6f**.

Препроцесорні директиви – це записані в тексті програми інструкції, які починаються з нового рядка зі знаку **#**. Вони виконуються препроцесором до компіляції, доповнюючи вхідний текст програми.

Директива **#include** включає в текст програми зміст зазначеного файлу. Ця директива має дві форми:

```
#include "ім'я_файлу"
#include <ім'я_файлу>
```

Директива **#define** виконує заміну ідентифікаторів константами або будь-якою послідовністю операторів (виразів), що найчастіше використовуються. Синтаксис директиви **#define** такий:

#define ідентифікатор рядок_заміщення

Заміни в тексті можна відмінити директивою **#undef**:

#undef ідентифікатор

Наприклад, перша директива замінить у тексті програми кожний символ **W** на число **80**, а друга директива відмінить цю заміну.

#define W 80

#undef W

Елементарні введення та виведення даних

Форматоване виведення даних на екран здійснюється функцією:

printf(рядок_формату, список_виведення) ;

Список_виведення може містити константи, змінні та будь-які вирази, розділені комами. Рядок_формату записується в подвійних лапках і може містити: послідовність звичайних символів, що виводяться на екран; спеціальні символи, що керують розміщенням даних на екрані (**ESC**–послідовності); специфікатори формату даних.

Специфікатори формату – це наступна послідовність:

%[довжина поля] [.точність] [l або L] специфікатор_типу

де **[довжина поля]** – кількість позицій для виведення; **[.точність]** – кількість цифр дробової частини дійсних чисел, чи кількість символів, що виводяться, для рядків; **l** або **L** – позначка для чисел типу **long, double**.

Для виведення даних використовують такі специфікатори типів:

%c – ціле число у вигляді символу;

%d – знакове ціле число у 10-ій системі числення (СЧ);

%u – беззнакове ціле число у 10-ій СЧ;

%o %x – ціле число типу у 8-ій СЧ або в 16-ій СЧ;

%ld %lu – знакове (беззнакове) ціле число типу **long** у 10-ій СЧ;

%lo %lx – число типу **long** у 8-ій СЧ або в 16-ій СЧ;

%lld %llu – знакове (беззнакове) число типу **long long** у 10-ій СЧ;

%llo %llx – ціле число типу **long long** у 8-ій СЧ або в 16-ій СЧ;

%f – дійсне число типу **float**;

%e – дійсне число типу **float** в експоненціальній формі;

%lf – дійсне число типу **double**;

%le – дійсне число типу **double** в експоненціальній формі;

%s – рядок; **%p** – адреса.

Форматоване введення даних здійснюється функцією:

scanf(рядок_формату, список_введення);

Список_введення може містити тільки адреси змінних, раніш визначених у програмі. Рядок_формату записується в подвійних лапках і може містити тільки специфікатори типів даних.

Функція **scanf()** здійснює введення чисел, символів та рядків з клавіатури до натискання клавіші **<Enter>** або до виявлення першого неприпустимого символу у вхідному потоці.

```
int a; float b; char c; char st[20];
scanf("%d %f %c %s", &a, &b, &c, st);
printf("a=%d, b=%f, c=%c st=%s \n", a, b, c, st);
```

Неформатоване введення символів з клавіатури здійснюється функціями **getchar()**, **getch()**, **getche()**, а неформатоване виведення символів на екран – **putchar()**, **putch()**.

```
char c1, c2, c3;
c1=getchar(); // введення символу з натисканням клавіші <Enter>
c2=getch();   // введення без клав. <Enter> і без виведення на екран
c3=getche();  // введення без клав.<Enter> і з виведенням на екран
putchar(c1);  // виведення символу на екран
putch(c2);    // виведення символу на екран
```

Неформатоване введення рядків виконується функцією **gets()**, а неформатоване виведення рядків – **puts()**.

```
char st1[20];
gets(st1); //введення рядка до натискання клавіші <Enter>
puts(st1); //виведення рядка на екран з переведенням на новий рядок
```

В мові C++ введення даних може здійснюватися за допомогою операції **cin>>**, а виведення даних – за допомогою операції **cout<<**.

```
int a; cout<<"Input a"<<endl; //виведення на екран повідомлення
cin>>a;   // введення з клавіатури значення змінної a
cout<<a;   // виведення на екран введеного значення
```

Маніпулятори визначені в файлі **iomanip**: **dec**, **oct**, **hex** – завдання системи числення; **endl** – переведення на новий рядок; **left**, **right** – вирівнювання; **showpos** – виведення знаку перед числом, **setw(4)** – завдання ширини поля; **setprecision(4)** – завдання точності і т.д.

```
char c1; double d1;
cin>>c1>>d1;
cout<<setw(5)<<c1<< showbase<<oct<<c1<<hex<<c1<<endl;
cout<<"d1="<<setprecision(4)<<d1<<endl;
```

ESC-послідовності (керуючі послідовності) – це послідовності символів, які використовують для запису неграфічних та деяких інших символів в рядках та символьних константах. ESC-послідовність складається з нахиленої риски вліво (зворотній слеш), після якого записується буква або знак. Наприклад: `\n` – новий рядок; `\t` – горизонтальна табуляція; `\'` – одиночні лапки; `\"` – подвійні лапки; `\\` – нахилена вліво риска; `\?` – знак питання.

Умовний оператор `if...else` обирає один з двох варіантів виразів обчислень та має такий синтаксис:

```
if(умова) вираз_1 else вираз_2
```

Якщо **умова** – істина, то виконується **вираз_1**, інакше – **вираз_2**.

Будь-яке значення **умови**, крім **0**, вважається істиною. Наприклад:

```
if(x>y) {max=x; min=y;}  
else {max=y; min=x;};
```

Допустима скорочена форма умовного оператору, в якій відсутня конструкція **else вираз_2**. В цьому разі при хибності перевіряємої умови ніяких дій не виконується.

```
if(x>max) x=max;  
res=x;// виконується завжди
```

Оператор-перемикач `switch` дозволяє обрати один з декількох варіантів обчислень та повідомлень. Синтаксис цього оператора такий:

```
switch(перемикаючий_вираз)  
{case константний_вираз_1: оператори_1; break;  
  case константний_вираз_2: оператори_2; break;  
  . . .  
  default: оператори_n+1;  
}
```

Оператор **switch** порівнює значення **перемикаючого_виразу**, який може приймати тільки цілі значення, з **константними виразами** та передає керування тому оператору, **константний_вираз** якого співпадає зі значенням **перемикаючого_виразу**. Якщо такі оператори відсутні, то виконуються оператори, що позначені міткою **default**. Оператор **break** здійснює примусовий вихід з оператора **switch**. Якщо оператор **break** відсутній, то виконуються не тільки оператори відповідної гілки **case**, а й усі інші гілки після цієї до закінчення оператора **switch**. Оператори **break** та **default** необов'язкові в операторі **switch**. Наприклад:

```
unsigned int a;  
puts("Input value:");
```

```
scanf("%u",&a); // введення з клавіатури значення змінної a
switch(a)
{case 0: puts("це нуль"); break;
 case 1: puts("це одиниця"); break;
 case 2: puts("це двійка"); break;
 case 3: puts("це трійка"); break;
 default: puts("це більше трьох");
}
```

Оператори циклу. В мові C++ є три оператори, за допомогою яких можна організувати цикли: **for**, **while**, **do...while**.

Цикл з оператором **for** частіше використовується у випадку, коли відома точна кількість повторів. Цей оператор має синтаксис:

for(ініціалізація_циклу; умова; список_виразів) тіло_циклу

Ініціалізація_циклу – це послідовність визначень та виразів, розділених комами, що виконуються один раз перед початком виконання циклу. Потім перевіряється **умова**. Якщо значення **умови** не дорівнює **0** (тобто є істиною), то виконується **тіло_циклу**, а потім **список_виразів** та знову перевіряється **умова**. При хибному значенні **умови** цикл завершується. Наприклад:

```
for(int i=0,sum=0,pr=1; i<3; i++) // i=1, 2, 3
{sum+=5; // sum=5, 10, 15
 pr*=(i+1); // pr=1, 2, 6
}
```

Цикл з оператором **while** використовується в тому випадку, коли невідоме точне число повторів. Синтаксис оператора наступний:

while(умова) тіло_циклу

Спочатку перевіряється **умова**, якщо **умова** є істиною, то виконується **тіло_циклу** і знову перевіряється **умова**. При хибному значенні **умови** цикл завершується. Наприклад:

```
int a=0, sum=0;
while(sum<20)
{a+=2; // a=2, 4, 6, 8
 sum+=a; // sum=2, 6, 12, 20
}
```

Цикл з оператором **do...while** використовується в тих випадках, коли **тіло_циклу** повинно бути виконане, принаймні, один раз. Синтаксис оператора наступний:

do тіло_циклу **while**(умова);

Спочатку виконується **тіло_циклу**, потім перевіряється **умова**. Якщо **умова** є істиною, то знову виконується **тіло_циклу**. При хибному значенні **умови** цикл завершується.

Наприклад:

```
int m=0, n=0;
do{m+=5; n++; // m=5, 10, 15 n=1, 2, 3
} while(n<3);
```

Оператори циклу можуть мати оператори **continue**, **break**, **return**. Оператор **continue** пропустить залишок тіла циклу та почне виконання наступної ітерації (повторення). Оператор **break** завершить виконання усього циклу. Оператор **return** завершить виконання функції або всієї програми.

```
int n1=0, n2=0, a, sum=0;
while((n1+n2)<20) //перевірка кількості введених значень змінної a
{scanf("%d",&a); // введення 20 значень змінної a
if((a%2)==0)
{n2++; continue;} // перехід до наступної ітерації, якщо a парне,
if(a==99) break; // закінчення циклу, якщо a=99
if(a==100) return; // закінчення програми, якщо a=100
n1++; sum+=a; } // обчислення суми непарних змінних a
}
```

Усі оператори циклу взаємозамінні. Допускається і часто використовується вкладення будь-яких циклів в будь-які цикли.

