Л а б о р а т о р н а р о б о т а № 3

Тема: Успадкування і віртуальні функції.

Мета роботи: Одержати практичні навички створення ієрархії класів і використання статичних компонентів класу.

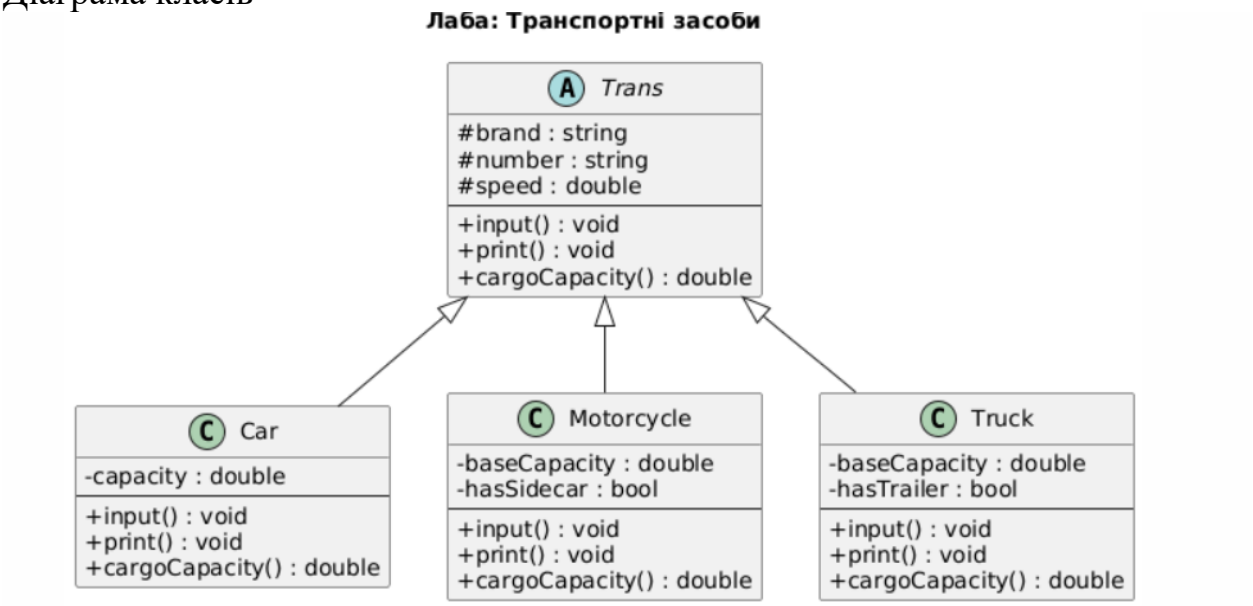
Варіант 24

Задача 1(9)

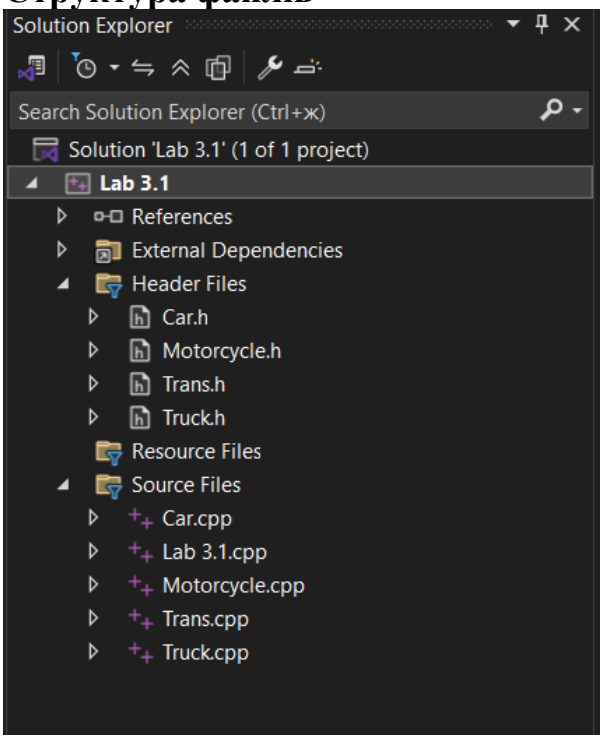
Умова:

1. Створити абстрактний клас Клієнт з методами, що дозволяють вивести на екран інформацію про клієнтів банку, а також визначити відповідність клієнта критерію пошуку.
2. Створити похідні класи: Вкладник (прізвище, дата відкриття внеску, розмір внеску, відсоток по внеску), Кредитор (прізвище, дата видачі кредиту, розмір кредиту, відсоток по кредиту, залишок боргу), Організація (назва, дата відкриття рахунку, номер рахунку, сума на рахунку) з своїми методами виведення інформації на екран, і визначення відповідності даті (відкриття внеску, видачі кредиту, відкриття рахунку).
3. Створити базу (масив) з n клієнтів, вивести повну інформацію з бази на екран, а також організувати пошук клієнтів, що почали співробітничати з банком в задану дату.

