ЛАБОРАТОРНА РОБОТА №1

Тема: Вивчення базових понять класу.

Мета: Закріпити базові знання про клас. Навчитись створювати класи засобами мови С++

КОРОТКІ ТЕОРЕТИЧНІ ВІДОМОСТІ

Крім визначення нових операцій за допомогою функцій, в С++ можна визначити і новий тип даних за допомогою конструкції класу. Клас в С++ — це структурований тип, створений на основі вже існуючих типів. У цьому сенсі клас є розширенням поняття структури.

Визначення класу

Клас можна визначити за допомогою конструкції

```
тип_класу ім'я_класу { компоненти класу };
```

Крапка з комою в кінці ставиться обов'язково. У цьому визначенні:

- тип класу — одне із службових слів class, struct, union;
- ім'я класу — ідентифікатор;
- компоненти класу — визначення і оголошення даних (атрибутів) та функцій (методів), що належать класу функцій.

Ім'я класу є по замовчуваню ім'ям типу об'єктів. Дані — це поля (атрибути) об'єкту, що утворюють його структуру. Значення атрибутів визначають стан об'єкту. Функція, що є компонентом класу, називається методом класу. Методами класу визначаються операції над об'єктами класу. Дозволяється визначати клас:

- з атрибутами і методами;
- тільки з атрибутами;
- тільки з методами;
- без атрибутів і без методів.

Клас без атрибутів і без методів називається порожнім класом.

Елементи класу типу struct за замовчуванням відкриті і доступні в інших частинах програми. Члени класу class за замовчуванням закриті і недоступні зовні відносно даного класу. Доступ до вмісту класу задається специфікаторами доступу, які позначаються ключовими словами public і private. Ключове слово public оголошує елемент класу, який доступний поза класом, а ключове слово private закриває доступ зовні. Після ключового слова потрібно поставити знак «:» (двокрапка).

І в класі, і в структурі можна написати стільки специфікаторів public і private, скільки необхідно, і в тому порядку, в якому потрібний. Черговий специфікатор діє до наступного. Допускається індивідуально оголошувати кожен елемент класу або відкритим, або закритим. Відкрита частина класу називається інтерфейсом. Прихованість інформації про внутрішню структуру — це один з принципів об'єктно-орієнтованого програмування, так звана інкапсуляція.

Приклади оголошень класів приведені в лістингу 1.1.

Лістинг 1.1. Приклади оголошень класів

```
class Null Type { };// порожній клас

struct Empty { };// пуста структура

struct Pair // тільки відкриті атрибути
{
	int first;
	double second;
};

struct date// і поля і методи - відкриті
{
	int month,day,year;// поля: місяць, день,  рік
	void set(int d, int m, int y);// метод - встановити дату
	void get (int &d,int &m,int &y);// метод - отримати дату
	void next();// метод - встановити наступну дату
	void print();// метод - показати дату
};

struct Money
{
	void Add(const long &r);// методи - відкриті
	void DISPLAY();
private:
	long Rubls;// закритий атрибут
};

class Fraction
{
	int num; int denum;// атрибути - закриті
public:
	void reduce();// метод - відкритий
};

class XXX
{
	int yyy();    // метод - закритий
public:
	int fff(int t);      // метод - відкритий
	int RRR;      // атрибут - відкритий
};

class Library // тільки відкриті методи
{
public:
	void furiction();
	int Function(int);
};
```

Використання класу

Маючи визначення класу, можна оголошувати змінні типу «ім’я_класу». Змінна класу називається об'єктом, або екземпляром класу. Клас оголошується один раз, а ось об'єктів можна створити стільки, скільки необхідно. Об'єкти оголошуються аналогічно змінним вбудованого типу:

```
ім’я_класу ім’я_об’єкту;// скалярний об'єкт
ім’я_класу *ім’я_об’єкту;// вказівник
ім’я_класу ім’я_об’єкту[кількість];//масив
```