Оператор безумовного переходу goto передає керування на будь-який позначений міткою оператор в тілі функції. При використанні оператора **goto** не можна входити усередину умовного оператора, перемикача, циклу. Наприклад:

```
if (n==n_max) goto M1; // перехід на m1 при n==n_max
n++;
M1: printf("n=%d", n);
```

Одновимірні масиви. Масив – це сукупність змінних одного типу, що об'єднані під одним ім'ям. В пам'яті елементи масиву розташовуються послідовно.

Масиви бувають різного класу пам'яті. Масиви, визначені поза функціями, у тому числі поза функцією **main()**, є глобальними (зовнішніми). Масиви, які визначені зі словом **static**, є статичними. Масиви, які визначені усередині функції, є автоматичними.

При визначенні масиву задають його розмір. Значення розміру повинно бути константним цілочисловим виразом.

Три способи завдання розміру масиву :

1) цілою числовою константою

```
int a1[10]; //масив елементів типу int, що складається з 10 елементів
```

2) пойменованою константою - із службовим словом **const**;

```
const int M=4;
```

```
float f1[M];//складається з 4 елементів типу float
```

3) за допомогою препроцесорної директиви **#define**.

```
#define N 20
```

```
char c1[N]; //масив, що містить 20 символів.
```

Доступ до елементів масиву може здійснюватися через індекси, причому перший елемент масиву має індекс **0**. Наприклад:

```
int a[3];
```

```
a[0]=2;
```

```
a[1]=6;
```

```
a[2]=a[0]+a[1]*5; // a[2]=32
```

Показчики – це цілі беззнакові змінні, що містять адреси пам'яті якого-небудь об'єкту програми. Показчик може містити адресу змінної, масиву, структури, іншого показчика і т.п. Розмір показчика – 4 байти: 2 байта – сегмент і 2 байта – зсув в сегменті. Тип показчика – це тип змінної, адреса якої може знаходитися в показчику.

Показчик можна ініціалізувати та йому можна привласнювати: 1) адресу раніш визначеної змінної, 2) значення раніш визначеного іншого показчика, 3) явно задану адресу пам'яті (з перетворенням типу). Наприклад:

```
int a=1;
```

```
int *ptrb, *ptrb; //показчики ptrb, ptrb
```

```
ptrb =&a; // адреса змінної a
```

```
ptrb =ptrb; // значення показчика ptrb
```

```
int *ptrc=(int*)0xB8000000; // явна адреса пам'яті
```

Значення показчика збільшується (при додаванні цілого числа **N**) або зменшується (при відніманні цілого числа **N**) на відповідну кількість змінних того типу, адресу яких вмістить показчик. Значення показчика завжди змінюється на кількість байтів, рівне **N*sizeof(тип_показчика)**.

Різниця показчиків показує, скільки змінних відповідного типу може розміститися між цими показчиками.

Ім'я масиву – це покажчик-константа, значення якого є адреса нульового елемента масиву. Це значення не можна змінити.

Наприклад, якщо визначити **int arr[8]**, то при цьому виділиться пам'ять для **8** елементів масиву, а ім'я **arr** стане покажчиком-константою на цей масив. При визначенні розміру покажчика **arr** визначається розмір усього масиву. Ім'я масиву можна привласнювати іншим покажчикам. З елементами масивів, зв'язаними з покажчиками, можна працювати за допомогою індексів і операції розіменування. Наприклад:

```
int arr[8];
int *ptr=arr;
arr[0]=2; *(arr+1)=3; *(2+arr)=4; 3[arr]=5;
ptr[4]=6; *(ptr+5)=7; *(6+ptr)=8; 7[ptr]=9;
```

Рядок – це послідовність елементів типу **char**, останнім елементом якої є **'\0'**. Ініціалізується рядок послідовністю символів у парних подвійних лапках. Адреса першого символу може використовуватися для ініціалізації масиву типу **char**, у цьому випадку ім'я масиву й адреса першого символу стають синонімами. Наприклад:

```
char str1[5]="name"; char str2[]="name";
char str3[5]={'n','a','m','e','\0'};
```

Адреса першого елемента рядка може використовуватися для ініціалізації покажчика типу **char**, наприклад:

```
char *strptr="name" ;
```

У цьому випадку змінна-покажчик **strptr** (але не покажчик-константа) одержує значення адреси символу **'n'**, а сам рядок розташовується в сегменті даних завантажувального модуля програми.

Масив покажчиків – це масив, елементами якого є покажчики.

Приклад визначення масиву з трьох покажчиків на об'єкти типу **int**:

```
int *mas[3]={&a,&b,&c};
```

Ім'я масиву – **mas**, він містить три покажчика типу **int***. Вираз **(mas+1)** відповідає зсуву у пам'яті на **sizeof(int*)** байтів від початку масиву (тобто на **4** байти).

Масиви покажчиків на рядки частіше використовуються для раціонального розміщення в пам'яті й сортування рядків. Наприклад:

```
char *name[3]={"Іванов", "Петро", "Єгорович"};
```

При такому визначенні в пам'яті виділяється **3*sizeof(char*)=12** байт для покажчиків **name[0]**, **name[1]**, **name[2]**, **7** байт – для рядка **"Іванов"**, **6** – для **"Петро"** і **9** – для **"Єгорович"**.

Багатовимірний масив – це масив масивів, тобто такий масив, елементами якого є масиви. Ім'я багатовимірного масиву є показчик-константа на масив показчиків-констант, елементами якого є показчики-константи з адресами початку кожного з рядків масиву елементів. Елементи багатовимірного масиву розташовуються в пам'яті за рядками, тобто в порядку зростання найправішого індексу. Доступ до елементів двовимірного масиву здійснюється за індексом або за допомогою показчиків. Наприклад, звернення до `arr[i][j]` еквівалентно `*(arr[i]+j)`, `*(*(arr+i)+j)` або `((*arr+i))[j]`.

Ініціалізація масивів – це привласнення початкових значень елементам масивів при їх визначенні. Ініціалізація масивів, може здійснюватися за умовчанням та явно.

За умовчанням глобальні і статичні масиви ініціалізуються нулями, автоматичні – довільними даними. Явна ініціалізація буває повна (з завданням кількості елементів і початкових значень усіх елементів масиву) та неповна (з явним завданням кількості елементів, але початкові значення задаються не для всіх елементів масиву):

```
int a[5]={2,5,7,9,1}; // явна повна ініціалізація
int b[7]={5,7,9};      // явна неповна ініціалізація
```

У останньому рядку інші чотири елементи масиву `b` будуть мати або нульові значення (для зовнішнього і статичного масивів), або їхні значення будуть невизначені (для автоматичного масиву).

При повній явній ініціалізації одновимірного масиву можливо не задавати розмір масиву. Компілятор визначає кількість елементів масиву за списком ініціалізації:

```
int c[]={5,7,9,8}; // Масив із чотирьох елементів
```

Нижче наведені приклади ініціалізації двовимірних масивів. При явній ініціалізації багатовимірних масивів можливо вводити додаткові скобки.

```
int m[3][2]={2,4,6,8,1,3}; // явна повна ініціалізація
int n[2][3]={1,9,3,2,8,4}; // явна повна ініціалізація
int d[4][3]={1}, {2}, {3}, {4}; // явна неповна ініціалізація
```

Багатовимірні масиви можна ініціалізувати без вказівки найлівої розмірності. Компілятор визначає кількість елементів за списком ініціалізації:

```
int p[][3]={3,2,1}, {3,8,9}, {2,7,4};
float f[][2]={1.2}, {2.2,3.4}, {2.5,6.4}, {2.0};
```

В цьому прикладі визначені масиви `int p[3][3]` і `float f[4][2]`.

Масиви динамічної пам'яті. Для даних масиву динамічної пам'яті виділяється динамічна пам'ять під час виконання програми в так званій "купі", розмір і положення якої залежать від моделі пам'яті. При цьому розмір масиву можна визначити як змінну. Це зручно у випадку, коли невідомо, скільки об'єктів буде створено при роботі програми.

В мо в Сі використовується функція **malloc()** для запиту і виділення пам'яті, кількість якої задається в байтах, і функція **free()** для її звільнення. Прототипи цих функцій описані в заголовному файлі **<stdlib.h>**.

Наприклад:

```
int n;
scanf("%d",&n); //введення розміру масиву (розмір – змінна)
int *ptri=(int*)malloc(n*sizeof(int)); //запит пам'яті
if(ptri==NULL) //перевірка виділення пам'яті
{puts("Error"); return -1;}
...
free(ptri); // звільнення пам'яті
```

У мо в С++ для цієї мети існують операції **new** і **delete**.

Операція **new** автоматично визначає необхідний обсяг пам'яті в байтах.

```
double *ptrd =new double[n]; //запит пам'яті
if(ptrd==NULL) //перевірка виділення пам'яті
{puts("Error"); return -1;}
...
delete[] ptrd; // звільнення пам'яті
```

Звернення до елементів динамічного масиву може здійснюватися через індекси або за допомогою операції розіменування.

Наприклад:

```
for(i=0;i<n;i++)
    ptri[i]=10*i;

for(i=0;i<n;i++)
    printf("%lf ",*(ptrd+i));
```

ЛАБОРАТОРНА РОБОТА №1

Базові типи змінних. Елементарне введення-виведення даних

Мета роботи – ознайомитися з основними типами даних, які використовуються в мові Cі, засвоїти елементарні функції введення-виведення.

