**Практична робота №6.Написання програм з використанням масивів та структур**

**Мета**: навчитися писати програми з використанням структур та масивів **на мові С++** в **консольному режимі.**

**Завдання.**

1. Ознайомитися з теоретичною частиною.
2. Запустити середовище програмування С++ . Записати програму, що виконує 3 завдання з пп.3.1-3.2. В першому рядку кожної програми записати

*// Група № Прізвище*

вказавши номер своєї групи та своє прізвище

1. В головній програмі скласти командний код подання меню та виклику функцій, в яких надається код, розроблений за пп.3.1-3.2. Як і в ПР №5, програма повинна запитати номер завдання (число 1, 2, 3 або 0 для закінчення) і в залежності від введеного значення виконувати відповідне завдання. Якщо введений 0 – програма припиняє роботу. В програмі використати потокове введення та виведення даних.

3.1. Написати код, який заповнює масив з 10 елементів випадковими числами з діапазону від 200 до 300, виводить всі елементи масиву з вказівкою індексу масиву, розділивши їх табуляцією, виконує пошук мінімального та максимального числа та їх виведення. *Доступ до масиву здійснити через індекси.*

3.2. Написати код, який заповнює масив з 10 елементів випадковими числами з діапазону від -20 до 20, виводить всі елементи масиву з вказівкою індексу масиву, розділивши їх табуляцією, виконує заміну всіх негативних чисел на 55, а позитивні робить негативними. Вивести отриманий масив аналогічно заповненому. *Доступ до масиву здійснити через покажчики*.

1. Результати надсилати на електронну адресу викладача [**t.i.lumpova@gmail.com**](mailto:t.i.lumpova@gmail.com)у вигляді cpp-файлу з іменем у форматі

**<Номер групи><Номер лабораторної><Прізвище англійською>**

Наприклад, 31-01Ivanov.cpp.

Іншим рішенням є надсилання поштою посилання на текст програми за URL адреси, яку надає C++Shell, вказавши в темі листа, номер групи прізвище студента та номер ПР.

В темі листа вказати, номер групи, прізвище студента та номер ПР як "ПР№6".

**Строк відсилки ПР для МІВТ/ЕТ-41 03.11.2020**

**МНТ/ЕТ- 41 03.11.2020.**

Всі запитання, що виникнуть, надсилайте на електронну адресу викладача, В темі листа вказати, номер групи, прізвище студента та номер ПР як "ПР№6" В темі листа вказати, номер групи, прізвище студента, номер ПР та фразу

**Теоретичні відомості.**

**Вкладені структури**.

Студентські списки містять дані: прізвище та ініціали, дата (день, місяць, рік) народження, група та середній бал успішності, а в списках викладачів присутні такі дані: прізвище, ініціали, дата народження, кафедра, посада. У процесі обробки списку студентів і списку викладачів можна оголосити відповідно такі структури:

**struct stud**

**{ char fio [25];**

**int den, god;**

**char mes [10];**

**char grup;**

**flout sb; }**

**struct prep  
{ char fio [25];  
int den, god;  
char mes [10];  
char kaf, dolg;  
}**

В оголошених типах однакові поля можна об’єднати в окрему структуру і застосовувати її при описі інших типів. Поетапно це виглядає так:

* загальна структура —

**struct spd**

**{ char fio [25];**

**int den, god;**

**char mas[10]; }**

* структура для опису інформації про студентів —

**struct stud**

**{ spd dr;**

**char grup;**

**float sb}**

**st1, st2;**

* структура для опису інформації про викладачів —

**struct prep**

**{ spd dr;**

**char kaf [10];**

**char dolg [15];**

**} pr1, pr2;**

У структурах **stud і prep** для оголошення поля, що містить дані про прізвище і дату народження, використовується раніше описаний тип **spd**. Тепер до поля **fio, den, god, mes**можна звернутися, використовуючи запис **st1.dr.fio**, наприклад, при зверненні до функції введення:

**gets (st1.dr.fio);** або    **gets (pr1.dr.fio);**

**Пояснення.**

Прототип функції gets: (б-ка cstdio) char \* gets( char \* string );

Отримати рядок зі стандартного потоку введення. Функція gets зчитує символи зі стандартного потоку введення до символу нового рядка n або доки, поки не буде досягнутий кінець файлу EOF, після чого зберігає лічені символи в рядок типу char.

