**Міністерство освіти і науки України**

**Національний університет “Львівська політехніка”**

Кафедра ЕОМ



**Курсова робота**

з предмету: «Програмування, частина 2 (Об’єктно-орієнтованепрограмування)»

на тему:

**«Базові принципи об’єктно-орієнтованогопрограмування»**

Індивідуальне завдання:«Навчальний заклад»

*Виконано згідно методичних матеріалів*

**Виконав:**

ст. гр. КІ-15

Когутич В.М.

**Прийняла:**

асистент

Козак Н.Б.

Львів-2020

**Анотація**

Для розв’язання багатьох практичних задач з різних галузей, необхідно використовувати базові принципи об’єктно-орієнтованого програмування, оскільки вони можуть адаптувати вирішення поставлених задач до їх швидкої зміни в реальному часі. Одним з шляхів забезпечення таких вимог є організація об’єктно-орієнтованих процесів і відповідних технічних засобів їх реалізації.

Причому, ефективність застосуванняоб’єктно-орієнтованого програмування у вирішенні поставленого завдання залежить, як від продуктивності комп’ютерів, так і від розмірів і структури пам’яті, пропускної здатності каналів зв’язку,

використанихоб’єктно-орієнтованихмов програмування, компіляторів, операційних систем, чисельних методів та інших математичних досліджень. Такий широкий обсяг параметрів вимагає проведення досліджень на різних рівнях: на рівні опису алгоритмів, створення спеціальних мов програмування, компіляторів, багатопроцесорних систем, неоднорідних систем, кластерів.

Для скорочення термінів розробки об’єктно-орієнтованих програм та дослідження їх роботи використовується моделювання.

***Метою виконання курсової роботи*** є засвоєння основних методів таалгоритмів об’єктно-орієнтованихструктур, принципів побудови відповідних структур, набуття початкових практичних навиків проектування таких засобів.

ЗМІСТ

[Завдання на курсову роботу 4](#_Toc41704076)

[1. Огляд та обґрунтовування вибору технологій об'єктно-орієнтованого програмування. 5](#_Toc41704077)

[2.Аналіз та розробка алгоритму згідно індивідуального завдання. 7](#_Toc41704078)

[3.Опис реалізації основних функцій/методів: 9](#_Toc41704079)

[4.Аналіз продуктивності (часу роботи) та структурної складності реалізованої програмзгідно індивідуального завдання. 15](#_Toc41704080)

[Висновки 16](#_Toc41704081)

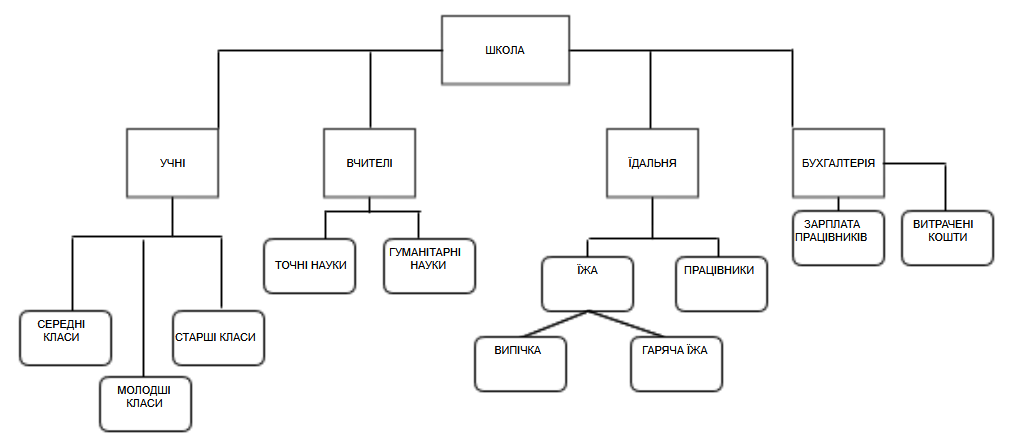
[Список літератури 17](#_Toc41704082)

[Додатки 18](#_Toc41704083)

# Завдання на курсову роботу

**Індивідуальне завдання: «Навчальний заклад».**

Згідно із отриманим індивідуальним завданням метою курсової роботи є реалізація мовою С++ із застосуванням об’єктно-орієнтованого підходу предметної галузі “Навчальний заклад”. Програма повинна підтримувати можливості роботи з файлами: запис, зчитування.



Структура предментої області

«Навчальний заклад»

# Огляд та обґрунтовування вибору технологій об'єктно-орієнтованого програмування.

**Абстракція** - в об'єктно-орієнтованому програмуванні це додання об'єкту характеристик, які відрізняють його від всіх інших об'єктів, чітко визначаючи його концептуальні межі. Основна ідея полягає в тому, щоб відокремити спосіб використання складових об'єктів даних від деталей їх реалізації у вигляді більш простих об'єктів, подібно до того, як функціональна абстракція розділяє спосіб використання функції і деталей її реалізації в термінах більш примітивних функцій, таким чином, дані обробляються функцією високого рівня за допомогою виклику функцій низького рівня. Такий підхід є основою об'єктно-орієнтованого програмування. Це дозволяє працювати з об'єктами, не вдаючись у особливості їх реалізації. У кожному конкретному випадку застосовується той чи інший підхід: інкапсуляція, поліморфізм або успадкування. Наприклад, при необхідності звернутися до прихованих даними об'єкта, слід скористатися інкапсуляцією, створивши, так звану, функцію доступу або властивість. Абстракція даних - популярна і в загальному невірно визначається техніка програмування. Фундаментальна ідея полягає в поділі несуттєвих деталей реалізації підпрограми і характеристик істотних для коректного її використання. Такий поділ може бути виражено через спеціальний «інтерфейс», зосереджують опис всіх можливих застосувань програми. З точки зору теорії множин, процес являє собою організацію для групи підмножин своєї множини.