Програма 1_1. Програма визначає код символу за кодовою таблицею **ASCII** та визначає розмір типу даних у байтах.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{puts("My first program"); //виведення рядка
 char s='a'; //визначення і ініціалізація символу
 printf("Symbol %c=%d(10S)=%#o(8S)",s,s,s);
 printf("=%#x(16S) size char=%d bait\n",s,sizeof(s));
 int a=78; //визначення і ініціалізація цілої змінної
 printf("a=%d size int=%d bait\n",a,sizeof(a));
 float b=526.2345; //визначення і ініціалізація дійсної змінної
 printf("b=%f=%e size float=%d bait\n",b,b,sizeof(b));
 system("pause");
 return 0;
}
```

Завдання до програми 1_1. Доповніть програму: виведіть на екран рядок з Вашим прізвищем, виведіть замість букви 'a' першу букву Вашого імені з кодами; визначте нові цілі змінні типу **short**, **long**, **long long** та дійсну змінну типу **double** з будь-якими значеннями та виведіть на екран їх значення та розмір в байтах.

Програма 1_2. Програма виконує прості арифметичні дії.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{int a1=9, a2=2, b1, b2, b3;
 a2+=a1; // a2=2+9=11
 printf("a1=%d a2=%d\n",a1,a2);
 b1=a2/4+a2%4; // b1=11/4+11%4=2+3=5
 b2=b1>a1&&b1>a2; // b2=(5>9) та (5>11) ? ні 0
 printf("b1=%d b2=%d\n", b1,b2);
 b3=-b2; // b2=b2-1=0-1=-1 b3=b2=-1
 printf("b2=%d b3=%d\n", b2,b3);
}
```

```
float f1=3.6, f2=0.5, r1, r2;
r1=f1/f2;    // r1=3.6/0.5=7.2
r2=(f1<f2)?f1+2:--f2;    // r2=(3.6<0.5)? ні (0.5-1)=-0.5
printf("r1=%5.2f r2=%5.2f\n",r1,r2);
system("pause");
return 0;
}
```

Завдання до програми 1_2. Розробіть програму, що обчислює задані вирази. Результати пояснить за допомогою коментарів.

1. $a1=3$; $a2=8$; $a2-=a1$; $b1=a1+2*a2^2$; $b2=!a1$; $b3=b2++$;
 $f1=1.5$; $f2=1.5$; $r1=f1^2+2*f2$; $r2=(f1!=f2)?f1-0.3:f2+0.4$;
2. $a1=2$; $a2=3$; $a2*=10$; $b1=a1^3+a2$; $b2=a1>a2$; $b3=++b2$;
 $f1=1.5$; $f2=1.5$; $r1=2*f1-f2$; $r2=(f1==f2)?f1*3:f2*5$;
3. $a1=6$; $a2=20$; $a2/=5$; $b1=5*a1+a2^2$; $b2=a1>a2$; $b3=-b2$;
 $f1=1.5$; $f2=3.5$; $r1=f1+4*f2$; $r2=(f1>=f2)?f1-2.5:f2+5.2$;
4. $a1=3$; $a2=2$; $a2+=a1+5$; $b1=a1^3+3*a2$; $b2=a1<a2$; $b3=b2--$;
 $f1=1.5$; $f2=2.5$; $r1=f1/f2*2$; $r2=(f1<=f2)?f1-0.2:f2+0.7$;
5. $a1=5$; $a2=13$; $a2-=a1+5$; $b1=a1+a2^2$; $b2=a1>a2$; $b3=b2++$;
 $f1=4.5$; $f2=1.8$; $r1=f1+f2/2$; $r2=(f1>f2)?f1+1.2:f2-1.4$;

Програма 1_3. Програма за введеними з клавіатури значеннями радіусу та висоти обчислює та виводить на екран об'єм циліндра.

```
#include <stdio.h>
#include <iostream>
#include <stdlib.h>
int main()
{float h,r,V;
 puts("Input r, h");
 scanf("%f %f", &r,&h);
 V=3.141592f*r*r*h;
 printf("r =%6.2f cm, h =%6.2f cm\n",r,h);
 printf("V =%8.3f cm^3\n",V);
 system("pause");
 return 0;
}
```

Завдання до програми 1_3. Розробіть програму, що виконує задані дії та виводить результат в заданому форматі. Перша цифра в специфікаторі формату – загальна кількість позицій для виведення даних, друга – кількість цифр дробової частини числа.

1. Введіть з клавіатури змінні **x, z** (типу **float**), обчисліть вираз $\text{res} = (4.2 * x - z)^2 / 3.5$ та виведіть на екран результат з трьома цифрами цілої частини і двома – дробової.
2. Введіть з клавіатури змінні **a, b, c** (типу **double**), обчисліть вираз $\text{res} = a + 2.8 * b - c^2$ та виведіть на екран результат з двома цифрами цілої частини і трьома – дробової.
3. Введіть з клавіатури змінні **k, g** (типу **double**), обчисліть вираз $\text{res} = k^2 + (g - k) / 2.3$ та виведіть на екран результат з чотирма цифрами цілої частини і трьома – дробової.
4. Введіть з клавіатури змінні **v, s, f** (типу **float**), обчисліть вираз $\text{res} = s^3 / v - f * v$ та виведіть на екран результат з двома цифрами цілої частини і чотирма – дробової.
5. Введіть з клавіатури змінні **x, z, h** (типу **double**), обчисліть вираз $\text{res} = x^3 + z / h^2$ та виведіть на екран результат з трьома цифрами цілої частини і трьома – дробової.

Програма 1_4. Програма за введенням з клавіатури значенням радіуса виводить на екран значення об'єму кулі. Значення радіуса **R** вводиться за допомогою функції **cin>>**, а об'єм виводиться на екран за допомогою функції **cout<<**. Маніпулятор **setw(10)** задає загальну ширини поля для виведення, **fixed** – виведення в фіксованому форматі, **setprecision(4)** – кількість цифр дробової частини дійсного числа.

```
#include <stdlib.h>
#include <math.h>
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{double R, V;
  cout<<"Input radius\n";// введіть радіус
  cin>>R;
  V=3.0/4.0*M_PI*R*R*R;
  cout<<fixed<<setprecision(4);
  cout<<"R = "<<R<<" cm"<<endl;
  cout<<"V = "<<setw(10)<<V<<" cm^3\n";
  system("pause");
  return 0;
}
```

Завдання до програми 1_4. Розробіть програму, в якій змінні введіть за допомогою операції **cin>>**, виконайте задані обчислення і виведіть введені змінні та результат за допомогою операції **cout<<**.

1. Обчисліть площу ромбу $S=d_1*d_2$ (d_1, d_2 – діагоналі ромбу).
2. Обчисліть площу куба $S=6*a^2$.
3. Обчисліть довжину кола $L=2\pi R$.
4. Обчисліть площу прямокутника $S=a*b$.
5. Обчисліть площу кола $S=\pi R^2$.

Програма 1_5*. Програма показує, як здійснюється перехід між максимальним та мінімальним значеннями для різних типів даних при виконанні арифметичних дій над ними.

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
using namespace std;
int main()
{short a1=32766, a2, a3;      // Цілі зі знаком
 char b1=126, b2, b3;      // Цілі зі знаком
 int c1=2147483646, c2, c3;  // Цілі зі знаком
 a2=a1+1; a3=a1+2;
 printf("short value: %d %d %d\n", a1, a2, a3);
 b2=b1+1; b3=b1+2;
 printf("char value: %d %d %d\n", b1, b2, b3);
 c2=c1+1; c3=c1+2;
 printf("int value: %d %d %d\n\n", c1, c2, c3);
 system("pause");
 return 0;
}
```

Завдання до програми 1_5*

1. Визначте та проініціалізуйте змінні **x1, y1, z1, v1**:

```
short x1=32763;    unsigned short y1=65531;
char z1=122;      unsigned char v1=2;
```

Додайте до значень змінних числа, щоб нові значення змінних стали максимальними (**x2, y2, z2, v2**) та більшими за максимальне значення кожного з типів на **1** (**x3, y3, z3, v3**). Результати виведіть на екран.

2. Визначте та проініціалізуйте змінні **x1, y1, z1, v1**:

```
short x1=-32763;   unsigned short y1=31;
char z1=-120;     unsigned char v1=20;
```

Відніміть від значень змінних числа, щоб нові значення стали мінімальними (**x2, y2, z2, v2**) та меншими за мінімальне значення кожного з типів на **1** (**x3, y3, z3, v3**). Результати виведіть на екран.

3. Визначте та проініціалізуйте змінні **x1, y1, z1, v1**:

```
short x1=30;      unsigned short y1=35000;  
char z1=-25;     unsigned char v1=20;
```

Додайте до значень змінних числа, щоб нові значення змінних стали максимальними (**x2, y2, z2, v2**) та більшими за максимальне значення кожного з типів на **1** (**x3, y3, z3, v3**). Результати виведіть на екран.

4. Визначте змінні **x1, y1, z1, v1**:

```
short x1=-32000;  unsigned short y1=65530;  
char z1=-108;    unsigned char v1=240;
```

Відніміть від значень змінних числа, щоб нові значення стали мінімальними (**x2, y2, z2, v2**) та меншими за мінімальне значення кожного з типів на **1** (**x3, y3, z3, v3**). Результати виведіть на екран.

5. Визначте змінні та проініціалізуйте **x1, y1, z1, v1**:

```
int x1=2147483645;  unsigned int y1=4294967290;  
char z1=125;       unsigned char v1=252;
```

Додайте до значень змінних числа, щоб нові значення змінних стали максимальними (**x2, y2, z2, v2**) та більшими за максимальне значення кожного з типів на **1** (**x3, y3, z3, v3**). Результати виведіть на екран.

Програма 1_6*. Програма виводить на екран задані рядки.