При оголошенні змінних можна задавати ключові слова class і struct:

```
class ім’я_класу ім’я_об’єкту;// скалярний об'єкт
class ім’я_класу * ім’я_об’єкту;// вказівник
struct ім’я_класу ім’я_об’єкту[кількість];// масив
```

Дозволяється поєднувати визначення класу і оголошення змінних:

```
class ім’я_класу { члени_класу  } ім’я_об’єкту;// об'єкт
class ім’я_класу { члени класу  } * ім’я_об’єкту;// вказівник
struct ім’я_класу { члени_класу } ім’я_об’єкту [кількість];// масив
```

Ці змінні отримують тип «ім’я_класу». Надалі можна оголошувати змінні без запису визначення класу і без службових слів class або struct, як показано вище. Оголошені змінні підлягають правилам видимості, як і змінні вбудованих типів, і час їх життя залежить від місця оголошення.

Інші змінні можуть бути оголошені як приведено в лістингу 1.1:

```
NullType a, *pb;// скалярна змінна і вказівник
Empty d[10];// масив
date *pd;// вказівник
class Money t;// скалярна змінна
class Money *p;// вказівник
class Money m[100];// масив
XXX YYY;// скалярна змінна
Library L[10], *pl;// масив і вказівник
class Class{ /*... */ }// визначення класу
v1, v2[20];// скалярна змінна і масив
```

Змінним v1 і v2 присвоюється тип Class, після чого для оголошення інших змінних достатньо вказання типу Class, наприклад

```
Class v3, v4[10];
```

Змінну можна ініціалізувати іншою змінною того ж класу. Дозволяється оголошувати і посилання на об'єкти із обов'язковою ініціалізацією.

```
Money x = t;// ініціалізація змінної
date y(d);// ініціалізація змінної
date &u = y;// обов'язкова ініціалізація посилання
```

Якщо атрибути відкриті, виконується звичайна ініціалізація атрибутів ініціалізатором структури, наприклад

```
class Person
{
public:
       string Fio;
       double Summa;
};

Person Kupaev = { "Купаєв М.", 10000.00 };
```

Об'єкти класу дозволяється визначати як атрибути іншого класу:

```
class Count
```

```
{
        Money Summa;// сума на рахунку
        unsigned  long NumberCount;// номер рахунку
        date D;// дата відкриття рахунку
public :
        void DISPLAY();// вивід на екран
};
```

Використання об'єкту як атрибути іншого класу називається композицією.

Об'єкти класу можна передавати як аргументи будь-якими способами (за значенням, по посиланню, по вказівнику) і отримувати як результат із функції:

```
Money AddMoney(const Money &a, const Money &b); //по константному
посиланню
Money InitCconst long double &r);//повернення значення
void f1(date t);//за значенням
void f2(date &t);//по посиланню
void f3(date *t);//по вказівнику
```

Функція, що повертає об'єкт класу, може бути використана для ініціалізації змінної класу:

```
Money t = Init(200.56);
Money r= AddMoney(x,y);
```

Можна оголошувати об'єкти-константи класу з обов'язковою ініціалізацією:

```
const Money k = p;
const date d(t);
const Money t = Init (200,50);
```

Доступ до відкритих атрибутів виконується звичайними для структури способами:

Об’єкт.атрибут вказівник ->атрибут

Виклик відкритих методів здійснюється аналогічно

Об’єкт.метод(параметри) вказівник ->метод(параметри)

Для масиву об'єктів звернення до відкритого атрибути виконується так: *Ім’я_масиву [індекс].атрибут*

Аналогічно виконується і виклик методу для елементу масиву

Ім’я_масиву [індекс].метод(аргументи)

Приклади:

```
date d;
d.day = 21: d.month = 11; d.year - 2006:
d.print();
d.set(22,2,1953);
Library t;
int a = t.function(5);
Library *pt = new Library;
int b = pt->function(5);
Count A[5];// оголошення масиву
for(int i = 0; 1 < 5; i++)
       A[i].Display();// виведення елементів масиву
```