**Інкапсуляція** - властивість мови програмування, що дозволяє користувачеві не замислюватися про складність реалізації використовуваного програмного компонента (що у нього всередині?), а взаємодіяти з ним за допомогою наданого інтерфейсу (публічних методів і членів), а також об'єднати і захистити життєво важливі для компонента дані. При цьому користувачеві надається тільки специфікація (інтерфейс) об'єкта. Користувач може взаємодіяти з об'єктом тільки через цей інтерфейс. Реалізується за допомогою ключового слова: public. Користувач не може використовувати закриті дані і методи. Реалізується за допомогою ключових слів: private, protected, internal. Інкапсуляція - один з чотирьох найважливіших механізмів об'єктно-орієнтованого програмування (поряд з абстракцією, поліморфізмом і спадкуванням). Приховування реалізації доцільно застосовувати в наступних випадках: гранична локалізація змін при необхідності таких змін, прогнозованість змін (які зміни в коді треба зробити для заданого зміни функціональності) і прогнозованість наслідків змін.

**Спадкування** - один з чотирьох найважливіших механізмів об'єктно-орієнтованого програмування (поряд з інкапсуляцією, поліморфізмом і абстракцією), що дозволяє описати новий клас на основі вже існуючого (батьківського), при цьому властивості і функціональність батьківського класу запозичуються новим класом. Іншими словами, клас-спадкоємець реалізує специфікацію вже існуючого класу (базовий клас). Це дозволяє користуватися об'єктами класу-спадкоємця так само, як з об'єктами базового класу. Просте наслідування: клас, від якого відбулося спадкування, називається базовим або батьківським (англ. Base class). Класи, які походять від базового, називаються нащадками, спадкоємцями або похідними класами (англ. Derived class). Деякі мови використовують абстрактні класи. Абстрактний клас - це клас, що містить хоча б один абстрактний метод, він описаний в програмі, має поля, методи і не може використовуватися для безпосереднього створення об'єкта. Тобто від абстрактного класу можна тільки наслідувати. Об'єкти створюються тільки на основі похідних класів, успадкованих від абстрактного. Наприклад, абстрактним класом може бути базовий клас «співробітник вузу», від якого успадковуються класи «аспірант», «професор» і т. Д. Так як похідні класи мають спільні поля і функції (наприклад, поле «рік народження»), то ці члени класу можуть бути описані в базовому класі. У програмі створюються об'єкти на основі класів «аспірант», «професор», але немає сенсу створювати об'єкт на основі класу «співробітник вузу».

**Множинне спадкування**

При множинному спадкуванні у класу може бути більш одного предка. В цьому випадку клас успадковує методи всіх предків. Переваги такого підходу в більшій гнучкості. Множинне спадкування реалізовано в C ++. Множинне спадкування - потенційне джерело помилок, які можуть виникнути через наявність однакових імен методів в предках. У мовах, які позиціонуються як спадкоємці C ++ (Java, C # і ін.), від множинного успадкування було вирішено відмовитися на користь інтерфейсів. Практично завжди можна обійтися без використання даного механізму. Однак, якщо така необхідність все-таки виникла, то, для вирішення конфліктів використання успадкованих методів з однаковими іменами, можливо, наприклад, застосувати операцію розширення видимості - «::» - для виклику конкретного методу конкретного батька. Спроба вирішення проблеми наявності однакових імен методів в предках була зроблена в мові Ейфель, в якому при описі нового класу необхідно явно вказувати імпортовані члени кожного з успадкованих класів і їх іменування в дочірньому класі. Більшість сучасних об'єктно-орієнтованих мов програмування (C #, Java, Delphi і ін.) Підтримують можливість одночасно успадковуватися від класу-предка і реалізувати методи декількох інтерфейсів одним і тим же класом. Цей механізм дозволяє багато в чому замінити множинне спадкування - методи інтерфейсів необхідно перевизначати явно, що виключає помилки при спадкуванні функціональності однакових методів різних класів-предків.

**Поліморфізм** - можливість об'єктів з однаковою специфікацією мати різну реалізацію. Мова програмування підтримує поліморфізм, якщо класи з однаковою специфікацією можуть мати різну реалізацію - наприклад, реалізація класу може бути змінена в процесі успадкування. Коротко зміст поліморфізму можна виразити фразою: «Один інтерфейс, безліч реалізацій». Поліморфізм - один з чотирьох найважливіших механізмів об'єктно-орієнтованого програмування (поряд з абстракцією, інкапсуляцією і спадкуванням). Поліморфізм дозволяє писати більш абстрактні програми і підвищити коефіцієнт повторного використання коду. Загальні властивості об'єктів об'єднуються в систему, яку можуть називати по-різному - інтерфейс, клас. Спільність має зовнішнє і внутрішнє вираження: зовнішня спільність проявляється як однаковий набір методів з однаковими іменами та сигнатурами (ім'ям методів і типами аргументів і їх кількістю); внутрішня спільність - однакова функціональність методів. Її можна описати інтуїтивно або висловити у вигляді строгих законів, правил, яким повинні підкорятися методи. Можливість приписувати різну функціональність одному методу (функції, операції) називається перевантаженням методу (перевантаженням функцій, перевантаженням операцій).

Оцінивши всі переваги та недоліки об’єктно-орієнтованого програмування, я зробив вибір на користь мови С++.

# 2.Аналіз та розробка алгоритму згідно індивідуального завдання.

**2.1. Опис алгоритму та оцінка його складності**

В алгоритмі наяві 3 підменю, шо утворюють меню консольної програми. Перше розгалуження відповідає за головне меню програми, що включає такі можливості:

а) отримати інформацію про школу

б) уроки;

в) вчителі;

г) учні;

д) їдальня;

е) закрити меню

При виконанні переходу до пункту «отримати інформацію про школу» перед користувачем з'являється вся потрібна інформація про школу.