```
#include <stdio.h>  
#include <stdlib.h>  
#include <iostream>  
#define STR1 "Today we are"  
#define STR2 "learning how to "  
#define STR3 "\"Programm on C++\""  
int main()  
{printf("|%s|\n", STR1);  
  printf("|%s|\n", STR2);  
  printf("|%s|\n", STR3);  
  printf("|%75.5s|\n", STR1);  
  printf("|%-75.8s|\n", STR2);  
  printf("|%75.9s|\n", STR3);  
  printf("|%-75.9s|\n", STR3);  
  system("pause");  
  return 0;  
}
```

Завдання до програми 1_6*. Додайте ще рядок **STR4** з двома заданими словами за допомогою директиви **#define**. Доповніть програму, щоб на екран виводились ці слова повністю (з межами на початку та кінці рядка екрана), потім на новому рядку тільки перше слово спочатку рядка екрана (також з межами на початку та кінці екрана) і на наступному рядку перше слово на кінці рядка екрана.

1. Ім'я та по батькові студента.
2. Прізвище та ім'я студента.
3. Назва предмету та прізвище викладача.
4. Прізвище автора та назва книги.
5. Назва міста та вулиці.

Контрольні питання до лабораторної роботи № 1

1. Які типи даних використовуються в мові C++? Скільки байт займає кожен із типів даних у мові Cі?
2. Для чого використовується операція **sizeof()**? Приведіть приклад.
3. Який знак є трьохарним? Як виконується? Приведіть приклад.
4. Який загальний склад аргументів функції **printf()**? Приведіть приклад використання.
5. Який загальний склад аргументів функції **scanf()**? Приведіть приклад використання.
6. Як ввести та вивести чисельні, символьні змінні з використанням операцій **cin>>**, **cout<<**? Приведіть приклад.
7. Знайдіть помилки та запишіть правильний код:

```
#include studio.h
void main;
{int i
  i:=52
  print('In year i weeks')
}
```

ЛАБОРАТОРНА РОБОТА №2

Оператори вибору

Мета роботи – ознайомитися з умовними операторами **if...else**, операторами-перемикачами, навчитися будувати найпростіші програми, які розгалужуються.

Програма 2_1. Програма визначає, чи є введене число парним.

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
int main()
{int n;
 puts("Input number:");
 scanf("%d",&n);
 if(n%2==0) printf("number %d even \n", n);
 else printf("number %d odd\n\n", n);
 system("pause");
 return 0;}
```

Завдання до програми 2_1. Розробіть програму, яка виконує задані обчислення та виводить на екран відповідні повідомлення.

1. Перевірте, чи більше **100** введене з клавіатури ціле число.
2. Перевірте, чи менше **20** введене з клавіатури ціле число.
3. Перевірте, чи закінчується на **3** введене з клавіатури ціле число.
4. Перевірте, чи закінчується на **0** введене з клавіатури ціле число.
5. Перевірте, чи ділиться на **5** введене з клавіатури ціле число.

Програма 2_2. Програма визначає, чи є введене число додатним та більшим чи меншим заданого числа.

```
#include <stdlib.h>
#include <iostream>
using namespace std;
int main()
{float x;
 cout<<"Input value:\n";
 cin>>x;
 if(x>0)
     if(x>100)
         cout<<x<<" >0 and > 100\n";
     else cout<<x<<" >0 and <= 100\n";
 else
```

```

if(x<-100)
    cout<<x<<" <0 and < -100\n";
else cout<<x<<" <0 and >= -100\n";
system("pause");
return 0;}

```

Завдання до програми 2_2. Розробіть програму, яка виконує задані обчислення, використовуючи вкладений умовний оператор, та виводить на екран чотири відповідні повідомлення.

1. Введіть з клавіатури ціле число, якщо воно парне, перевірте чи є додатним, якщо – непарне, чи більше **25**.
2. Введіть з клавіатури два числа, для більшого з них числа перевірте, чи знаходиться воно в діапазоні від **5** до **30**.
3. Введіть з клавіатури ціле число, якщо воно більше **50**, перевірте, чи кратне воно **4**, інакше – чи кратне воно **3**.
4. Введіть з клавіатури два цілих числа, для меншого з них числа перевірте, чи є воно непарним.
5. Введіть з клавіатури ціле число, якщо воно додатне, перевірте, чи кратне **5**, інакше – чи менше воно за **-100**.

Програма 2_3. Програма в залежності від введеного знаку визначає результат виконання дії за допомогою оператора **switch()** із застосуванням оператору **break**.

```

#include <stdio.h>
#include <stdlib.h>
#include <iostream>
int main()
{float a,b; char s;
 puts("Input value a, b:");
 scanf("%f %f",&a,&b);
 fflush(stdin);
 puts("Input sing (-, +, *, /):");
 s=getchar();
 switch(s)
 {case '-': printf("%.2f-%.2f=%.2f\n",a,b,a-b); break;
 case '+': printf("%.2f+%.2f=%.2f\n",a,b,a+b); break;
 case '*': printf("%.2f*%.2f=%.2f\n",a,b,a*b); break;
 case '/': printf("%.2f/%.2f=%.2f\n",a,b,a/b); break;
 default: printf("Error! sings (-, +, *, /)\n");
 }
 system("pause"); return 0;}

```

Завдання до програми 2_3. Розробіть програму з використанням оператора **switch()**, яка згідно з введеною з клавіатури першої букви (великої або малої) виводить на екран задану повну назви або повідомлення про помилку, якщо введена буква не співпадає з заданими.

1. Назви **5** країн Азії.
2. Назви **6** річок.
3. Назви **7** столиць Європи.
4. Назви **5** квітів.
5. Назви **6** дерев.

Програма 2.4*. Програма в залежності від введеної цифри виводить назву дня тижня.

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
using namespace std;
int main()
{int a ;
 cout<<"Input number from 1 to 7, you know day of week:\n";
 cin>>a;
 switch(a)
 {case 1: cout<<"Monday\n";      break;
  case 2: cout<<"Tuesday\n";    break;
  case 3: cout<<"Wednesday\n";  break;
  case 4: cout<<"Thursday\n";   break;
  case 5: cout<<"Friday\n";     break;
  case 6: cout<<"Saturday\n";   break;
  case 7: cout<<"Sunday\n";     break;
  default: cout<<"Error! Value from 1 to 7\n");
 }
 system("pause"); return 0;}
```

Завдання до програми 2_4*. Розробіть програму, в якій вводиться з клавіатури ціле число та на екран виводиться відповідне повідомлення або повідомлення про помилку, якщо введене число не співпадає з заданими (кожне з повідомлень може бути позначене кількома мітками).

1. Введіть номер місяця та виведіть одне з повідомлень: "**Осінній семестр**", "**Весняний семестр**", "**Канікули**", чи "**Error**".
2. Введіть номер дня тижня та виведіть кількість учбових пар цього дня тижня: "**Дві пари**", "**Три пари**", "**Вільний день**", чи "**Error**".

3. Введіть номер місяця та виведіть одне з повідомлень: "Весна", "Літо", "Осінь", "Зима" чи "Error, Value from 1 to 12".
4. Введіть номер дня тижня та виведіть одне з повідомлень: "Робочий день", "Субота", "Неділя" чи "Error".
5. Введіть номер місяця та виведіть номер кварталу: "1-й квартал", "2-й квартал", "3-й квартал", "4-й квартал", чи "Error" .

Програма 2_5*. Програма визначає максимальне число серед трьох введених з клавіатури.

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
using namespace std;
int main()
{int a, b, c, max;
  cout<<"Input value a, b, c:\n";
  cin>>a>>b>>c;
  if(a>b && a>c) max=a;
  else if(b>c) max=b;
    else max=c;
  cout<<"max="<<max<<endl;
  system("pause");
  return 0;
}
```

Завдання до програми 2_5*. Розробіть програму, яка виконує задані обчислення та виводить на екран відповідні повідомлення.

1. Введіть з клавіатури чотири дійсних числа, визначте найменше з них, перевірте додатне воно чи від'ємне.
2. Введіть з клавіатури два цілих числа та для більшого числа визначте, у якому з наведених діапазонів знаходиться це число: **менше 0, від 0 до 50, від 51 до 100, більше 100.**
3. Введіть з клавіатури три цілих числа, визначте найменше з них та перевірте парне воно чи непарне.
4. Введіть з клавіатури два цілих числа та для меншого числа визначте, у якому з наведених діапазонів знаходиться це число: **менше -100, від -99 до 0, від 1 до 100, більше 100.**
5. Введіть з клавіатури чотири дійсних числа, визначте найбільше з них, перевірте чи більше це число за **500** чи не більше.

Контрольні питання до лабораторної роботи № 2

1. Які оператори вибору Ви знаєте? Для чого вони призначені?
2. Яке значення може прийняти вираз умови в операторі **if (умова)**? Що є істиною, а що хибністю?
3. Дайте визначення вкладеного оператору умови. Наведіть приклад.
4. Які повідомлення виведуться в фрагменті програми? Поясніть.


```
int x=5,y=6,z=10;
if(x==3) puts("AA1");
else puts("BB2");
if(x=10, x<y) puts("CC3");
else puts("DD4");
if(z<y, z=-4) puts("EE5");
else puts("FF6");
```
5. Визначте результати роботи логічної конструкції для змінних **x** і **y** з усіма можливими варіантами значень **0** або **1** (чотири випадки). Що зміниться, якщо рядок, помічений міткою **m1**, взяти в фігурні дужки? Наведіть блок-схеми алгоритмів обох варіантів.


```
int x,y;
cin>>x>>y;
if(x)
m1: if(y) cout<<"M0";
else cout<<"T0";
cout<<"PE"<<endl;
```
6. Які значення може приймати **вираз** в операторі **switch (вираз)**?
7. Яке призначення мітки **default** у операторі множинного вибору? Чи завжди вона потрібна? Що станеться при її вилученні?
8. Яке призначення оператора **break** у операторі множинного вибору? Що станеться при його вилученні? У якому випадку він непотрібний?

