**Лабораторна робота №08-1.** **Розробка програм із використанням простого спадкування**

**Мета:** Набуття навичок в розробці програм, де використовується спадкування.

**Порядок виконання роботи**

1. Повторити лекцію №15.
2. Ознайомитися з теоретичними відомостями.
3. Продовжити працювати над програмою минулої лабораторної роботи за темою своєї курсової роботи.
   1. Створити для обраного класу вашої курсової через просте спадкування похідний клас, в якому створити:
4. члени-дані : курс, спеціальність, поточна кількість балів по предметах фізика, математика, програмування;
5. три конструктори (порожній, ініціалізації, копіювання) та деструктор, в яких поставити виведення повідомлення для моніторингу їх роботи;
6. селектори та модифікатори до всіх членів похідного класу,
7. метод введення даних в інтерактивному режимі (використати успадкований від базового класу, якщо такого в базовому класі не було – записати),
8. метод виведення даних на консоль (використати успадкований від базового класу, якщо такого в базовому класі не було – записати);
9. дружні функції 1) виведення даних, попередньо введених в інтерактивному режимі, до файлу в двох режимах: формування нового файлу та дозапису до існуючого; 2) введення даних з файлу та заповнення масиву об’єктів..
   1. В головній програмі (функції main()):
10. включити виклик нових методів та функцій до відповідних позицій меню;
11. створити позицію меню "The state of the objects of the *ххх* class" (Стан об'єктів класу *ххх – обраного класу*), де виводити інформацію про кількість об’єктів та стан їхньої обробки.
12. Виведення до файлу та введення даних з файлу щодо студентів виконувати за відпрацьованими алгоритмами на ЛР№№3-4, а саме: інформація записується до файлу, де дані атрибутів відділяємо знаком ";", а кінець позначаємо знаком "#", після якого записуємо поточну дату у вигляді РРММДД (де РР – дві останні цифри року, ММ – номер місяця, ДД – номер дня). Відповідно, введені дані потім потрібно буде аналізувати та "розбирати" на складові, щоб присвоїти значення відповідним членам класу. На цьому етапі ім’я файлу надаєте літералом у форматі FS\_**<Прізвищеанглійською>.tхt.**
13. Для виведення даних класу *похідного класу* створити позицію меню "Formation of file objects *ххх*" (Формування файлу об’єктів *ххх – обраного класу*), а для введення – "Read the *ххх* object file" (Читання файлу об’єктів *ххх – обраного класу*).
14. Встановити аналіз виклику нових функцій, зокрема, виконувати виведення в разі наявності заповнених об’єктів класу *ххх – обраного класу*.
15. Якщо в курсовій не передбачається зв‘язок спадкування, натомість зв‘язок композиції, то скласти свій власний приклад спадкування із зазначеною вище функціональністю.
16. В першому рядку програми та заголовкового файлу повинні бути записаними в коментарі номер групи та прізвище, а також номер ЛР (через кому до попередньої).
17. Результати (програма, заголовковий файл, файл з виведеними даними) надсилати на електронну адресу викладача

[**t.i.lumpova@gmail.com**](mailto:t.i.lumpova@gmail.com)у вигляді заголовкового файлу та cpp-файлу з іменем у форматі

Для заголовкового файлу

**<Прізвище англійською>.h**

Наприклад, Ivanov.h

Для cpp-файлу:

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

Наприклад, 31-01Ivanov.cpp.

Тему в заголовку листа записати

**ООП<Номер групи>-ЛР <Номер лабораторної>-<Прізвищеанглійською>**

**Строк відсилки ЛР ІПЗ-31 15.03.2024**

**ІПЗ-32 16.03.2024**

Всі запитання, що виникнуть, надсилайте на електронну адресу викладача, тему в заголовку листа записати

**ООП<Номер групи>-Запитання-<Прізвище>**.

**Теоретичні відомості.**

**Підрахунок створених об'єктів**

Припустимо, нам необхідно здійснювати підрахунок об'єктів певного класу. Можна створити лічильник – статичний елемент даних, в якому зберігатиметься кількість об'єктів, присутніх у пам'яті. Збільшення значення лічильника здійснюємо в конструкторі, а зменшення – у деструкторі. Програма може мати такий вигляд:

#include <iostream>

#include <Windows.h>

**using** std::cout;

**using** std::endl;

**class** ObjectCount

{

**private**:

**static** **int** count;

**public**:

**static** **int** getCount()

{

**return** count;

}

ObjectCount()

{

count++;

}

~ObjectCount()

{

count--;

}

};

// Статичний елемент даних слід визначити й ініціалізувати поза межами класу:

**int** ObjectCount::count = 0;

**void** main()

{ system("color F0");

ObjectCount c1;

cout << c1.getCount() << endl; // 1

ObjectCount \*p1 = &c1; // копіюємо адресу, конструктор не викликається

cout << p1->getCount() << endl; // 1

ObjectCount \*p2 = **new** ObjectCount();