При виконанні переходу до пункту «уроки», користувач поптрапляє у меню «уроки»:

а) показати уроки;

б) провести урок;

в) закрити меню.

При виконанні переходу до пункту «вчителі», користувач потрапляє у меню «вчителі»:

а) показати вчителів;

б) додати вчителів;

в) звільнити вчителів;

г) закрити меню.

При виконанні переходу до пункту «учні», користувач потрапляє у меню «учні»:

а) показати учнів;

б) додати учнів;

в) забрати учнів;

г) закрити меню.

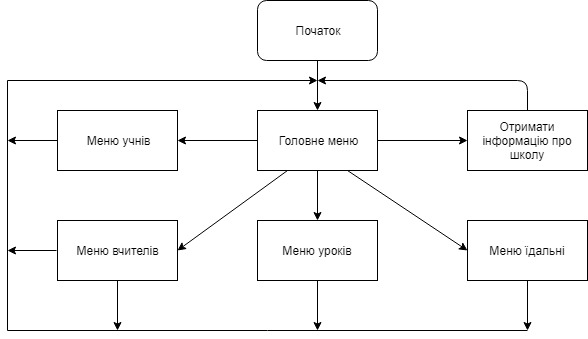
При виконанні переходу до пункту «їдальня», користувача потрапляє у меню «їдальня»:

а) показати їжу;

б) додати їжу;

в) продати їжу.

При виконанні переходу до пункту «закрити меню», користувач закриває програму.



**Рис.1 блок-схема програми**

**2.2 Програмна реалізація алгоритму згідно індивідуального завдання.**

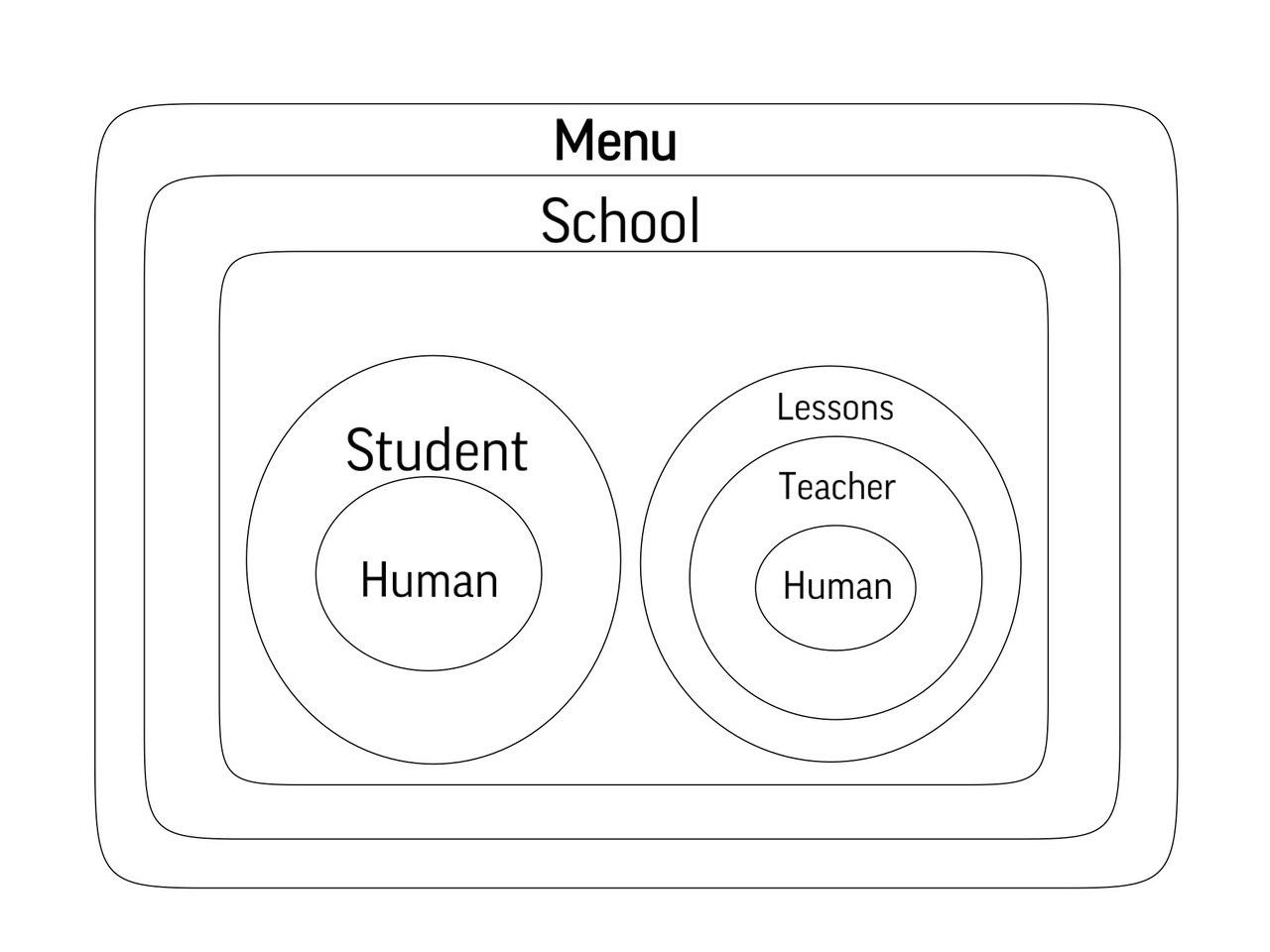
В програмі наявні змінні:

а) типу int, string, vector для виконання обчислень та виведення в консоль;

б) vector об’єктів класу для виконання методів.

**2.3 Порядок викликів функцій:**

* Виклик головного методу «Головне меню» (run();)
* Виклик методу «Меню уроків» (lessonMenu();)
* Виклик методу «Меню вчителя» (teacherMenu();)
* Виклик методу «Меню студента» (studentMenu();)
* Виклик методу «Меню їдальні» (canteenMenu();)

Діаграма класів наведена нижче. 