Символ нового рядка n не копіюватиметься в рядок.

Нульовий символ автоматично додається після останнього копійованого символу в string, щоб сигналізувати про кінець рядка.

Функція gets дещо відрізняється від функції fgets, по-перше повертає рядок зі стандартного потоку введення в якості аргументу, а, по-друге, кінцевий символ нового рядка не входить до її складу . на відміну від fgets. А ще, gets не дозволяє встановити обмеження на кількість зчитувальних символів, тому потрібно бути обережними з розміром масиву, на який вказує параметр str, щоб уникнути переповнення буфера.

Параметри: **string**Покажчик на масив типу char, в якому зберігається рядок.

Значення, що повертається. У разі успіху, функція повертає той же параметр, string. Якщо відбувається помилка, повертається нульовий покажчик.

Використовуйте функції ferror або feof для перевірки внутрішніх станів показників помилки або кінця файлу – EOF.

Після оголошення структурного типу змінних для роботи з їхніми полями можна застосовувати і покажчики, тоді опис структури матиме вигляд:

**struct stud**

**{ char fam [25];**

**int mat, fiz, prg;**

**float sb;**

**} st1, \*pst;**

Доступ до полів може здійснюватися двома способами:

* з використанням операції розіменування «\*», тобто  
  **gets ((\*pst).fam);    (\*pst).fiz = 5;**
* з використанням покажчика **->**, наприклад,  
  **gets (pst -> fam);      pst-> fiz = 5;** тощо.

**a->b це Розкриття посилання на структуру ("на член *b* об'єкту вказує *a*")**

Крім того, до полів змінної **st1** можна звертатися, вказуючи поля через операцію **«.»**, як це робилося раніше.

Дані типу структура можна об’єднати в масиви, для попереднього прикладу з урахуванням кількості студентів, структуру можна записати так:

**struct stud**

**{ char fam [20];**

**int mat, fiz, prg;**

**float sb;**

**} spis[15], \*sp = &spis[0];.**

У випадку, коли масив описується десь у тексті програми, тобто не саме після опису структури, його можна оголосити у вигляді: **stud spis [15];** — масив типу структура з ім’ям **stud**, що містить відповідну інформацію про групу із **15** студентів.

Доступ до елементів масиву типу структура здійснюється із застосуванням індексу або через покажчик-константу, яким є ім’я масиву, тобто одним з таких способів:

**strcpy (spis[1].fam, ” “);**

**spis[1].fiz = 5;**

або

**strcpy ((sp + 1) -> fam, ” “);**

**(sp + 1) -> fiz = 5;**

або

**strcpy ((\*(spis + 1)).fam, ” “);**

**(\*(sp + 1)).fiz = 5;.**

У останньому виразі **(\*(spis + 1)).fiz = 5;** потрібна зовнішня пара дужок, тому що **операція «.» («крапка») має пріоритет вище, ніж операція розіменування «\*».**

Поля структури можуть також бути масивами. Наприклад, у розглянутій структурі **stud** можна оцінки з різних предметів об’єднати в масив. Тоді структуру слід описати у вигляді:

**struct stud1**

**{ char fam [25];**

**int pred [3];**

**float sb**

**} spis[15],** **\*ps = &spis[0];.**

Звернення до полів здійснюватиметься одним із способів:

**((\*ps).fam)**

**(ps->pred [0]),**

наприклад,

**gets ((\*ps).fam);**  
**сіn >> ps -> pred[0] >> ps -> pred[1] >> ps -> pred[2];**

або**сіn >> ps -> \*(pred + 0) >> ps -> \*(pred + 1) >> ps -> \*(pred + 2).**

Доцільно також зазначити, що **бібліотека stdlib.h містить спеціальні функції для пошуку та сортування структурних змінних.**

**Масиви (повторення)**

Масив являє собою сукупність даних, що організована певним чином. Тобто масив – це окремий випадок структури даних.