Визначення розміру в пам'яті, що виділяється об'єктам (у байтах) виконується операцією sizeof (ім’я_класу). Методи не займають місце в класі, а фактичний розмір класу залежить від режиму вирівнювання. Навіть порожній клас займає деяку частину пам'яті. По замовчуванню в кожному інтегрованому

середовищі встановлений власний режим вирівнювання, що впливає на розмір класу. Вирівнювання по границі байта можна встановити за допомогою директиви препроцесора `#pragma pack(1)` або `#pragma pack(push, 1)`

В цьому випадку класу потрібний найменший об'єм пам'яті. Конструкція Class ім'я-класу; називається оголошенням класу. Зазвичай таке оголошення використовується в тих випадках, коли один клас залежить від іншого, але визначення другого класу недоступне. Об'єкти такого класу оголошувати не можна — сам клас не визначений. Дозволяється оголошувати вказівники на об'єкти такого класу.

Визначення методів класу

Методи класу мають необмежений доступ до всіх елементів класу незалежно ні від порядку оголошення елементів класу, ні від специфікаторів доступу. Методи можуть визначатися як всередині класу, так і по за ним. Визначення методу всередині класу не відрізняється від визначення звичайної функції. Метод, визначений всередині класу, вважається по замовчуванням вбудованою функцією (inline).

Якщо метод визначається поза класом, то приналежність методу до класу вказується префіксним ім'ям класу. У класі присутній лише прототип. Метод, визначений по за класом, по замовчуванню не вважається inline-функцією. Методи можуть бути перегружені і можуть приймати аргументи по default. Аргументом і/або значенням, що повертається, в методі може бути об'єкт того ж класу, і такі об'єкти дозволено оголошувати всередині методу.

Фактично лівим аргументом методу є об'єкт, для якого цей метод викликається, наприклад

```
time t;
t.settimet (12,54):
```

Метод неявно отримує як аргумент вказівник на об'єкт, для якого він викликаний. Цей вказівник позначається зарезервованим словом this і може використовуватися в тілі методу. Через нього ж розумно здійснювати доступ до атрибутів і методів класу у разі неоднозначності (однакових імен).

```
this->Summa
```

Запис *this є значенням поточного об'єкту. Часто це значення використовується для повернення значення визначеного класу.

Методи можуть бути перевантажені. Перевантаження методів — це один з проявів принципу поліморфізму в С++.

Методи можуть бути константними. Константний метод не змінює значення атрибутів класу. Константний метод оголошується з допомогою слова const після списку аргументів методу. Константні і неконстантні методи не є еквівалентними, навіть якщо у них повністю співпадають прототипи (окрім слова const після списку аргументів). Для об'єкту-константи може бути викликаний тільки константний метод. Для об'єкту-змінної можна викликати як константні, так і неконстантні методи.

Для введення значень нового типу зазвичай реалізується метод Read( ). В простому вигляді виконується заповнення атрибутів нового класу із стандартного потоку вводу cin з клавіатури. У методі перед вводом кожного значення можна

вивести підказку, що пояснює черговість набору. значення, що передаються, зазвичай перевіряються на допустимість.

Для виведення значень нового типу реалізується метод DISPLAY( ). У найпростішому випадку він є виводом із атрибутів нового класу на екран — у стандартний потік cout. Метод виводу реалізується як константний (див. лістинг 1.2).

Лістинг 1.2. Визначення методів

```
using namespace std;

class time
{
public:
	void settime (int h, int m, int s = 0)// визначення всередині
класу
	{
		hours=h; minutes=m; seconds=s;
	}
	void settime(const time &t);// метод перевантаження
	time nexthour();
	time addhours(const time &t);
	void Display() const;
	void Read();
private:
	int hours, minutes, seconds;
};

void time::Display() const
{cout<<hours<<"."<<minutes<<"."<<seconds; }

void time::Read()
{
	do {
		cout<<"Hours:";
		cin>>hours;}
	while (hours>23);
	cout<<"Minutes:";
	cin>>minutes;
	seconds=0;
}

void time::settime(const time &t)
{
	hours=t.hours;
	minutes=t.minutes;
	seconds=t.seconds;
}

time time::addhours(const time &t)
{
	time r= *this;
	r.hours+=t.hours;
```

```
        r.hours%=24;
        return r;
}
time time::nexthour()
{
        hours++;
        hours%=24;
        return *this; // повернення поточного об'єкту
}
```