***Рис. 1.2. Спрощений вигляд діаграми класів програмного проекту.***

# 3.Опис реалізації основних функцій/методів:

3.1Функція зчитування файлів

void Canteen::readFiles() {

foodName.open("foodName.txt");

foodPrice.open("foodPrice.txt");

string str, strTemp;

int prc, prcTemp = 0;

if (foodName.is\_open()) {

while (true) {

foodName >> str;

if (str == strTemp)

break;

else {

food.push\_back(str);

strTemp = str;

}

foodPrice >> prc;

if (prc == prcTemp)

break;

else {

price.push\_back(prc);

prcTemp = prc;

}

}

}

else

cout << "File not found!" << endl;

foodName.close();

foodPrice.close();

}

3.2Функція додавання нової їжі

void Canteen::addFood() {

foodName.open("foodName.txt", ios::app);

foodPrice.open("foodPrice.txt", ios::app);

int choice = 1;

string nameTemp;

int priceTemp;

while (choice == 1) {

cout << "What food would you like to add?" << endl;

cin >> nameTemp;

cout << "Enter " << nameTemp << " price:" << endl;

cin >> priceTemp;

foodName << nameTemp << endl;

foodPrice << " " << priceTemp << " ";

cout << "Add more ?" << endl << "1 - yes" << endl << "0 - no" << endl;

cin >> choice;

}

foodName.close();

foodPrice.close();

}

3.3Функція продажу їжі

void Canteen::sellFood() {

int choice2;

cout << "Who is buying food ?" << endl <<

"1 - teacher" << endl <<

"2 - student" << endl;

cin >> choice2;

if (choice2 == 1) {

if (teacherCount != 0) {

int choice = pickTeacher(),

choice1;

cout << "What food would you like to buy ?" << endl;

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

cout << i + 1 << ": " << food[i] << " - " << price[i] << endl;

}

cin >> choice1;

choice1--;

teachers[choice].setMoney(teachers[choice].getMoney() - price[choice1]);

cout << food[choice1] << " successfully bought !" << endl <<

"Money left: " << teachers[choice].getMoney() << "$" << endl;

}

else

cout << "There is no teachers !" << endl;

}

else if (choice2 == 2) {

if (studentCount != 0) {

int choice = pickStudent(),

choice1;

cout << "What food would you like to buy ?" << endl;

showFood();

cin >> choice1;

choice1--;

students[choice].setMoney(getMoney() - price[choice1]);

cout << food[choice1] << " successfully bought !" << endl <<

"Money left: " << students[choice].getMoney() << endl;

}

else

cout << "There is no students !" << endl;

}

}

3.4 Функція перевірки можливості проведення уроку

int Teacher::searchTeacher(string lesson) {

int count = 0,

choice;

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

if (teachers[i].getObject() == lesson) {

count++;

cout << i + 1 << ": " << teachers[i].getName() << " AVAILABLE" << endl;

}

else {

cout << i + 1 << ": " << teachers[i].getName() << " NOT AVAILABLE" << endl;

}

}

if (count == 0) {

cout << "No available teacher for that lesson !" << endl;

return -1;

}

else {

cout << "Which teacher would you like to choose ?" << endl;

cin >> choice;

choice--;

return choice;

}

}

3.5 Функція додавання вчителя

void Teacher::addTeacher() {

Teacher \*t;

t = new Teacher;

t->setAllTeacher();

teachers.push\_back(\*t);

teacherCount++;

}

3.6 Функція проведення уроку

void Lessons::takeLesson() {

int choice,

teacher;

cout << "Which lesson do you want to take ?" << endl;

for (int i = 0; i < 7; i++) {

cout << i + 1 << ": " << lessons[i] << endl;

}

cin >> choice;

choice--;

teacher = searchTeacher(lessons[choice]);

if (teacher != -1) {

teachers[teacher].setMoney(getMoney() + 100);

cout << "Lesson went good!" << endl <<

"Teacher " << teachers[teacher].getName() << " earned money." << endl <<

"Money: " << teachers[teacher].getMoney() << endl;

}

}

3.7 Функція меню уроків

void lessonMenu(School & obj) {

system("CLS");

int menu\_item = 0, run, x = 1;

bool running = true;

gotoXY(0, 1); cout << "->";

while (running)

{

gotoXY(2, 0); cout << "Lessons Menu";

gotoXY(2, 1); cout << "1) Show lessons";

gotoXY(2, 2); cout << "2) Take lesson";

gotoXY(2, 3); cout << "Close submenu";

system("pause>nul"); // the >nul bit causes it the print no message

if (GetAsyncKeyState(VK\_DOWN) && x != 11 && x >= 0) //down button pressed

{

gotoXY(0, x); cout << " ";

x++;

if (x > 3) {

x--;

gotoXY(0, x); cout << "->";

}

else {

gotoXY(0, x); cout << "->";

menu\_item++;

}

continue;

}

if (GetAsyncKeyState(VK\_UP) && x != 7) //up button pressed

{

gotoXY(0, x); cout << " ";

x--;

if (x < 1) {

x++;

gotoXY(0, x); cout << "->";

}

else {

gotoXY(0, x); cout << "->";

menu\_item--;

}

continue;

}

if (GetAsyncKeyState(VK\_RETURN)) { // Enter key pressed

switch (menu\_item) {

case 0: {

gotoXY(0, 7);

obj.showLessons();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 1: {

gotoXY(0, 7);

obj.takeLesson();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 2: {

running = false;

system("CLS");