Примітка. В звіті до лабораторної роботи №2 до програми **2_2m.cpp** наведіть блок-схеми алгоритму.

ЛАБОРАТОРНА РОБОТА №3

Оператори циклу

Мета роботи – ознайомитися з організацією циклічного обчислювального процесу із застосуванням операторів **while()**, **do...while()**, **for()**, навчитися здійснювати контроль вхідних даних при наявності обмежень на їхні значення.

Програма 3_1. Програма обчислює суму ряду цілих парних чисел, доки сума не стане більше **20**, використовуючи оператор **while**.

```
#include <stdlib.h>
#include <iostream>
using namespace std;
int main()
{int a=0, sum=0, n=0;
while(sum<20)
    {n+=1;      // Кількість циклів
      a+=2;      // Парне число
      sum+=a;    // Сума парних чисел
      cout<<"n="<<n<<"  a="<<a<<"  sum="<<sum<<endl; }
cout<<"Loops n="<< n<<endl;
  system("pause");
  return 0;
}
```

Завдання до програми 3_1. Розробіть програму, що виконує задані обчислення та виводить на екран результат обчислення, останній член та кількість циклів.

1. Обчисліть суму ряду цілих чисел, кратних **3**, доки останнє число не стане більше **21**.
2. Обчисліть суму цілих чисел, що вводяться з клавіатури, доки сума не стане більше **100**.
3. Обчисліть суму ряду цілих чисел, кратних **4**, доки останнє число не стане більше **36**.
4. Обчисліть суму цілих чисел, що вводяться з клавіатури, доки сума не стане більше **50**.
5. Обчисліть суму ряду цілих чисел, кратних **5**. Кінцева кількість циклів вводиться з клавіатури.

Програма 3_2. Програма обчислює квадрати послідовності чисел.

```
#include <stdlib.h>
#include <iostream>
using namespace std;
int main()
{int i, b, a=0,kol;
  cout<<"Input number of cycles\n";
  cin>>kol; // Введення кількості циклів
  for(i=0; i<kol; i++)
    {if(a>500){cout<<"a>500\n"; // Завершення циклу при a>500
      break;}
    a+=1; b=a*a;
    cout<<"a="<<a<<" b="<<b<<endl;    }
  system("pause"); return 0;}
```

Завдання до програми 3_2. Розробіть програму, що виконує задані обчислення та виводить на екран результати (оператор **for**). Загальна кількість чисел та числа вводяться з клавіатури. При введенні заданого числа цикл завершується (оператор **break**).

1. Обчисліть суму та кількість тільки цілих додатних непарних чисел. При введенні числа **-11** цикл завершується.
2. Обчисліть суму та кількість тільки цілих додатних парних чисел. При введенні числа **55** цикл завершується.
3. Обчисліть суму та кількість тільки дійсних чисел, що більше **5**. При введенні числа, що більше **100** цикл завершується.
4. Обчисліть суму та кількість тільки дійсних чисел, що менше **-5**. При введенні числа менше **-99** цикл завершується.
5. Обчисліть суму та кількість тільки цілих від'ємних непарних чисел. При введенні числа **13** цикл завершується.

Програма 3_3. Програма обчислює та виводить на екран таблицю множення, використовуючи вкладений оператор циклу.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{int i,j;
  for(i=1; i<=10; i++)
    {for(j=1; j<=10; j++)
      printf("%d * %d = %2d\n",i,j,i*j);
      printf("\n"); system("pause");    }
  system("pause"); return 0;}
```

Завдання до програми 3_3. Обчисліть та виведіть на екран значення **a**, **b** та функції **F**, використовуючи вкладений оператор циклу, якщо **a**, **b** змінюються у заданих діапазонах з заданим кроком.

1. $F=a^2+b^2$; діапазон зміни **a** від **4** до **20** з кроком **2**;
діапазон зміни **b** від **3** до **21** з кроком **3**.
2. $F=a^2-4b$; діапазон зміни **a** від **2** до **10** з кроком **1**;
діапазон зміни **b** від **5** до **20** з кроком **5**.
3. $F=a^2-b^2$; діапазон зміни **a** від **0** до **40** з кроком **4**;
діапазон зміни **b** від **6** до **20** з кроком **2**.
4. $F=2a^2+b$; діапазон зміни **a** від **3** до **30** з кроком **3**;
діапазон зміни **b** від **8** до **20** з кроком **4**.
5. $F=5a+b^2$; діапазон зміни **a** від **5** до **50** з кроком **5**;
діапазон зміни **b** від **4** до **10** з кроком **2**.

Програма 3_4*. Програма обчислює суму та добуток десяти непарних значень введених з клавіатури чисел. При введенні числа **0** цикл завершується (**break**).

```
#include <stdlib.h>
#include <iostream>
using namespace std;
int main()
{int n=0, n1=0, n2=0, a, sum=0, pr=1;
  cout<<"Input value:\n";
  while(n1<10)
  {cin>>a;    // введення значень змінної a
   if(a==0) break;    // закінчення циклу при a=0
   n++;              // підрахунок кількості введених значень a
   if(!(a%2)){n2++; // підрахунок кількості парних значень a
   continue;} //закінчення ітерації
   n1++;           // підрахунок кількості непарних значень a
   sum+=a; pr*=a; // підрахунок суми та добутку непарних значень a
   cout<<"Number of odd values = "<<n1<<endl;
   cout<<"sum of odd values = "<<sum<<endl;
   cout<<"Work odd values = "<<pr<<endl; }
  cout<<"Number of input values = "<<n<<endl;
  cout<<"Number of even values = "<<n2<<endl;
  system("pause");
  return 0;}
```

Завдання до програми 3_4*. Розробіть програму, що виконує задані обчислення з числами, які вводяться з клавіатури, та виводить на екран результати обчислень.

1. Знайдіть найбільше число з введених **10** цілих чисел та обчисліть суму, кількість чисел, що більші **5**. При введенні числа **10** цикл завершується (оператор **break**).
2. Знайдіть найменше число з введених **10** дійсних чисел та обчисліть суму, кількість чисел, які більші **2.5**. При введенні числа **-1** цикл завершується (оператор **break**).
3. Підрахуйте кількість серед введених **15** цілих чисел, що знаходяться в межах від **48** до **122**, виведіть ці числа в символьному вигляді та обчисліть їх суму. При введенні числа **0** цикл завершується (оператор **break**).
4. Знайдіть найбільше за абсолютним значенням серед введених **6** чисел типу **long** та обчисліть суму чисел, що менші за абсолютним значенням числа **3000**, підрахуйте їх кількість. При введенні числа **999** цикл завершується (оператор **break**).
5. Обчисліть суму введених **15** дійсних чисел та суму і кількість тільки від'ємних чисел. При введенні числа **-100** цикл завершується (оператор **break**).

Програма 3_5*. В програмі використовується оператор **do...while()** для контролю вхідних даних, при введенні числа x в заданому діапазоні обчислюються функції **acos(x)**, **asin(x)**.

```
#include <stdlib.h>
#include <iostream>
#include <math.h>
using namespace std;
int main()
{double x; int kol=0;
 do{cout<<"Enter value -1<=x<=1\n";
   cin>>x;
   kol++; // підрахунок кількості введених значень
 } while(x<-1.0||x>1.0);
 cout<<"asin("<<x<<" ) = "<<asin(x)<<endl;
 cout<<"acos("<<x<<" ) = "<<acos(x)<<endl;
 cout<<"Input "<<kol<<" value\n";
 system("pause"); return 0;}
```

Завдання до програми 3_5*. Розробіть програму, що виконує обчислення з числами, які вводяться з клавіатури (оператор **for**) і знаходяться в заданому діапазоні (оператор **do...while**), та виводить на екран результати обчислень і кількість введених та обчислених чисел. Загальна кількість чисел, що вводяться, не повинна перевищувати **12**, інакше програма завершується (оператор **return -1;**).

1. Обчисліть синус та косинус (**sin(x)**, **cos(x)**) тих **п'яти** дійсних чисел, що знаходяться в діапазоні від **0** до **2 π** .
2. Обчисліть секанс та косеканс (**1/sin(x)**, **1/cos(x)**) тих **шести** дійсних чисел, що знаходяться в діапазоні від **$\pi/20$** до **$\pi/2$** .
3. Обчисліть гіперболічні синус та косинус (**sinh(x)**, **cosh(x)**) тих **п'яти** дійсних чисел, що знаходяться в діапазоні від **-3.0 π** до **3.0 π** .
4. Обчисліть тангенс та котангенс (**tan(x)**, **1/tan(x)**) тих **шести** дійсних чисел, що знаходяться в діапазоні від **$\pi/20$** до **$\pi/2$** .
5. Обчисліть арктангенс та арккотангенс (**atan(x)**, **atan(1/x)**) тих **п'яти** дійсних чисел, що знаходяться в діапазоні від **1.0 π** до **20.0 π** .

Контрольні питання до лабораторної роботи № 3

1. Які оператори циклу Ви знаєте? Чим вони відрізняються?
2. Поясніть призначення операторів **continue**, **break**, **return** та наведіть приклади їх використання в операторах циклу.
3. Замініть оператор **for()** у фрагменті програми на оператори **while()**, **do...while()**.

```
int i,a=2,res;
for(i=0;i<3;i++)
{res=i*i-a*a; a+=2; printf ("%d\n",res);}
```

4. Як виконується оператор **for** із роздільником **кома**? Який буде результат роботи наступного фрагменту програми? Поясніть.