Ю.І. Грицюк, Т.Є. Рак. Розділ 2-3. с. 29-80

ПОРЯДОК ВИКОНАННЯ РОБОТИ

Завдання 1. Створіть клас Int, що імітує стандартний тип int. Єдине поле цього класу повинно мати тип int. Створіть методи, які будуть встановлювати значення поля рівним нулю, ініціалізувати його цілим значенням, виводити значення поля на екран і складати два значення типу Int.

Напишіть програму, в якій будуть створені три об'єкти класу Int, два з яких будуть ініціалізованими. Додайте два ініціалізованних об'єкта, надайте результат третьому, а потім відобразіть результат на екрані.

Завдання 2. Уявіть пункт для прийому платежів за проїзд по автостраді. Кожна проїжджаюча машина повинна заплатити за проїзд 50 центів, однак частина машин платить за проїзд, а частина проїжджає безкоштовно. У касі ведеться облік числа проїхавших машин і сумарна виручка від плати за проїзд.

Створіть модель такої каси за допомогою класу Kasa. Клас повинен містити два поля. Одне з них, типу unsigned int, призначене для обліку кількості проїхали автомобілів, а друге, що матиме тип double, міститиме сумарну виручку від оплати проїзду. Конструктор повинен ініціалізувати обидва поля нульовими значеннями. Метод payingCar () інкрементує число машин і збільшує на 0,50 сумарну виручку. Інший метод, norayCar (), збільшує на одиницю число автомобілів, але залишає без зміни виручку. Метод display () виводить обидва значення на екран. Там, де це можливо, зробіть методи константними.

Створіть програму, яка продемонструє роботу класу. Програма повинна запропонувати користувачеві натиснути одну клавішу для того, щоб зімітувати оплату водієм, і іншу клавішу, щоб зімітувати несумлінного водія. Натискання клавіші Esc повинно привести до видачі поточних значень кількості машин і виручки, і до завершення програми.

Завдання 3. Створіть клас з ім'ям time, що містить три поля типу int, призначених для зберігання годин, хвилин і секунд. Один з конструкторів класу повинен ініціалізувати поля нульовими значеннями, а інший конструктор - заданим набором значень. Створіть

метод класу, який буде виводити значення полів на екран у форматі 11:59:59, і метод, складає значення двох об'єктів типу time, переданих в якості аргументів.

У функції main () слід створити два ініціалізованих об'єкта (подумайте, чи повинні вони бути константними) і один неініціалізованний об'єкт. Потім складіть два ініціалізованих значення, а результат надайте третьому об'єкту і виведіть його значення на екран. Де можливо, зробіть методи константними.

Завдання 4. * У кожному завданні потрібно реалізувати клас. У програмі обов'язково повинні бути продемонстровані різні способи створення об'єктів і масивів об'єктів. Програма повинна демонструвати використання всіх функцій і методів.

У всіх завданнях обов’язково повинні бути присутні:

- метод ініціалізації Init( ), метод повинен контролювати значення аргументів на коректність;
- введення з клавіатури Read( );
- виведення на екран Display( ).

Варіанти:

1 Атрибут first – ціле число, ціла частина числа; атрибут second – додатнє ціле число, дробова частина числа. Реалізувати метод multiply() – множення на довільне ціле число типу int. Метод повинен правильно працювати при любих допустимих значеннях first і second..

2 Атрибут first – дробове число; атрибут second – дробове число, показник степені. Реалізувати метод power() - піднесення числа first в степінь second. Метод повинен правильно працювати при будь-яких допустимих значеннях fist і second.