}

}

}

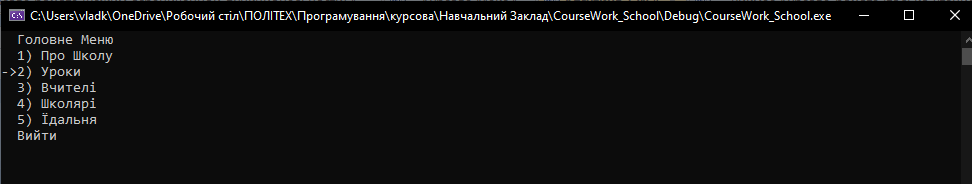
}

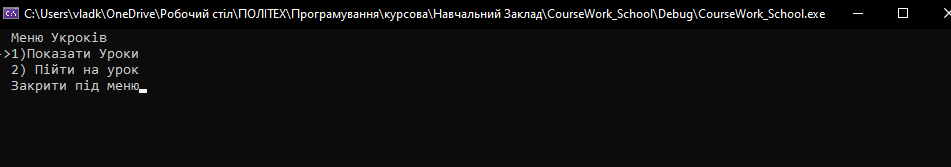
**3.Відлагодження та тестування реалізованої програми згідно індивідуального завдання.**

Під час тестування окремих частин програм з’являлись певні недоліки та помилки. Щоб все правильно працювало потрібно було не дати можливості користувачу змушувати програму некоректно працювати. Тож всі можливі неполадки були ізольовані і опрацьовані так, щоб з’являлось певне повідомлення з певною інформацією, звідки користувач дізнавався як слід користуватися програмою, щоб уникати таких невизначеностей.

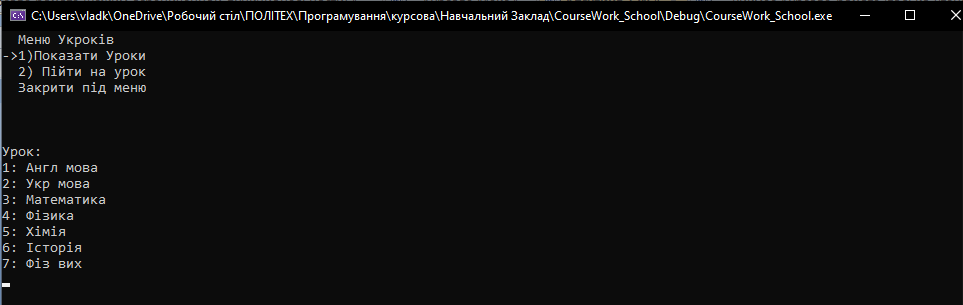
**3.1Відлагодження та тестування програми в цілому:**

Подібні кроки були виконані і з перевіркою всієї програми. Нижче наведені скріншоти, які запевнюють в коректній роботі програми:

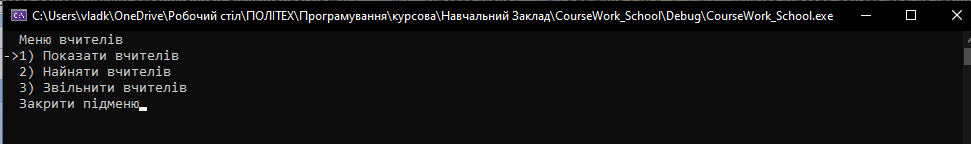
1) 

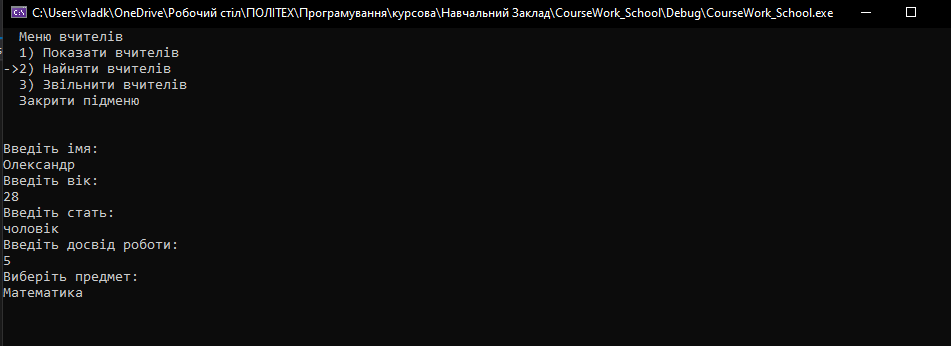
2) 

3)

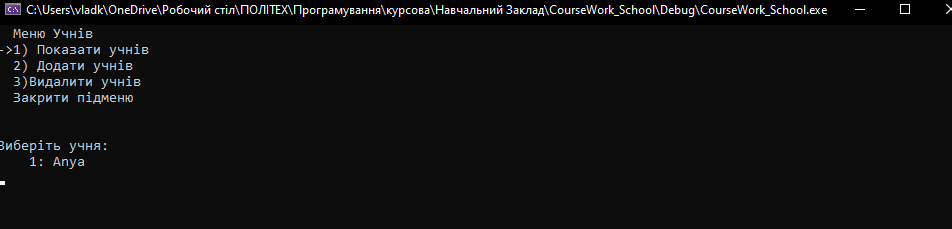


4)

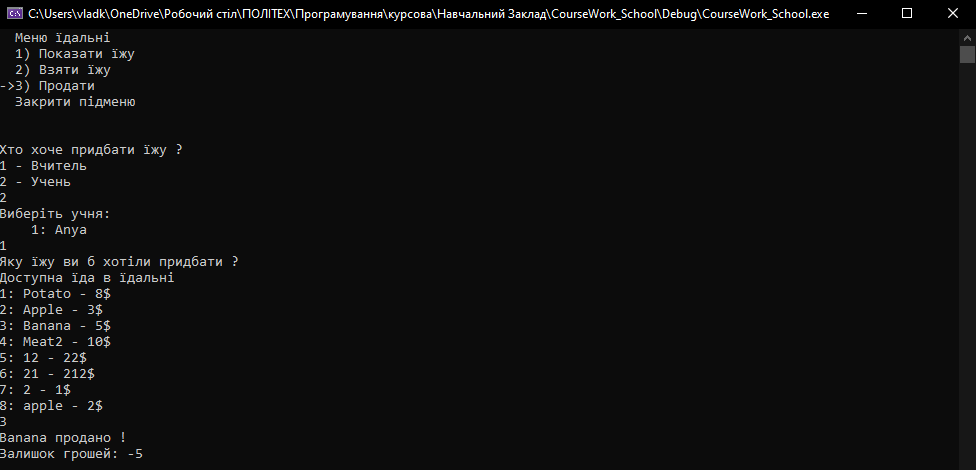


5) 

6)



7)



# 4.Аналіз продуктивності (часу роботи) та структурної складності реалізованої програмзгідно індивідуального завдання.

.