```
int a,b,c;
for(a=0,b=0, c=5; c>10,a<3; a++,b+=c)
{c+=3; printf("a=%d b=%d c=%d\n",a,b,c);}
printf("Результат a=%d b=%d c=%d\n",a,b,c);
```

5. Поясніть результати роботи кожного з фрагментів програм.

short n=0;	short m=0;	short k=0;
cout<<"Loop: 0 ";	cout<<"Loop: 0 ";	cout<<"Loop: 0 ";
while(n++,n<5)	while(m<5,m++)	while(k<5,++k)
cout<<n<<" ";	cout<<m<<" ";	cout<<k<<" ";
cout<<" N = "<<n;	cout<<" M = "<<m;	cout<<" K = "<<k;

ЛАБОРАТОРНА РОБОТА №4

Масиви і покажчики

Мета роботи – ознайомитися з формами визначення одновимірних масивів у мові C++, використанням покажчиків при роботі з масивами, навчитися формувати числові і символьні масиви.

Програма 4_1. У програмі визначається та ініціалізується одновимірний масив. Визначається мінімальний елемент масиву.

```
#include <stdlib.h>
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{int arr[10]={4,2,-3,8,12,1,-2,9,0,15}, min, i;
  for(i=0; i<10; i++)
    {cout<<"arr["<<i<<"]="<<setw(3)<<arr[i];
      cout<<" "<<&arr[i]<<endl; }
  cout<<"\nSize massiv arr: "<<sizeof(arr)<<" bait\n";
  min=arr[0];
  for(i=0; i<10; i++)
    if(arr[i]<min) min=arr[i];
  cout<<"min ="<<min<<endl;
  system("pause");
  return 0;}
```

Завдання до програми 4_1. Розробіть програму, в якій з клавіатури вводиться масив цілих чисел заданого розміру, виконуються задані обчислення. На екран виводяться значення елементів масиву і результати обчислень.

1. Обчисліть середнє арифметичне тільки ненульових елементів. Розмір масиву **6**.
2. Обчисліть середнє арифметичне елементів без обліку мінімального елемента масиву. Розмір масиву **8**.
3. Обчисліть середнє арифметичне тільки додатних елементів. Розмір масиву **5**.
4. Обчисліть середнє арифметичне тільки парних елементів. Розмір масиву **6**.
5. Обчисліть суму квадратів від'ємних елементів. Розмір масиву **7**.

Програма 4_2. У програмі визначається і заповнюється випадковими числами одновимірний масив та підраховується кількість додатних чисел між мінімальним і максимальним елементами масиву.

```
#include <iostream>
using namespace std;
#include <time.h> //для запуску генератора випадкових чисел
#include <stdlib.h> //для підключення генератора випадкових чисел
int main()
{const int n=20;
 int a[n],i,imax=0,imin=0,kol=0;
 srand((unsigned)time(NULL)); //запуск генератора випадк. чисел
 for(i=0; i<n; i++)
 {a[i]=rand()%151-50; // значення від -50 до +100
 cout<<" a["<<i<<"]="<<a[i]<<"\n";
 }
 for(i=0; i<n; i++) // знаходження max та min значення
 {if(a[i]>a[imax])imax=i;
 if(a[i]<a[imin])imin=i;
 }
 cout<<" max a["<<imax<<"]="<<a[imax];
 cout<<"\n min a["<<imin<<"]="<<a[imin];
 int ibeg=imax<imin? imax : imin;
 int iend=imax<imin? imin : imax;
 cout<<"\n\nBetween a["<<ibeg<<"] and a["<<iend<<"]\n";
 for(i=ibeg+1; i<iend; i++)
 if(a[i]>0)
 {cout<<" a["<<i<<"]="<<a[i]<<"\n";
 kol++;}
 cout<<" positive value "<<kol<<endl;
 system("pause");
 return 0;}
```

Завдання до програми 4_2. Розробіть програму, яка виконує задані обчислення та виводить на екран масив і результат обчислення.

1. Заповніть масив цілих чисел випадковими числами від **-40** до **40**, обчисліть кількість від'ємних чисел та виведіть їх на екран. Розмір масиву **15**.
2. Заповніть масив цілих чисел випадковими числами від **0** до **60**, обчисліть кількість чисел, кратних **3**, та виведіть їх на екран. Розмір масиву **12**.

3. Заповніть масив цілих чисел випадковими числами від **-20** до **20**, обчисліть кількість парних чисел та виведіть їх на екран. Розмір масиву **16**.
4. Заповніть масив цілих чисел випадковими числами від **0** до **100**, обчисліть кількість чисел, кратних **5**, та виведіть їх на екран. Розмір масиву **15**.
5. Заповніть масив цілих чисел випадковими числами від **-30** до **30**, обчисліть кількість непарних чисел та виведіть їх на екран. Розмір масиву **14**.

Програма 4_3. В програмі визначається значення максимального елементу масиву, використовуючи метод доступу через розіменування покажчика.

```
#include <stdlib.h>
#include <iostream>
using namespace std;
int main()
{int arr[10]={5,3,2,-4,6,7,110,-17,0,13},max,i;
  cout<<"Masiv arr\n";
  for(i=0; i<10; i++)
    cout<<arr[i]<<" ";
  int *px= arr; // визначення покажчика типу int на масив
  max=*px; //привласнення max значення нульового елемента масиву
  i=0;
  while(i<10)
  {if(*(px+i)>max) max=*(px+i);
    i++;
  }
  cout<<"\nmax = "<<max<<endl;
  system("pause");
  return 0;
}
```

Завдання до програми 4_3. Розробіть програму, в якій визначається масив цілих чисел заданого розміру. Перший елемент масиву задається, всі інші елементи вводяться з клавіатури, потім виконуються задані обчислення, використовуючи метод доступу до елементів масиву через розіменування покажчика. На екран виводяться значення елементів масиву і результати обчислень.

1. Визначте мінімальний елемент масиву, що кратний 5. Перший елемент масиву дорівнює **500**. Розмір масиву **8**.
2. Визначте номер мінімального додатного елемента масиву. Перший елемент масиву дорівнює **1000**. Розмір масиву **6**.
3. Визначте номер максимального від'ємного елемента масиву. Перший елемент масиву дорівнює **-2000**. Розмір масиву **7**.
4. Визначте максимальний парний елемент масиву. Перший елемент масиву дорівнює **-100**. Розмір масиву **8**.
5. Визначте мінімальний непарний елемент масиву. Перший елемент масиву дорівнює **201**. Розмір масиву **7**.

Програма 4_4*. Програма сортує елементи масиву методом вибору максимального елемента масиву.

```
#include <stdlib.h>
#include <iostream>
#include <iomanip>
using namespace std;
#define N 10
int main()
{int dig[N]={50,3,25,4,6,7,11,17,0,13};
  int i, j, mm, max, tmp;
  cout<<"Old massiv\n";
  for(i=0; i<N; i++) cout<<dig[i]<<" ";
  for(i=N-1; i>=1; i--)
  { max=dig[0]; mm=0;
    for(j=1; j<=i; j++)
      if(dig[j]>max){ max=dig[j];
                    mm=j;}
    tmp=dig[i];      // запам'ятати останній елемент
    dig[i]=max;      // максимум на місце останнього елемента
    dig[mm]=tmp;     // останній елемент на місце максимуму
  }
  cout<<"\nNew massiv\n";
  for(i=0; i<N; i++)
    cout<<dig[i]<<" ";
  cout<<endl;
  system("pause");
  return 0;
}
```

Завдання до програми 4_4*. Розробіть програму, що сортує заданим методом одновимірний масив цілих чисел та виводить масив до і після сортування.

1. Заповніть масив цілих чисел випадковими числами від **-40** до **40** та відсортуйте цей масив за убутанням бульбашковим методом.
2. Заповніть масив цілих чисел випадковими числами від **-60** до **60** та відсортуйте цей масив за зростанням методом прямого упорядкування.
3. Заповніть масив дійсних чисел випадковими числами від **-50.5** до **50.5** та відсортуйте цей масив за убутанням методом вибору мінімального елемента масиву.
4. Заповніть масив цілих чисел випадковими числами від **-100** до **100** та відсортуйте цей масив за зростанням бульбашковим методом.
5. Заповніть масив цілих чисел випадковими числами від **-50** до **50** та відсортуйте за убутанням масив чисел методом прямого упорядкування.

Програма 4_5*. Програма виконує введення з клавіатури рядка та виведення його на екран різними способами.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <iostream>
using namespace std;
#define K 80

int main()
{char str1[K],str2[K],str3[K]; //резервування пам'яті
// ведить рядок з 2-х або 3-х слів
puts("Input 1 string of 2 or 3 words:");
scanf("%s",str1); //введення рядка до натискання <Enter>
fflush(stdin); //очищення вхідного потоку
printf("\nString 1:\n%s\n",str1);//виведення рядка до прогалини
puts("\nInput 2 string of 2 or 3 words:");
gets(str2); // введення рядка до натискання <Enter>.
printf("\nString 2:\n");
puts(str2); //виведення всього рядка
cout<<"\nInput 3 string of 2 or 3 words:\n";
cin>>str3; // введення рядка до натискання <Enter>.
fflush(stdin); //очищення вхідного потоку
```

```

cout<<"\nString 3:\n";
cout<<str3<<endl;//виведення рядка до прогалини
string str4; //рядок типу string
cout<<"\nInput 4 string of 2 or 3 words:\n";
getline(cin,str4);// введення рядка до натискання <Enter>.
cout<<"\nString 4:\n";
cout<<str4<<endl;//виведення всього рядка
system("pause");
return 0;
}

```

Завдання до програми 4_5*. Розробіть програму, в якій визначаються три рядки. В перший рядок вводяться з клавіатури задані слова (з прогалинами), підраховується кількість символів в ньому. Другий рядок ініціалізується заданим повідомленням, і в ньому теж підраховується кількість символів. Потім в третій рядок заноситься сума першого рядка та другого рядка, і підраховується кількість символів. На екран виводиться всі рядки та кількість символів в них.

