Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут прикладної математики та фундаментальних наук

Кафедра прикладної математики

**Звіт**

про виконання лабораторної роботи № 5

з курсу «Алгоритми та програмування, ч.2»

**Виконав**:

студент групи ФІ-11

Діяк І.В.

**Перевірила:**

Рижа І.А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата) (підпис викладача)

Львів-2023 р.

**Варіант 6 (Сховище)**

**Тема роботи: Просте наслідування на С++**

**Перелік ідентифікаторів:**

**Shelter{}** –  **клас Сховище.**

**Item{} – клас Предмет.**

**Electronic{} – породжений клас від предмета.**

**Product{} – породжений клас від предмета.**

**Fruits{} – породжений клас від продукта.**

**Temp\_product – поле класу продукта.**

**County\_origin – поле класу фрукта.**

**Graduation\_year – поле класу електроніки.**

**Name** – **поле класу Сховище.**

Type – **поле класу Сховище.**

**\*\* item** – масив вказівників**.**

**Size** – розмір масиву.

Counter – статичне поле класу.

**Shelter() – конструктор без параметрів.**

**~Shelter() – деструктор.**

**~Item() – віртуальний деструктор.**

**Storage() – перевантажений метод у класі сховище.**

**Friend operator>>(istream) – дружній перевантажений оператор вводу.**

**Friend operator<<(ostream) – дружній перевантажений оператор виводу**

**PrintName() – віртуальний метод, що встановлює тип предмету.**

**F\_print\_item() – віртуальний поліморфний метод друку.**

**Код програми** **(**Написано мовою C++). Програма написання— Visual Studio 2022.

#pragma once

#include<iostream>

#include<fstream>

#include<string.h>

#pragma warning (disable:4996)

using namespace std;

const int size1 = 30;

class Item {

static int counter;

int number\_item;

protected:

char name[size1];

public:

Item();

Item(const char\* \_name);

Item(const char\* \_name, int \_number\_item);

virtual ~Item();

bool set\_item(const char\* \_name, int \_number\_item);

bool set\_name(const char\* \_name);

bool set\_number\_item(int \_number\_item);

inline const char\* get\_name() const

{

return name;

}

inline int get\_number\_item() const

{

return number\_item;

}

static void set\_counter(int seed) { counter = seed; }

virtual void PrintName(ostream& out)const = 0;

virtual void print\_item() const;

virtual void f\_print\_item(ostream& out) const;

friend istream& operator>>(istream& in, Item& item);

friend ostream& operator<<(ostream& out, const Item& item);

};

// породжений клас

class Product : public Item {

private:

double temp\_product;

public:

Product() : Item(),temp\_product(0.0) {}

Product(const char\* \_name) : Item(\_name), temp\_product(0.0) {}

Product(const char\* \_name, int \_number\_item) : Item(\_name, \_number\_item), temp\_product(0.0) {}

Product(const char\* \_name, int \_number\_item, double \_temp\_product) : Item(\_name, \_number\_item), temp\_product(\_temp\_product) {}

Product(const Product& \_item) : Item(\_item), temp\_product(\_item.temp\_product) {};

virtual ~Product();

bool set\_item(const char\* \_name, int \_number\_item, double \_temp\_product);

void set\_temp\_product(double \_temp\_product);

inline double get\_temp\_product() const

{

return temp\_product;

}

virtual void PrintName(ostream& out)const override { out << "Product" << ' '; }

virtual void print\_item() const;

virtual void f\_print\_item(ostream& out) const;

};

// породжений клас

class Electronic : public Item {

private:

int graduation\_year;

public:

Electronic() : Item(), graduation\_year(0) {}

Electronic(const char\* \_name) : Item(\_name), graduation\_year(0.0) {}

Electronic(const char\* \_name, int \_number\_item) : Item(\_name, \_number\_item), graduation\_year(0.0) {}

Electronic(const Electronic& \_item) : Item(\_item), graduation\_year(\_item.graduation\_year) {};

Electronic(const char\* \_name, int \_number\_item, double \_graduation\_year) :Item(\_name, \_number\_item), graduation\_year(\_graduation\_year) {}

virtual ~Electronic();

bool set\_graduation\_year(int \_graduation\_year);

inline int get\_graduation\_year() const

{

return graduation\_year;

}

virtual void PrintName(ostream& out)const override { out << "Electronic" << ' '; }

virtual void print\_item() const;

virtual void f\_print\_item(ostream& out) const;

};

class Fruits : public Product {

private:

char country\_origin[size1];

public:

Fruits() : Product(), country\_origin("NULL") {}

Fruits(const char\* \_name) : Product(\_name), country\_origin("NULL") {}

Fruits(const char\* \_name, int \_number\_item) : Product(\_name, \_number\_item), country\_origin("NULL") {}

virtual ~Fruits() { cout << "Fruits destroyed"; }

virtual void PrintName(ostream& out) const override { out << "Fruits" << ' '; }

void set\_country\_origin(const char\* \_country\_origin) { strcpy(country\_origin, \_country\_origin); }

inline const char\* get\_country\_origin() const

{

return country\_origin;

}

virtual void print\_item() const;

virtual void f\_print\_item(ostream& out) const;

};

class Shelter {

Item\*\* items;//динамічний масив вказівників

int size;

public:

char name[size1];

protected:

char type[size1];

public:

Shelter();

Shelter(const char\* \_name, const char\* \_type, Item\*\* \_items, int \_size);

Shelter(const Shelter& shelter);

const Shelter & operator=(const Shelter& shelter);

~Shelter();

bool set\_Shelter(const char\* \_name, const char\* \_type, Item\*\* \_items, int \_size);

bool set\_type(const char\* \_type);

bool set\_name(const char\* \_name);

bool set\_size(int \_size);

int storage(int \_number);

void storage(const char\* item\_name, const char\* \_item\_type);

void storage(const char\* item\_name, int item\_number, const char\* \_item\_type);

int get\_size()

{

return size;

}

void bubble\_sort();

void quick\_sort(int start, int end);

//функція розділення масиву

int partition(int start, int end);

friend ostream& operator<<(ostream& out, const Shelter& shelter);

friend istream& operator>>(istream& in, Shelter& shelter);

};

bool operator<(const Item& item1, const Item& item2);

bool operator<=(const Item& item1, const Item& item2);

bool operator>(const Item& item1, const Item& item2);

bool operator>=(const Item& item1, const Item& item2);

enum class menu\_items { add\_item = 1, find\_item, print\_shelter, scan\_shelter, read\_file, safe\_file, sort\_items, end };

void print\_all(Item& item1);

void Item::print\_item() const

{

cout << "Name of item: " << name << endl;

cout << "Number of item: " << number\_item << endl;

}

void Item::f\_print\_item(ostream & out) const

{

out << name << ' ';

out << number\_item << ' ';

}

void Product::f\_print\_item(ostream & out) const

{

Item::f\_print\_item(out);

cout << get\_temp\_product() << ' ';

}

void Electronic::f\_print\_item(ostream & out) const

{

Item::f\_print\_item(out);

cout << get\_graduation\_year() << ' ';

}

void Fruits::f\_print\_item(ostream & out) const

{

Product::f\_print\_item(out);

cout << get\_country\_origin() << ' ';

}

void Product::print\_item() const

{

Item::print\_item();

cout << "Temperature: " << get\_temp\_product() << endl;

}

void Electronic::print\_item() const

{

Item::print\_item();

cout << "Graduation Year: " << get\_graduation\_year() << endl;

}

void Fruits::print\_item() const

{

Product::print\_item();

cout << "Country Origin: " << get\_country\_origin() << endl;

}

int main()

{

menu\_items menu\_item;

int choice;

char item\_name[size1];

int number\_item, max\_number\_item = 0;

char shelter\_name[size1];

char shelter\_type[size1];

char item\_type[size1];

Shelter shelter;

do

{

cout << "\n\_\_Menu\_\_" << endl;

cout << "1. Add Item" << endl;

cout << "2. Find Item" << endl;

cout << "3. Print Shelter" << endl;

cout << "4. Scan Shelter" << endl;

cout << "5. Read from file" << endl;

cout << "6. Save file" << endl;

cout << "7. Sort Items" << endl;

cout << "8. Exit" << endl << endl;

cout << "enter menu number: ";

cin >> choice;

cin.ignore();

menu\_item = (menu\_items)choice;

system("cls");

switch (menu\_item)

{

case menu\_items::add\_item:

cout << "enter name of Item: ";

cin.getline(item\_name, size1);

cout << "enter type of Item (Product, Electronic, Fruits): ";

cin.getline(item\_type, size1);

shelter.storage(item\_name,item\_type);

cout << "successfully" << endl;

break;

case menu\_items::find\_item:

cout << "enter number of item: ";

cin >> number\_item;

cin.ignore();

cout << "position Item in shelter is " << shelter.storage(number\_item) + 1 << endl;

break;

case menu\_items::print\_shelter:

cout << shelter;

break;

case menu\_items::scan\_shelter:

cin >> shelter;

break;

case menu\_items::read\_file:

{

ifstream fin("shelter.txt");

if (!fin.is\_open())

{

cerr << "file not found" << endl;

break;

}

while (!fin.eof())

{

fin >> shelter;

}

fin.close();

cout << "File read successfully" << endl;

}

break;;

case menu\_items::safe\_file:

{

ofstream fout("Shelter.txt");

if (!fout.is\_open())

{

cerr << "file not saved" << endl;

break;

}

fout << shelter;

fout.close();

cout << "File saved successfully" << endl;

}

break;

case menu\_items::sort\_items:

shelter.quick\_sort(0, shelter.get\_size() - 1);

cout << shelter << endl;

break;

case menu\_items::end:

cout << "end" << endl;

exit(0);

break;

default:

cerr << "incorrect menu number" << endl;

break;

}

system("pause");

system("cls");

} while (menu\_item != menu\_items::end);

cout << "Ends of program" << endl;

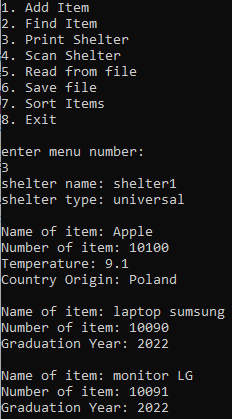
system("pause");

system("cls");

return 0;

}

**Контрольний приклад:**



**Висновок:** на цій лабораторній я навчився працювати з простим наслідуванням, будувати ієрархію класів і описувати поліморфні функції на С++.