**4.1Аналіз часу виконання програми в залежності від об'єму вхідних даних.**

Всі операції всередині програми виконуються миттєво, немає затримки між переходами по меню та виведенням інформації на консоль. Читання даних з файлу та виконання операцій з числами виконується бездоганно.

# 

# Висновки

**В ході виконання курсової роботи я написав програмне забезпечення для адміністрування і керування роботою школи, дослідив загальноприйняті парадигми побудови робочого процессу відповідних структур закладів громадського харчування, засвоїв головні доктрини і особливості об’єктно-орієнтованого програмування : інкапсуляцію, наслідування, поліформізм. Вдосконалив свої навички написання та відлагодження складних програм, отримав досвід в ході роботі над крупним проектом. Розроблене консольне меню.**

# 

# Список літератури

1. *Прата С. Язык программирования C++. Лекции и упражнения. 6-е издание / Стивен Прата; пер. с англ. – М.: ООО «И.Д. Вильямс», 2012. – 1248 с.: ил. – Парал. тит. англ.*
2. *Полный справочник по C++, 4-е издание / ГербертШилдт — М.: «Вильямс», 2011. - 800 с.*
3. *Chris Giametta «Pro Flex on Spring», 2009.- p.445.*
4. *Роберт Дж. Оберг «ТехнологияCOM + Основы и программирование = Understanding and ProgrammingCOM+: A Practical Guideto Windows 2000 First Edition». — М.:* [*«Вильямс»,*](http://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%BB%D1%8C%D1%8F%D0%BC%D1%81_%28%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE%29) *2000.- С. 480.*
5. *Тамре Л. «Введение в тестирование програмного обеспечения ».- М.: Издательский дом*

*«Вильямс», 2003.- С. 368.*

1. *Липаев В.В. «Обеспечение качества программных средств. Методы и стандарты». - М.:*

*Синтег, 2001.- С. 246.*

1. *Макгрегор Дж., Сайкс Д. «Тестирование объектно-ориентированного програмного обеспечения».- К: Диасофт, 2002.- С. 432.*
2. *Эммерих В. Конструирование распределенных объектов. Методы и средства*

*программирования интероперабельных объектов в архитектурах OMG/CORBA, Microsoft/COM и Java/RMI. Пер. с англ. — М.: Мир, 2002. — С. 510.*

*9. Макгрегор Дж., Сайкс Д. Тестирование объектно-ориентированного програмного обеспечения.- К: Диасофт, 2002.- С. 432.*

1. *Тамре Л. Введение в тестирование програмного обеспечения.- М.: Издательский дом―Вильямс‖, 2003.- С. 368.*
2. *Э. Гамм, Р. Хелм, Р. Джонсон, Д. Влиссидес «Приѐмы объектно-ориентированного проектирования. Паттерны проектирования», Питер, Москва, 2007.- С. 366.*
3. *Джон Влиссидес «Применение шаблонов проектирования», Москва, 2003.- С. 130.*

# Додатки

**А. Лістинг модулів програми згідно індивідуального завдання**

// pch.cpp: source file corresponding to pre-compiled header; necessary for compilation to succeed

#include "pch.h"

// In general, ignore this file, but keep it around if you are using pre-compiled headers.

/////////////////////////////////////////////////////////////////////////////////////////

#include "Canteen.h"

#include "Menu.h"

int main() {

Menu obj;

obj.run();

return 0;

}

/////////////////////////////////////////////////////////////////////////////////////////

#include "pch.h"

#include "Canteen.h"

#include <string>

#include <iostream>

#include<clocale>

using namespace std;

Canteen::Canteen(){}

void Canteen::readFiles() {

setlocale(LC\_CTYPE, "ukr");

foodName.open("foodName.txt");

foodPrice.open("foodPrice.txt");

string str, strTemp;

int prc, prcTemp = 0;

if (foodName.is\_open()) {

while (true) {

foodName >> str;

if (str == strTemp)

break;

else {

food.push\_back(str);

strTemp = str;

}

foodPrice >> prc;

if (prc == prcTemp)

break;

else {

price.push\_back(prc);

prcTemp = prc;

}

}

}

else

cout << "File not found!" << endl;

foodName.close();

foodPrice.close();

}

void Canteen::addFood() {

setlocale(LC\_CTYPE, "ukr");

foodName.open("foodName.txt", ios::app);

foodPrice.open("foodPrice.txt", ios::app);

int choice = 1;

string nameTemp;

int priceTemp;

while (choice == 1) {

cout << "Яку б їжу ви б хотiли взяти?" << endl;

cin >> nameTemp;

cout << "Введіть " << nameTemp << " цiна:" << endl;

cin >> priceTemp;

foodName << nameTemp << endl;

foodPrice << " " << priceTemp << " ";

cout << "Добавки ?" << endl << "1 - так" << endl << "0 - нi" << endl;

cin >> choice;

}

foodName.close();

foodPrice.close();

}

void Canteen::sellFood() {

setlocale(LC\_CTYPE, "ukr");

int choice2;

cout << "Хто хоче придбати їжу ?" << endl <<

"1 - Вчитель" << endl <<

"2 - Учень" << endl;

cin >> choice2;

if (choice2 == 1) {

if (teacherCount != 0) {

int choice = pickTeacher(),

choice1;

cout << "Яку їжу ви б хотiли придбати ?" << endl;