1. Введіть рядок з назвою предмету і прізвищем викладача. Визначте рядок з повідомленням, що консультація відбудеться в середу на п'ятій парі.
2. Введіть рядок з прізвищем, ім'ям свого друга. Визначте рядок з побажанням йому гарного настрою і успіхів в навчанні.
3. Введіть рядок з назвою фільму і прізвищем режисера. Визначте рядок з повідомленням, що фільм з таким фільмом з'явився в продажу.
4. Введіть рядок з назвою міста, вулиці й номера будинку. Визначте рядок з повідомленням, що на цю адресу надійшов лист.
5. Введіть рядок з назвою книги і прізвищем автора та сповістите, що така книга з'явилася в бібліотеці.

Контрольні питання до лабораторної роботи № 4

1. Що таке масив? Що Ви розумієте під ініціалізацією масиву? Приведіть приклад масиву з ініціалізацією.
2. Що таке глобальний, автоматичний та статичний масиви? Як вони неявно ініціалізуються?
3. Як можна задавати розмір масиву?
4. Що таке покажчик? Який розмір покажчика? Що визначає тип покажчика? Що показує різниця покажчиків?
5. Що таке покажчик-константа? Яка різниця між ім'ям масиву й звичайним покажчиком? Які помилкові конструкції в наведеному фрагменті програми? Чому?

```
int A[10], X[10];
int *PA, *PB;
PA=&A[0];
PB=A;
X=PA;
```

6. Визначено масив та покажчик на нього. Привласніть будь-які значення елементам масиву, використовуючи різні способи доступу.

```
int A[8]; int *ptrA=A;
```

7. Визначить та пояснить результат роботи програми:

```
#include <iostream>
using namespace std;
int main()
{short a[5]={50,30,20,60,80};
 short *p=a;
 cout<<"*p="<<*p<<endl;
 cout<<"*(p+2)="<<*(p+2)<<endl;
 cout<<"*p+2="<<*p+2<<endl;
 cout<<"(++p-1)="<<++p-1<<endl;
 cout<<"sizeof(a)="<<sizeof(a)<<endl;
 cout<<"sizeof(a[3])="<<sizeof(a[3])<<endl;
 cout<<"sizeof(p)="<<sizeof(p)<<endl;
 cout<<"sizeof(*p)="<<sizeof(*p)<<endl;
 return 0;
}
```

ЛАБОРАТОРНА РОБОТА №5

Двовимірні та динамічні масиви

Мета роботи – вдосконалення навичок роботи з різними масивами (двовимірними, покажчиків, динамічними) та покажчиками на масиви.

Програма 5_1. Програма обчислює суму значень елементів двовимірного масиву в рядках та стовпцях.

```
#include <stdlib.h>
#include <iostream>
#include <iomanip>
using namespace std;
int arr[4][3]={ {2,3,4}, {7}, {1,9}, {8,4,10}}, s1[4], s2[3];
int main()
{int i,j;
  cout<<"Array arr[4][3] \n";
  for(i=0;i<4;i++)
    {for(j=0;j<3;j++)
      cout<<setw(5)<<arr[i][j];
      cout<<endl;
    }
  cout<<"\n Sum elements strings\n";
  for(i=0;i<4;i++)
    {for(j=0;j<3;j++)
      s1[i]+=arr[i][j];
      cout<<"s1["<<i<<"]="<<s1[i]<<endl;
    }
  cout<<"\n Sum elements columns\n";
  for(j=0;j<3;j++)
    {for(i=0;i<4;i++)
      s2[j]+=arr[i][j];
      cout<<"s2["<<j<<"]="<<s2[j]<<endl;
    }
  system("pause"); return 0;}
```

Завдання до програми 5_1. Розробіть програму, в якій вводиться із клавіатури і виводиться на екран двовимірний масив цілих чисел заданого розміру та виконуються задані обчислення.

1. Обчисліть суму додатних непарних елементів в стовбцях масиву. Розмір масиву **5** на **4**.
2. Обчисліть кількість додатних парних елементів в рядках масиву. Розмір масиву **3** на **6**.

3. Обчисліть кількість непарних елементів в стовбцях масиву. Розмір масиву **5** на **3**.
4. Обчисліть суму від'ємних елементів в рядках. Розмір масиву **3** на **5**.
5. Обчисліть добуток елементів в діапазоні від **1** до **5** в стовбцях. Розмір масиву **4** на **5**.

Програма 5_2. В програмі розглядається можливість доступу до елементів двовимірної масиву різними способами.

```
#include <stdio.h>
#include <time.h> //для запуску генератора випадкових чисел
#include <stdlib.h> //для підключення генератора випадкових чисел
#include <iostream>
#include <iomanip>
using namespace std;
const int N=4, M=3;
float A[N][M], sum;
int main()
{
    srand((unsigned)time(NULL)); //запуск генератора випадк. чисел
    int i, j;
    cout<<"  Array A:\n";
    for(i=0; i<N; i++)
        {
            for(j=0; j<M; j++)
                {
                    /*(* (A+i)+j)=rand()%(10001*0.01f-50.0f); //від -50.00 до 50.00
                    cout<<setw(8)<<fixed<<setprecision(2);
                    cout<<* (A[i]+j);
                }
            cout<<endl; }
    for(i=0; i<N; i++)
        for(j=0; j<M; j++)
            if(A[i][j]>0&&A[i][j]<=30)
                sum+=(* (A+i))[j];
    cout<<"\n  Sum elements from 0 to 30 = "<<sum<<endl;
    system("pause"); return 0;}
```

Завдання до програми 5_2. Розробіть програму, в якій двовимірний масив заданого розміру заповнюється випадковими числами у заданому діапазоні і виводиться на екран. Потім виконуються задані обчислення, використовуючи різні способи доступу до елементів масиву.

1. Обчисліть кількість чисел в масиві, що більші за **15.5**. Розмір масиву **5** на **4**. Діапазон випадкових чисел від **10.50** до **30.50**.

2. Обчисліть середнє арифметичне додатних елементів. Розмір масиву **6** на **3**. Діапазон випадкових чисел від **-25.0** до **65.0**.
3. Обчисліть добуток ненульових елементів масиву. Розмір масиву **4** на **5**. Діапазон випадкових чисел від **-5.000** до **5.000**.
4. Обчисліть кількість чисел, що менші за **20.5**. Розмір масиву **5** на **3**. Діапазон випадкових чисел від **-10.5** до **+55.5**.
5. Обчисліть кількість від'ємних чисел в масиві. Розмір масиву **5** на **4**. Діапазон випадкових чисел від **-50.50** до **20.50**.

Програма 5_3. В програмі використовується масив покажчиків.

```
#include<stdlib.h>
#include<iostream>
using namespace std;
#include <string.h>
int main()
{ char *fi[3]={ "September", "October", "November"};
//string fi[3]={ "September", "October", "November"};
cout<<fi[0]<<endl<<fi[1]<<endl<<fi[2]<<endl;
cout<<"Size massiv pointer = "<<sizeof(fi);
cout<<"\nSize pointer 1 string = "<<sizeof(fi[0]);
cout<<"\nSize pointer 2 string = "<<sizeof(fi[1]);
cout<<"\nSize pointer 3 string = "<<sizeof(fi[2]);
cout<<"\n Massiv pointer string "<<endl;
for(int i=0;i<3;i++)
    cout<<fi[i]<<" len = "<<strlen(fi[i])<<endl;
//cout<<fi[i]<<" len = "<<fi[i].length()<<endl;
system("pause");
return 0;}
```

Завдання до програми 5_3*. Розробіть програму, в якій визначається масив покажчиків на три задані рядки. Ці рядки виводяться на екран, визначається їх розмір та розмір масиву покажчиків. Потім виконуються задані обчислення.

1. Визначте масив з **3**-х покажчиків на рядки з прізвищем, ім'ям та по батькові. У третьому рядку підрахуйте кількість букв 'о'.
2. Визначте масив з **3**-х покажчиків на рядки з назвами країни, міста та вулиці. У третьому рядку підрахуйте кількість букв 'е'.
3. Визначте масив з **3**-х покажчиків на рядки з назвами університету, факультету, спеціальності. У першому рядку підрахуйте кількість букв 'і'.

4. Визначте масив з **3**-х покажчиків на рядки з назвами континенту, країни та столиці цієї країни. У другому рядку підрахуйте кількість букв 'a'.
5. Визначте масив з **3**-х покажчиків на рядки з назвами книги, глави та параграфа. У першому рядку підрахуйте кількість букв 'с'.

Програма 5_4*. В програмі створюється динамічний масив для змінних типу **int**. Розмір масиву та значення елементів масиву вводяться з клавіатури. По закінченні роботи з масивом пам'ять звільняється.

```
#include <time.h> //для запуску генератора випадкових чисел
#include <stdlib.h> //для підключення генератора випадкових чисел
#include <iostream>
using namespace std;
int main()
{int n,i;
  int *ptr; //визначення покажчика на масив чисел типу int
  cout<<"Input size array\n";
  cin>>n;
  //запит пам'яті
  ptr=new int[n]; // ptr=(int*)malloc(n*sizeof(int));
  if(ptr==NULL) // перевірка виділення пам'яті
    {cout<<"Error\n"; return -1;}
  cout<<"\nInput array\n";
  for(i=0;i<n;i++) cin>>ptr[i];
  cout<<"\nArray\n";
  for(i=0;i<n;i++) cout<<*(ptr+i)<<" ";
  cout<<endl;
  delete [] ptr; // free(ptr); // звільнення пам'яті
  system("pause"); return 0;}
```

Завдання до програми 5_4. Розробіть програму, в якій створюється динамічний масив для змінних заданого типу з використанням операції **new** або функції **malloc**. Розмір масиву введіть з клавіатури. Масив заповніть випадковими числами у заданому діапазоні та виводиться на екран. Потім звільняється пам'ять.