Діаграма класів



Структура файлів



Код:

1)Опис класу, Trans.h

```
#pragma once
#include <string>
```

```
using namespace std;

class Trans
{
protected:
	string brand;
	string number;
	double speed;
public:
	Trans();
	virtual void input();
	virtual void print();
	virtual double cargoCapacity() = 0;
	virtual ~Trans();
};
```

Trans.cpp

```
#include "Trans.h"
#include <iostream>
#include <string>

using namespace std;

Trans::Trans() {
	brand = "";
	number = "";
	speed = 0;
}

void Trans::input(){
	cout << " Brand: ";
	getline(cin, brand);
	cout << " Number: ";
	getline(cin, number);
	cout << " Speed: ";
	cin >> speed;
}

void Trans::print() {
	cout << "\nBrand: " << brand << "\nNumber: " << number << "\nSpeed: " << speed
		<< " km per hour ";
}

Trans::~Trans(){}
```

2)Car.h

```
#pragma once
#include "Trans.h"
#include <iostream>
using namespace std;
class Car:public Trans
{
	double capacity;
public:
	Car();
	void input() override;
	void print() override;
	double cargoCapacity();
	~Car();
};
```

Car.cpp

```
#include "Car.h"
Car::Car() { capacity = 0; }

void Car::input() {
	cout << "\n Passenger car \n";
	Trans::input();
	cout << "Load capacity: ";
	cin >> capacity;
	cin.ignore(1000, '\n');
}
void Car::print() {
	cout << " Type: passenger car";
	Trans::print();
	cout<< "\nCapacity: " << capacity << " kilo " << endl;
}
double Car::cargoCapacity() { return capacity; }
Car::~Car(){}
```

3)Motorcycle.h

```
#pragma once
#include "Trans.h"
#include <iostream>
using namespace std;
class Motorcycle:public Trans
{
	double baseCapacity;
	int hasSidecar;
public:
```

```
        Motorcycle();
        void input() override;
        void print() override;
        double cargoCapacity();
        ~Motorcycle();
};
```

Motorcycle.cpp

```
#include "Motorcycle.h"
Motorcycle::Motorcycle() { baseCapacity = 0; hasSidecar = 0; }

void Motorcycle::input() {
        cout << "\n Motorcycle \n";
        Trans::input();
        cout << " Do it have sideCar?(1-yes;2-no): ";
        cin >> hasSidecar;
        cout << " Capacity with sideCar: ";
        cin >> baseCapacity;
        cin.ignore(1000, '\n');
}
void Motorcycle::print() {
        cout << "Type: motocycle";
        Trans::print();
        if (hasSidecar == 1) cout << " sideCar: yes";
        else cout << " sideCar: no";
        cout << "\nCapacity: " << cargoCapacity() << " kilo " << endl;
}
double Motorcycle::cargoCapacity() {
        if (hasSidecar == 1) return baseCapacity;
        else return 0;
}
Motorcycle::~Motorcycle(){}
```

4) **Truck.h**

```
#pragma once
#include "Trans.h"
#include <iostream>
using namespace std;
class Truck : public Trans
{
   double baseCapacity;
   int hasTrailer;
public:
   Truck();
   void input() override;
   void print() override;
```

```
    double cargoCapacity();
    ~Truck();
};
```

Truck.cpp

```
#include "Truck.h"
Truck::Truck(){baseCapacity = 0;hasTrailer = 0;}
void Truck::input() {
        cout << "\n Truck \n";
    Trans::input();
    cout << "Do it have trailer?(1-yes;2-no): ";
    cin >> hasTrailer;
    cout << "Base capacity: ";
    cin >> baseCapacity;
    cin.ignore(10000, '\n');
}
void Truck::print() {
    cout << "Type: Truck";
    Trans::print();
    if (hasTrailer == 1) cout << " Trailer:yes ";
    else cout << " Trailer:no ";
    cout << "\nCapacity: " << cargoCapacity() << " kilo " << endl;
}
double Truck::cargoCapacity()
{
    if (hasTrailer == 1) return baseCapacity * 2;
    else return baseCapacity;
}

Truck::~Truck(){}
```