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

cout << i + 1 << ": " << food[i] << " - " << price[i] << endl;

}

cin >> choice1;

choice1--;

teachers[choice].setMoney(teachers[choice].getMoney() - price[choice1]);

cout << food[choice1] << " successfully bought !" << endl <<

"Залишок грошей: " << teachers[choice].getMoney() << "$" << endl;

}

else

cout << "Тут нема вчителiв !" << endl;

}

else if (choice2 == 2) {

if (studentCount != 0) {

int choice = pickStudent(),

choice1;

cout << "Яку їжу ви б хотiли придбати ?" << endl;

showFood();

cin >> choice1;

choice1--;

students[choice].setMoney(getMoney() - price[choice1]);

cout << food[choice1] << " продано !" << endl <<

"Залишок грошей: " << students[choice].getMoney() << endl;

}

else

cout << "Тут немає учнiв !" << endl;

}

}

void Canteen::showFood() {

setlocale(LC\_CTYPE, "ukr");

cout << "Доступна їда в їдальнi" << endl;

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

cout << i + 1 << ": " << food[i] << " - " << price[i] << "$" << endl;

}

}

Canteen::~Canteen(){}

/////////////////////////////////////////////////////////////////////////////////////////

#include "pch.h"

#include "Human.h"

#include<clocale>

Human::Human(){

setlocale(LC\_CTYPE, "ukr");

this->age = 0;

this->name = "Невiдомо";

this->sex = "Невiдомо";

this->money = 0;

}

Human::Human(int age, string name, string sex, int money) {

this->age = age;

this->name = name;

this->sex = sex;

this->money = money;

}

void Human::setAllHuman() {

setlocale(LC\_CTYPE, "ukr");

cout << "Введiть iмя: " << endl;

cin >> name;

cout << "Введiть вiк: " << endl;

cin >> age;

cout << "Введiть стать: " << endl;

cin >> sex;

}

void Human::setAge(int age) {

this->age = age;

}

int Human::getAge() {

return age;

}

void Human::setName(string name) {

this->name = name;

}

string Human::getName(){

return name;

}

void Human::setSex(string sex) {

this->sex = sex;

}

string Human::getSex() {

return sex;

}

void Human::setMoney(int money) {

this->money = money;

}

int Human::getMoney() {

return money;

}

Human::~Human(){

}

**////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

#include "Lessons.h"

#include<clocale>

Lessons::Lessons() {

}

void Lessons::takeLesson() {

setlocale(LC\_CTYPE, "ukr");

int choice,

teacher;

cout << "Якi б уроки ви б хотiли вiдвiдати?" << endl;

for (int i = 0; i < 7; i++) {

cout << i + 1 << ": " << lessons[i] << endl;

}

cin >> choice;

choice--;

teacher = searchTeacher(lessons[choice]);

if (teacher != -1) {

teachers[teacher].setMoney(getMoney() + 100);

cout << "Урок пройшов добре!" << endl <<

"Учитель " << teachers[teacher].getName() << " Заробили грошей." << endl <<

"Грошей: " << teachers[teacher].getMoney() << endl;

}

}

void Lessons::showLessons() {

setlocale(LC\_CTYPE, "ukr");

cout << "Урок:" << endl;

for (int i = 0; i < 7; i++) {

cout << i + 1 << ": " << lessons[i] << endl;

}

}

Lessons::~Lessons() {}

**////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

#include "School.h"

#include<clocale>

School::School() {

readFiles();

};

void School::showSchool() {

setlocale(LC\_CTYPE, "ukr");

cout << "Назва шеколи: " << name << endl <<

"Число вчителiв: " << teacherCount << endl <<

"Число учнiв: " << studentCount << endl;

}

School::~School() {};

/////////////////////////////////////////////////////////////////////////////////////////

bool Student::showStudents() {

setlocale(LC\_CTYPE, "ukr");

if (studentCount != 0) {

int count = 1;

cout << "Виберiть учня: " << endl;

for (it = students.begin(); it != students.end(); it++) {

cout << setw(5) << count << ": ";

string name = it->getName();

cout << name << endl;

count++;

}

return true;

}

else {

cout << "Вiдсутнi учнi,введiть ще!" << endl;

return false;

}

}

int Student::pickStudent() {

setlocale(LC\_CTYPE, "ukr");

int choice;

//cout << "Pick student:" << endl;

if (showStudents() == true) {

cin >> choice;

choice--;

return choice;

}

else {

cout << "Вiдсутнi учнi,введiть ще!" << endl;

}

}

Student::~Student(){

}

**/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

#include "pch.h"

#include "Teacher.h"

#include <iomanip>

#include<clocale>

Teacher::Teacher(){

setlocale(LC\_CTYPE, "ukr");

this->experince = 0;

this->object = "Невiдомо";

}

Teacher::Teacher(int experience, string object) {

this->experince = experience;

this->object = object;

}

void Teacher::setExperience(int experience) {

this->experince = experience;

}

int Teacher::getExperience() {

return experince;

}

void Teacher::setObject(string object) {

this->object = object;

}

string Teacher::getObject() {

return object;

}

void Teacher::setAllTeacher() {

setAllHuman();

setlocale(LC\_CTYPE, "ukr");

cout << "Введiть досвiд роботи: " << endl;

cin >> experince;

cout << "Виберiть предмет: " << endl;

cin >> object;

}

void Teacher::addTeacher() {

Teacher \*t;

t = new Teacher;

t->setAllTeacher();

teachers.push\_back(\*t);

teacherCount++;

}

bool Teacher::showTeacher() {

setlocale(LC\_CTYPE, "ukr");

if (teacherCount != 0) {

int count = 1;

cout << "Виберiть вчителя: " << endl;

for (it = teachers.begin(); it != teachers.end(); it++) {

cout << setw(5) << count << ": ";

string name = it->getName();

cout << name << endl;

count++;