1. Створіть динамічний масив для змінних типу **long** з використанням операції **new**. Масив заповніть випадковими числами у діапазоні від **-5000** до **+5000**.
2. Створіть динамічний масив для змінних типу **float** з використанням функції **malloc**. Масив заповніть випадковими числами у діапазоні від **-50.5** до **+50.5**.

3. Створіть динамічний масив для змінних типу **double** з використанням операції **new**. Масив заповніть випадковими числами у діапазоні від **0.00** до **32.00**.
4. Створіть динамічний масив для змінних типу **char** з використанням функції **malloc**. Масив заповніть випадковими числами у діапазоні від **60** до **125**. Масив виведіть на екран десятковому та символьному вигляді.
5. Створіть динамічний масив для змінних типу **double** з використанням операції **new**. Масив заповніть випадковими числами у діапазоні від **-15.000** до **15.000**.

Програма 5_5*. В програмі створюються двовимірні динамічні квадратні матриці. Розмір матриць вводиться з клавіатури. Значення елементів першої матриці вводяться з клавіатури, а другої – заповнюються випадковими числами від **-50** до **+50**. Потім в другу матрицю заносяться суми елементів обох матриць.

```
#include <stdlib.h>
#include <iostream>
#include <time.h>
#include <iomanip>
using namespace std;
int main()
{int i,j,N;
  cout<<"Input the size of the square matrix: \n";
  cin>>N;
  int **matr; // покажчик на масив покажчиків
  matr=new int*[N]; // запит пам'яті для масиву покажчиків
  if(matr==NULL){ cout<<"Error!\n"; return -1;}
  for(i=0;i<N;i++)
  {matr[i]=new int[N]; // запит пам'яті для елементів рядків
    if(matr[i]==NULL){cout<<"Error!\n";return -2;}
  }
  cout<<"\n Input the elements of the matrix matr:\n";
  for(i=0;i<N;i++)
    for(j=0;j<N;j++)
      cin>>matr[i][j];
  cout<<"\n\tMatrix matr:\n";
  for(i=0;i<N;i++)
    {for(j=0;j<N;j++) cout<<setw(8)<<matr[i][j];
      cout<<endl; }
```

```

int **mas;// Показчик на масив показчиків
mas=new int*[N];// Запит пам'яті для масиву показчиків
if(mas==NULL){cout<<"Error!\n"; return -1;}
for(i=0;i<N;i++)
{mas[i]=new int[N]; //Запит пам'яті для елементів рядків
  if(mas[i]==NULL) {cout<<"Error!\n"; return -2;}
}
srand((unsigned)time(NULL)); //запуск генератора випадк. чисел
cout<<"\n\tMatrix mas: \n";
for(i=0;i<N;i++)
{for(j=0;j<N;j++)
  {mas[i][j]=rand()%101-50;
   cout<<setw(8)<<mas[i][j]; }
  cout<<endl; }
cout<<"\nNew values of elements of matrix mas:\n";
for(i=0;i<N;i++)
{for(j=0;j<N;j++)
  {mas[i][j]=matr[i][j]+mas[i][j];
   cout<<setw(8)<<mas[i][j]; }
  cout<<endl; }
//Звільнення пам'яті
for(i=0;i<N;i++)
{delete matr[i]; // звільнення пам'яті з-під масивів елементів
 delete mas[i]; // звільнення пам'яті з-під масивів елементів
}
delete [] matr; // звільнення пам'яті з-під масиву показчиків
delete [] mas; // звільнення пам'яті з-під масиву показчиків
system("pause"); return 0;}

```

Завдання до програми 5_5*. Доповніть програму, створивши третю динамічну матрицю. Виконайте задані відповідно варіанту дії. Результат занесіть в третю матрицю та виведіть на екран.

1. Порівняйте значення елементів перших двох матриць, найбільше значення занесіть в третю матрицю.
2. Першу матрицю заповніть випадковими числами до **100**. В третю матрицю занесіть суму елементів перших двох матриць, якщо вона знаходиться в діапазоні від **60** до **120**, інакше занесіть число **46**. Результат виведіть на екран у символьному вигляді.
3. Порівняйте значення елементів перших двох матриць. Якщо відповідні елементи від'ємні, то в третю матрицю занесіть **-1**, якщо додатні – **+1**, інакше **0**.

4. Перемножьте першу та другу матрицю, результат занесіть в третю матрицю. Примітка: $c_{ij} = \sum_{k=0}^N a_{ik} b_{kj}$.
5. Першу матрицю заповніть випадковими числами від -100 до +100. В третю матрицю занесіть суму абсолютних значень елементів перших двох матриць.

Контрольні питання до лабораторної роботи № 5

1. Що таке багатовимірний масив? Як розташовуються в пам'яті елементи багатомірних масивів?
2. Наведіть приклади повної і неповної явної ініціалізації двовимірного масиву розміром **2** на **3** елементів типу **int**.
3. Визначте та поясніть результат роботи програми:

```
#include <iostream>
using namespace std;
int main()
{short m[2][3]={1,5,7,2,6,8};
  short *pm=m[0];
  cout<<"*(m+1)="<<*(m+1)<<" *(*(m+1)+1)="<<*(*(m+1)+1);
  cout<<"\n *pm+2="<<*pm+2<<" *(pm+2)="<<*(pm+2);
  cout<<"\n sizeof(m)="<<sizeof(m);
  cout<<"\n sizeof(m[0])="<<sizeof(m[0]);;
  cout<<"\n sizeof(m[0][0])="<<sizeof(m[0][0]);
  cout<<"\n sizeof(pm)="<<sizeof(pm) <<endl;
  return 0;}
```
4. Що таке масив покажчиків? Наведіть приклад з явною повною ініціалізацією такого масиву.
5. Що таке динамічний масив? Чим відрізняється від звичайного?
6. Створіть динамічний масив для **10** елементів типу **double** за допомогою операції **new**, перевірте, чи створився масив, та заповніть його випадковими числами. Звільніть пам'ять.
7. Створіть динамічний масив для **20** елементів типу **long** за допомогою функції **malloc**, перевірте, чи створився масив, та введіть його з клавіатури. Звільніть пам'ять.
8. Створіть динамічну матрицю **4** на **5** для елементів типу **float** за допомогою оператора **new..** Значенням елементів масиву привласніть випадкові числа від **-50.00** до **+50.00**. Звільніть пам'ять.

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. Шпак З.Я. Програмування мовою С: навч. посіб. – 2-е вид, доп. – Львів: Вид-во Львівської політехніки, 2011. – 431 с.
2. Подбельский В.В., Фомин С.С. Программирование на языке Си: учебник. – М.: ДМК Пресс, 2012. – 384 с.
3. Подбельский В.В. Практикум по программированию на языке Си: Учебн. пособие. – М.: Финансы и статистика, 2004. – 576 с.
4. Подбельский В.В. Стандартный Си++: Учеб. пособие. – М.: Финансы и статистика, 2008. – 688 с.
5. Дейтел Х., Дейтел П. Как программировать на С++. – 5-е изд. / Пер. с англ. – М.: ООО "Бином-Пресс", 2008. – 1456 с.
6. Дейтел П., Дейтел Х. С для программистов с введением в С11. – М.: ДМК Пресс, 2014. – 544 с.
7. Шилдт, Г. Полный справочник по С++. – 4-е изд. / Пер. с англ. – М.: Вильямс, 2008. – 800 с.
8. Прата С. Язык программирования С. Лекции и упражнения. – 6-е изд. / Пер. с англ. – М.: Вильямс, 2015. – 928 с.
9. Страуструп Б. Язык программирования С++. Краткий курс. – 2-е Издание. – СПб.: Диалектика, 2019. – 322 с.
10. Страуструп Б. Программирование: принципы и практика использования С++. – М.: Вильямс, 2011. – 1248 с.
11. Орленко П.А., Евдокимов П.В. С++ на примерах. Практика, практика и только практика. – СПб.: Наука и Техника, 2019. – 288 с.

Додаток А
Таблиця кодів ASCII основних символів

Код 10	Код 16	Символ	Код 10	Код 16	Символ	Код 10	Код 16	Символ
32	20	порожній символ	64	40	@	96	60	`
33	21	!	65	41	A	97	61	a
34	22	"	66	42	B	98	62	b
35	23	#	67	43	C	99	63	c
36	24	\$	68	44	D	100	64	d
37	25	%	69	45	E	101	65	e
38	26	&	70	46	F	102	66	f
39	27	'	71	47	G	103	67	g
40	28	(	72	48	H	104	68	h
41	29	)	73	49	I	105	69	i
42	2A	*	74	4A	J	106	6A	j
43	2B	+	75	4B	K	107	6B	k
44	2C	,	76	4C	L	108	6C	l
45	2D	-	77	4D	M	109	6D	m
46	2E	.	78	4E	N	110	6E	n
47	2F	/	79	4F	O	111	6F	o
48	30	0	80	50	P	112	70	p
49	31	1	81	51	Q	113	71	q
50	32	2	82	52	R	114	72	r
51	33	3	83	53	S	115	73	s
52	34	4	84	54	T	116	74	t
53	35	5	85	55	U	117	75	u
54	36	6	86	56	V	118	76	v
55	37	7	87	57	W	119	77	w
56	38	8	88	58	X	120	78	x
57	39	9	89	59	Y	121	79	y
58	3A	:	90	5A	Z	122	7A	z
59	3B	;	91	5B	[	123	7B	{
60	3C	<	92	5C	\	124	7C	
61	3D	=	93	5D	]	125	7D	}
62	3E	>	94	5E	^	126	7E	~
63	3F	?	95	5F	_	127	7F	del