**Властивості масивів**

* масив складається з елементів, які мають однаковий тип;
* елементи масиву послідовно, без проміжків, розташовані в одній ділянці оперативної пам’яті;
* кожен з елементів масиву має свій порядковий номер, що зветься індексом;
* нумерація елементів починається з 0;
* до елементів масиву можна звертатися використовуючи ім’я масиву і індекс у квадратних дужках.
* масив може бути одновимірним, або багатовимірним,
* У двовимірному масиві кожний з елементів є одновимірним масивом;
* У тривимірному масиві кожний з елементів є двовимірним масивом.

Оголошення одновимірного масиву

**<тип елементів> <ім’я масиву> [<кількість елементів>] ;**

Приклади оголошень масивів:

int arr [10] ;

int A [10] ={2,5,10};

int A [] ={2, 5, 10, 3, 6, 0, 9, 4, 5, 7};

Доступ до елементів масиву

Для доступу до елементів масиву використовується синтаксична конструкція, що складається з імені масиву та індексу, який записується у квадратних дужках.

Наприклад, A[i], A [0], A [9].

Індексовані елементи масиву можуть бути використані так само, як і прості змінні. Вони можуть бути операндами у виразах, їм можна присвоювати значення, відповідні їх типу.

int arr[ ] ={2, 5, 10, 3, 6, 0, 9, 4, 5, 7}



Масиви символів

* У мові С++ символьний рядок може розглядатися як масив символів.
* Для масиву символів у пам’яті виділяється ділянка, розмір якої на один байт більше, ніж кількість символів у рядку.
* У цей додатковий байт записується ознака кінця рядка - символ ‘\0’.

char <ім’я символьного рядка> [<кількість байтів >] ;

Приклад:

char str[80] ;

char str[10] ={‘H’, ‘e’, ‘l’, ‘l’, ‘o’, ‘!’, ‘\0’} ;

char str[10] =”Hello!”;

char str[] =”Hello!”;

Особливості роботи з масивами в С++

* Ніякого контролю за значеннями індексів, що використовуються для доступу до елементів масиву, нема.
* Ви можете звернутися до «елементу масиву» з номером, який більше максимального, але отримаєте невідомо що. Ще гірше, якщо ви щось напишете за цим номером

Приклади

**Заповнення масиву випадковими числами**

В С ++ для цього є спеціальні фyнкції **rand()**. Вони знаходяться в бібліoтечному файлі cstdlib, тому щоб їх застосовувати в програмі, необхідно підключити цей бібліотечний файл: #include <cstdlib>.

Окрім того, для ця функція може використовувати поточний системний час через використання функції time(), для якої необхідно підключити бібліотечний файл ctime: #include <ctime>.

Діапазон випадкових чисел, які генеруються таким чином – від 0 до 32767. Можливо вам знадобиться заповнити масив числами від 200 дo 300, від 0.1 дo 1, від -20 дo 20. Таку генерацію випадкових чисел можливо і нескладно реалізувати. У прикладі нижче розглянуто кілька випадків:

**#include <iostream>**

**#include <cstdlib>**

**#include <ctime>**

**using namespace std;**

**int main()**

**{**

**int randomDigits[10];**

**int randomDigits\_2[10];**

**int randomDigits\_3[10];**

**int randomDigits\_4[10];**

**float randomDigits\_5[10]; // для чисел з плаваючею точкою**

**srand(time(NULL));**

**for (int i = 0; i < 10; i++)**

**{**

**randomDigits[i] = rand() % 7; // 0 ... 6**

**randomDigits\_2[i] = 1 + rand() % 7; // 1 ... 7**

**randomDigits\_3[i] = 200 + rand() % 101; // 200 ... 300**

**randomDigits\_4[i] = rand() % 41 - 20; // -20 ... 20**

**randomDigits\_5[i] = 0.01 \* (rand() % 101);// 0.01 ... 1**

**}**

**cout << "Масив з числами від 0 до 6: ";**

**for (int i = 0; i < 10; i++)**

**{**

**cout << randomDigits[i] << " ";**

**}**

**cout << endl << "Масив з числами від 1 до 7: ";**

**for (int i = 0; i < 10; i++)**

**{**

**cout << randomDigits\_2[i] << " ";**

**}**

**cout << endl << "Масив з числами від 200 дo 300: ";**

**for (int i = 0; i < 10; i++)**

**{**

**cout << randomDigits\_3[i] << " ";**

**}**

**cout << endl << "Масив з числами від -20 до 20: ";**

**for (int i = 0; i < 10; i++)**

**{ cout << randomDigits\_4[i] << " "; }**

**cout << endl << "Масив з числами від 0.01 дo 1: ";**

**for (int i = 0; i < 10; i++)**

**{ cout << randomDigits\_5[i] << " "; }**

**cout << endl;**

**}**

**Виведення масиву на консоль**

**int const size=10;**

**int ar[size];**

**/\* Здійснити заповнення масиву через введення даних або ініціалізацію \*/**

**for(int i=0; i<size; i++){**

**cout<<ar[i];**

**if(i<size-1) cout<<'\t';**

**}**

**cout<<endl;**

**Обчислення суми елементів масиву**

**int const size =10;**

**int ar[size];**

**/\* Здійснити заповнення масиву через введення даних або ініціалізацію \*/**

**int sum=0;**

**for (int i = 0; i < size; ++i) {**

**sum+=ar[ i ];**

**}**

**cout<< "sum="<< sum << endl;**

**Пошук найбільшого елементу у масиві**

**int const size =10;**

**int ar[size];**

**/\* Здійснити заповнення масиву через введення даних або ініціалізацію \*/**

**int mx=0;**

**for(int i=0; i<size;i++)**

**if (ar[i]>mx) mx=ar[i];**

**cout<< "mx="<< mx<<endl;**

**Пошук найменшого елементу у масиві та його індексу**

**int const size =10;**

**int ar[size];**

**/\* Здійснити заповнення масиву через введення даних або ініціалізацію \*/**

**int mn=ar[0];**

**for(int i=1; i<size;i++)**

**if (ar[i]<mn) mn=ar[i];**

**cout<< "mn="<< mn<<endl;**

**Вилучення елементу із масиву**

**int const size =10;**

**int ar[size];**

**for(int i=0; i<size; i++) {ar[i]=200+ rand() % 5;**

**cout<<ar[i]<<endl;**

**}**

**int element = 200;**

**int size1 = size;**

**/\* Здійснити заповнення масиву через введення даних або ініціалізацію \*/**

**int i=0;**

**while(i < size1){**

**if(ar[i] == element) {**

**size1 = size1 - 1;**

**for(int j = i; j<size1; j++)**

**ar[ j ] = ar[ j+1 ];**

**}**

**else i++;**

**}**

**Вставка елементу до масиву**

**int x = 55;**

**int pos = 3;**

**int size1 = size;**

**/\* Здійснити заповнення масиву через введення даних або ініціалізацію \*/**

**if(pos<0 || pos>size) cout << "Bad position "<<endl;**

**else {**

**for(int i=size1;i>pos;i--)**

**{ ar[i]=ar[i-1];**

**ar[pos]=x;**

**size1++; }**

**};**

**Пошук позиції символу у рядку**

**char str[] = " Bad news has wings ";**

**char ch='s';**

**int NumPos=-1; //Номер позиції шуканого символу;**

**// =-1 символ не знайдений**

**for(uint i=0; str[i] != '\0'; i++)**

**if(str[i]==ch) { NumPos= i; break;}**

**cout << "Position "<< NumPos <<endl;;**

**Приклад роботи з масивом через покажчики**

Розрахунок середнього арифметичного позитивних елементів

**#include <iostream>**

**#include <string>**

**using namespace std;**

**int main ( )**

**{ system("color F0");**

**const int n = 10;**

**float mas[n], s;**

**int i, kol = 0;**

**for (i = 0, s = 0; i < n; i++)**

**{ cin >> \*(mas+i);**

**if (\*(mas+i) > 0)**

**{ s += \*(mas+i);**

**kol++; }**

**}**

**cout.precision(3);**

**cout <<"\n average positive elements = " << s/kol << endl;**

**system("pause");**

**return 0;**

**}**

Результати роботи