5) Main.cpp

```
#include <iostream>
#include "Trans.h"
#include "Car.h"
#include "Motorcycle.h"
#include "Truck.h"
using namespace std;
int main() {
        cout << " -Base of transport- " << endl;
        Trans* db[3];
    for (int i = 0; i < 3; i++)
    {
        cout << "\nChose the type of transport #" << i + 1 << endl;
        cout << "1 - Passenger car" << endl;
        cout << "2 - Motorcycle" << endl;
```

```
        cout << "3 - Truck" << endl;
        cout << "Your choice: ";
        int t;
        cin >> t;
        cin.ignore(10000, '\n');
        if (t == 1) db[i] = new Car;
        else if (t == 2) db[i] = new Motorcycle;
        else db[i] = new Truck;
        db[i]->input();
    }
    cout << "\n -All transport- " << endl;
    for (int i = 0; i < 3; i++) db[i]->print();
    cout << "\n Min car Capacity: ";
    double need;
    cin >> need;
    cout << "\nResults" << endl;
    bool found = false;
    for (int i = 0; i < 3; i++)
    {
        if (db[i]->cargoCapacity() >= need)
        {
            db[i]->print();
            found = true;
        }
    }

    if (!found)
        cout << "None found!" << endl;

    for (int i = 0; i < 3; i++)
    {
        delete db[i];
    }
}
```

РЕЗУЛЬТАТ ПРОГРАМИ (внизу)

```
 -Base of transport-

Chose the type of transport #1
1 - Passenger car
2 - Motorcycle
3 - Truck
Your choice: 1

 Passenger car
 Brand: Audi
 Number: 1568
 Speed: 220
Load capacity: 25000

Chose the type of transport #2
1 - Passenger car
2 - Motorcycle
3 - Truck
Your choice: 2

 Motorcycle
 Brand: Kawasaki
 Number: 9847
 Speed: 330
 Do it have sideCar?(1-yes;2-no): 1
 Capacity with sideCar: 800

Chose the type of transport #3
1 - Passenger car
2 - Motorcycle
3 - Truck
Your choice: 3

 Truck
 Brand: Mersedes
 Number: 6574
 Speed: 120
Do it have trailer?(1-yes;2-no): 1
Base capacity: 87000
```

```
 -All transport-
 Type: passenger car
Brand: Audi
Number: 1568
Speed: 220 km per hour
Capacity: 25000 kilo
Type: motocycle
Brand: Kawasaki
Number: 9847
Speed: 330 km per hour  sideCar: yes
Capacity: 800 kilo
Type: Truck
Brand: Mersedes
Number: 6574
Speed: 120 km per hour  Trailer:yes
Capacity: 174000 kilo

 Min car Capacity: 25000

Results
 Type: passenger car
Brand: Audi
Number: 1568
Speed: 220 km per hour
Capacity: 25000 kilo
Type: Truck
Brand: Mersedes
Number: 6574
Speed: 120 km per hour  Trailer:yes
Capacity: 174000 kilo
```

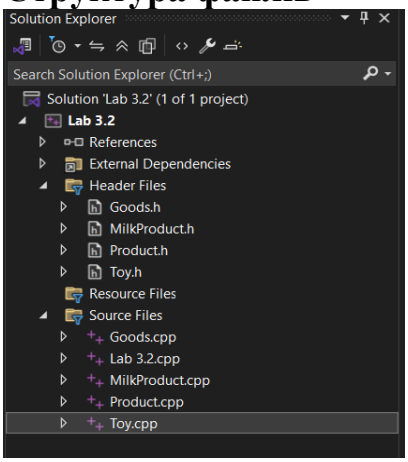
Задача 2(9)

Умова:

Дано перелік класів. Побудуйте обєктну модель предметної області, враховуючи, види залежностей між класами. Реалізуйте проект за ООП.