3 Атрибут first – ціле додатнє число, чисельник; атрибут second – ціле додатнє число, знаменник. Реалізувати метод ipart() – виділення цілої частини дробу fist/second. Метод повинен перевіряти нерівність знаменника нулю.

4 Атрибут first – ціле додатнє число , номінал купюри; номінал може приймати значення 1,2,5,10,50,100,500,1000,5000. Атрибут second – ціле додатнє число , кількість купюр данного достоїнства. Реалізувати метод summa() – обчислення грошової суми.

5 Атрибут first –дробове додатнє число , ціна товару; Атрибут second – ціле додатнє число , кількість одиниць товару. Реалізувати метод cost() – обчислення вартості товару.

6 Атрибут first – ціле додатнє число, калорійність100г продукту; атрибут second – дробове додатнє число, маса продукту в кілограмах. Реалізувати метод power() - обчислення загальної калорійності продукту.

7 Атрибут first – дробове число , ліва границя діапазону ; атрибут second – дробове число, права границя діапазону. Реалізувати метод rangecheck()- перевірку заданого числа на приналежність діапазону.

8 Атрибут first – ціле число, ліва границя діапазону, включається в діапазон, атрибут second –ціле число, права границя діапазону, не включається в діапазон. Пара чисел представляє напіввідкритий інтервал(fist,second). . Реалізувати метод rangecheck()- перевірку заданого цілого числа на приналежність діапазону.

9 Атрибут first – ціле додатнє число , години; Атрибут second – ціле додатнє число, хвилини. Реалізувати метод minutes() – приведення часу в хвилини.

10 Лінійне рівняння y=Ax+B. Атрибут first – дробове число, коефіцієнтA; атрибут second –дробове число, коефіцієнтB. Реалізувати метод function() – обчислення для заданого x значення функції y.

11 Лінійне рівняння y=Ax+B. Атрибут first – дробове число, коефіцієнтA; атрибут second –дробове число, коефіцієнтB. Реалізувати метод root() – обчислення кореня лінійного рівняння. Метод повинен перевіряти нерівність коефіцієнта B нулю.

12 Атрибут first – дробове число, координата точки x на площині; атрибут second –дробове число, координата точки y на площині. Реалізувати метод distance() –відстань точки від початку координат.

13 Атрибут first – дробове додатнє число ,катет a прямокутного трикутника; атрибут second – дробове додатнє число, катет b прямокутного трикутника. Реалізувати метод hypotenuse()- обчислення гіпотенузи.

14 Атрибут first – дробове додатнє число , оклад; атрибут second – ціле число, кількість відпрацьованих днів в місяці. Реалізувати метод summa()- обчислення начисленої суми за дану кількість днів для заданого місяця:

Оклад/дні_місяця*відпрацьовані_дні.

15 Атрибут first – ціле додатнє число, тривалість телефонної розмови в хвилинах;атрибут second – дробове додатнє число, вартість однієї хвилини в гривнях. Реалізувати метод cost() – обчислення загальної вартості розмови.

16 Атрибут first – дробове число , ціла частина числа; атрибут second – дробове додатнє число, дробова частина числа. Реалізувати метод Multiply()- множення на довільне дробне число типу double. Метод повинен правильно працювати при любих допустимих значеннях fist і second.

17 Атрибут first – ціле додатнє число, координата курсора/вказівника по горизонталі; Атрибут second – ціле додатнє число, координата курсора по вертикалі. Реалізувати метод changex() – зміна горизонтальної координати курсора; реалізувати метод changey() – зміна вертикальної координати курсора. Методи повинні перевіряти вихід за межі екрану.

Оформлення звіту:

1. Титульний лист.
2. Мета роботи.
3. Для кожного виконаного завдання:
   - текст завдання згідно варіант;
   - код програми;
   - знімок вікна виконання програми.

4. Висновки.

Контрольні питання

1. Поняття класу та структури.
2. Оголошення та визначення класу.
3. Методи класу. Inline-функції.
4. Ініціалізація об’єктів.
5. Вкладеність класів.
6. Композиція об’єктів.