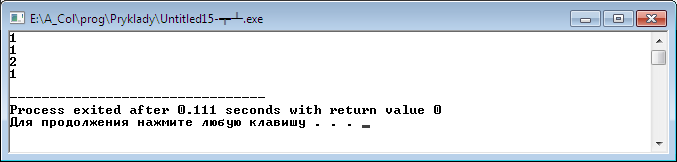
cout << p2->getCount() << endl; // 2

**delete** p2;

cout << p2->getCount() << endl; // 1

}

Результат роботи



Як видно з результатів виконання функції main(), функцію getCount() можна викликати для об'єкта навіть після того, як він був видалений з динамічної пам'яті. Це пов'язано з тим, що під час виклику статичної функції для компілятора важливим є лише тип вказівника, а не об'єкт, на який він указує. Більш коректним є виклик функції через ім'я класу:

**void** main()

{

ObjectCount c1;

cout << ObjectCount::getCount() << endl; // 1

ObjectCount \*p1 = &c1;

cout << ObjectCount::getCount() << endl; // 1

ObjectCount \*p2 = **new** ObjectCount();

cout << ObjectCount::getCount() << endl; // 2

**delete** p2;

cout << ObjectCount::getCount() << endl; // 1

}

**Синтаксис наслідування**

**class Base**

**{**

**//оголошення базового класу**

**};**

**class Derived : специфікатор\_доступу Base [, специфікатор\_доступу Base2, … ]**

**{**

**//оголошення класу нащадка**

**};**

**Специфікатори доступу**

За замовчуванням для класів використовується специфікатор доступу ***private,*** для структур ***-***  ***public***.

Для будь-якого члену класу може також застосовуватися специфікатор доступу ***protected,*** який для одиночних класів, що не входять до ієрархії, аналогічний до ***private*** . Різниця виявляється при спадкуванні, що демонструє наступна таблиця.

|  |  |  |
| --- | --- | --- |
| Специфікатор доступу | | |
| Базовий клас | В базовому класі | Доступ в класі-нащадку |
| ***private*** | ***private*** | відсутній |
| ***protected*** | ***private*** |
| ***public*** | ***private*** |
| ***protected*** | ***private*** | відсутній |
| ***protected*** | ***protected*** |
| ***public*** | ***protected*** |
| ***public*** | ***private*** | відсутній |
| ***protected*** | ***protected*** |
| ***public*** | ***public*** |

**Приклад передачі параметрів конструкторам базових класів при простому спадкуванні**

#include <stdio.h>

#include <Windows.h>

using namespace std;

class TInner

{

public:

int i;

TInner(int n):i(n){printf("Ctor Inner\n");}

TInner(TInner& x){ \*this = x; printf("Copy ctor TInner\n");}

~TInner(){printf("Dtor TInner\n");}

};

class TBase1

{

public:

double a;

***TInner b;***

TBase1(int x, double y):b(x),a(y){printf("Ctor TBase1\n");}

TBase1(TBase1& x):b(10){ \*this = x; printf("Copy ctor TBase1");}

~TBase1(){printf("Dtor TBase1\n");}

void printBase() {printf("TBase1::TInner::i = %d a = %lf\n",b.i, a);}

};

class TBase2

{

public:

char c;

TBase2(char x):c(x){printf("Ctor TBase2\n");}

TBase2(TBase2& x){\*this = x; printf("Copy ctor TBase2");}

~TBase2(){printf("Dtor TBase2\n");}

void printBase2() {printf("TBase2::c = c = %c\n",c);}

};

class TDerived: public TBase1, public TBase2

{

public:

float f;

**TDerived(int x, double y, char z, float w):TBase1(x, y),TBase2('z')**

{ f = w; printf("Ctor TDerived\n"); }

~TDerived(){printf("Dtor TDerived\n");}

void printDerived2() {printf("TDerived2::%lf\n",f);}

};

int main()

{ system("color F0");

**TDerived obj(1.0, 2, 'Y', 3.0);**

printf("TBase1::a = %lf \n",obj.a);

printf("TBase1::b = %d \n",obj.b.i);

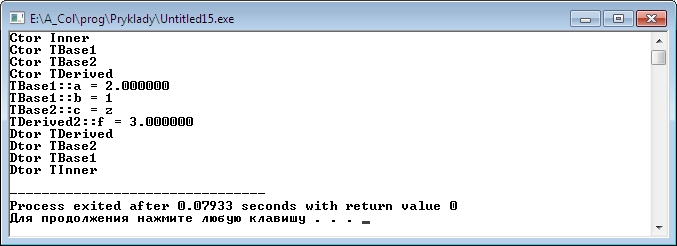
printf("TBase2::c = %c \n",obj.c);

printf("TDerived2::f = %lf \n",obj.f);

return 0;

}

Результат роботи



***Контрольні запитання*.**

1. Наведіть приклад відношення узагальнення.
2. Визначте основні концепції наслідування.
3. В чому полягає принцип підстановки?
4. Які форми наслідування Ви можете визначити?
5. В чому полягає просте спадкування?
6. Як визначається поведінка конструкторів та деструкторів при спадкуванні?
7. [Скільки “дружніх” функцій та “дружніх” класів можна оголошувати у тілі класу?](https://www.bestprog.net/uk/2018/07/21/friends-of-the-class-friendly-classes-and-functions-the-keyword-friend-examples_ua/#q05)
8. [Як отримати об’єкт потрібного класу у “дружній” функції, щоб мати доступ до усіх його членів?](https://www.bestprog.net/uk/2018/07/21/friends-of-the-class-friendly-classes-and-functions-the-keyword-friend-examples_ua/#q06)
9. Що таке вкладений клас і які його особливості?