9. іграшка, продукт, товар, молочний продукт

Структура файлів



Діаграма класів

1)Goods.h

```
#pragma once
#include <string>
#include <iostream>
using namespace std;
class Goods
{
protected:
    string name;
    double price;
public:
    Goods();
    virtual void input();
    virtual void print();
    virtual ~Goods();
};
```

Goods.cpp

```
#include "Goods.h"

Goods::Goods()
{
    name = "";
    price = 0;
```

```
}

void Goods::input()
{
   cout << "Name of the product: ";
   getline(cin, name);
   cout << "Price: ";
   cin >> price;
   cin.ignore(10000, '\n');
}

void Goods::print()
{
   cout << "Name: " << name << "\nPrice: " << price << " UAH";
}
Goods::~Goods(){}
```

2)**Product.h**

```
#pragma once
#include "Goods.h"
#include <iostream>
using namespace std;

class Product : public Goods
{
protected:
   string expirationDate;

public:
   Product();
   void input();
   void print();
   ~Product();
};
```

Product.cpp

```
#include "Product.h"
Product::Product() : Goods(){expirationDate = "";}

void Product::input()
{
   Goods::input();
   cout << "Expiration date: ";
   getline(cin, expirationDate);
}

void Product::print()
```

```
{
   Goods::print();
   cout << "\nExpiration date: " << expirationDate;
}

Product::~Product(){}
```

3)**MilkProduct.h**

```
#pragma once
#include "Product.h"
#include <iostream>
using namespace std;

class MilkProduct : public Product
{
protected:
   double fat;

public:
   MilkProduct();
   void input();
   void print();
   ~MilkProduct();
};
```

MilkProduct.cpp

```
#include "MilkProduct.h"

MilkProduct::MilkProduct() : Product() {fat = 0;}

void MilkProduct::input()
{
   Product::input();
   cout << "Enter fat in per cent: "; cin >> fat;
   cin.ignore(10000, '\n');
}

void MilkProduct::print()
{
   Product::print();
   cout << "\nFat: " << fat << "%" << endl;
}

MilkProduct::~MilkProduct(){}
```

4)**Toy.h**

```
#pragma once
```

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

class Toy : public Goods
{
protected:
   string material;

public:
   Toy();
   void input();
   void print();
   ~Toy();
};
```

Toy.cpp

```
#include "Toy.h"

Toy::Toy() : Goods(){material = "";}

void Toy::input()
{
   Goods::input();
   cout << "Material: ";
   getline(cin, material);
}

void Toy::print()
{
   Goods::print();
   cout << "\nMAterial: " << material << endl;
}

Toy::~Toy(){}
```

5)**Main.cpp**

```
#include "Goods.h"
#include "Product.h"
#include "MilkProduct.h"
#include "Toy.h"
using namespace std;

int main()
{
   cout << "--Hierarchy--" << endl;
   Goods* db[3];
   for (int i = 0; i < 3; i++)
```

```
    {
        cout << "\nChoose type of product #" << i + 1 << endl;
        cout << "1 - Goods" << endl;
        cout << "2 - Product" << endl;
        cout << "3 - Milk product" << endl;
        cout << "4 - Toy" << endl;
        cout << "Your choise: ";
        int t;
        cin >> t;
        cin.ignore(10000, '\n');
        if (t == 1)db[i] = new Goods;
        else if (t == 2)db[i] = new Product;
        else if (t == 3)db[i] = new MilkProduct;
        else db[i] = new Toy;

        cout << "\nEntering info about object #" << i + 1 << endl;
        db[i]->input();
    }

    cout << "\nAll products" << endl;
    for (int i = 0; i < 3; i++)
    {
        db[i]->print();
        cout << endl;
    }

    for (int i = 0; i < 3; i++)
        delete db[i];
}
```

РЕЗУЛЬТАТ ПРОГРАМИ

```
--Hierarchy--

Choose type of product #1
1 - Goods
2 - Product
3 - Milk product
4 - Toy
Your choise: 1

Entering info about object #1
Name of the product: Metal
Price: 40

Choose type of product #2
1 - Goods
2 - Product
3 - Milk product
4 - Toy
Your choise: 2

Entering info about object #2
Name of the product: Bread
Price: 25
Expiration date: 18.10.25

Choose type of product #3
1 - Goods
2 - Product
3 - Milk product
4 - Toy
Your choise: 3

Entering info about object #3
Name of the product: Milk
Price: 55
Expiration date: 25.11.25
Enter fat in per cent: 5

All products
Name: Metal
Price: 40 UAH
Name: Bread
Price: 25 UAH
Expiration date: 18.10.25
Name: Milk
Price: 55 UAH
Expiration date: 25.11.25
Fat: 5%
```