}

return true;

}

else {

cout << "Нема вчителiв,добавте!" << endl;

return false;

}

}

int Teacher::pickTeacher() {

setlocale(LC\_CTYPE, "ukr");

int choice;

//cout << "Pick Teacher:" << endl;

if (showTeacher() == true) {

cin >> choice;

choice--;

return choice;

}

else {

cout << "Нема вчителiв,добавте!" << endl;

}

}

int Teacher::searchTeacher(string lesson) {

setlocale(LC\_CTYPE, "ukr");

int count = 0,

choice;

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

if (teachers[i].getObject() == lesson) {

count++;

cout << i + 1 << ": " << teachers[i].getName() << " Доступний" << endl;

}

else {

cout << i + 1 << ": " << teachers[i].getName() << " Недоступний" << endl;

}

}

if (count == 0) {

cout << "Нема вiльних вчителiв для цього уроку !" << endl;

return -1;

}

else {

cout << "Якого вчителя ви хочете обрати ?" << endl;

cin >> choice;

choice--;

return choice;

}

}

int Teacher::getTeacherCount() {

return teacherCount;

}

Teacher::~Teacher() {}

**////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

**Бібліотеки:**

#pragma once

#include<clocale>

#include <conio.h>

#include <windows.h>

#include <cstdlib>

#include "School.h"

#include <iostream>

using namespace std;

class Menu : virtual public School {

public:

HANDLE console = GetStdHandle(STD\_OUTPUT\_HANDLE); // used for goto

COORD CursorPosition; // used for goto

void gotoXY(int x, int y){

CursorPosition.X = x;

CursorPosition.Y = y;

SetConsoleCursorPosition(console, CursorPosition);

}

int cleanConsole(int x) {

getchar();

system("CLS");

gotoXY(0, 1);

cout << "->";

gotoXY(0, 1);

return 1;

}

void teacherMenu(School & obj) {

system("CLS");

int menu\_item = 0, run, x = 1;

bool running = true;

gotoXY(0, 1); cout << "->";

while (running){

setlocale(LC\_CTYPE, "ukr");

gotoXY(2, 0); cout << "Меню вчителiв";

gotoXY(2, 1); cout << "1) Показати вчителiв";

gotoXY(2, 2); cout << "2) Найняти вчителiв";

gotoXY(2, 3); cout << "3) Звiльнити вчителiв";

gotoXY(2, 4); cout << "Закрити пiдменю";

system("pause>nul"); // the >nul bit causes it the print no message

if (GetAsyncKeyState(VK\_DOWN) && x != 11 && x >= 0) //down button pressed

{

gotoXY(0, x); cout << " ";

x++;

if (x > 4) {

x--;

gotoXY(0, x); cout << "->";

}

else {

gotoXY(0, x); cout << "->";

menu\_item++;

}

continue;

}

if (GetAsyncKeyState(VK\_UP) && x != 7) //up button pressed

{

gotoXY(0, x); cout << " ";

x--;

if (x < 1) {

x++;

gotoXY(0, x); cout << "->";

}

else {

gotoXY(0, x); cout << "->";

menu\_item--;

}

continue;

}

if (GetAsyncKeyState(VK\_RETURN)) { // Enter key pressed

switch (menu\_item) {

case 0: {

gotoXY(0, 7);

obj.showTeacher();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 1: {

gotoXY(0, 7);

obj.addTeacher();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 2: {

gotoXY(0, 7);

//obj.deleteTeacher();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 3: {

running = false;

system("CLS");

}

}

}

}

}

void studentMenu(School & obj) {

system("CLS");

int menu\_item = 0, run, x = 1;

bool running = true;

gotoXY(0, 1); cout << "->";

while (running) {

setlocale(LC\_CTYPE, "ukr");

gotoXY(2, 0); cout << "Меню Учнiв";

gotoXY(2, 1); cout << "1) Показати учнiв";

gotoXY(2, 2); cout << "2) Додати учнiв";

gotoXY(2, 3); cout << "3)Видалити учнiв";

gotoXY(2, 4); cout << "Закрити пiдменю";

system("pause>nul"); // the >nul bit causes it the print no message

if (GetAsyncKeyState(VK\_DOWN) && x != 11 && x >= 0) //down button pressed

{

gotoXY(0, x); cout << " ";

x++;

if (x > 4) {

x--;

gotoXY(0, x); cout << "->";

}

else {

gotoXY(0, x); cout << "->";

menu\_item++;

}

continue;

}

if (GetAsyncKeyState(VK\_UP) && x != 7) //up button pressed

{

gotoXY(0, x); cout << " ";

x--;

if (x < 1) {

x++;

gotoXY(0, x); cout << "->";

}

else {

gotoXY(0, x); cout << "->";

menu\_item--;

}

continue;

}

if (GetAsyncKeyState(VK\_RETURN)) { // Enter key pressed

switch (menu\_item) {

case 0: {

gotoXY(0, 7);

obj.showStudents();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 1: {

gotoXY(0, 7);

obj.addStudent();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 2: {

gotoXY(0, 7);

//obj.deleteStudent();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 3: {

running = false;

system("CLS");

}

}

}

}

}

void canteenMenu(School & obj) {

system("CLS");

int menu\_item = 0, run, x = 1;

bool running = true;

gotoXY(0, 1); cout << "->";

while (running)

{

setlocale(LC\_CTYPE, "ukr");

gotoXY(2, 0); cout << "Меню їдальнi";

gotoXY(2, 1); cout << "1) Показати їжу";

gotoXY(2, 2); cout << "2) Взяти їжу";

gotoXY(2, 3); cout << "3) Продати";

gotoXY(2, 4); cout << "Закрити пiдменю";

system("pause>nul"); // the >nul bit causes it the print no message

if (GetAsyncKeyState(VK\_DOWN) && x != 11 && x >= 0) //down button pressed

{

gotoXY(0, x); cout << " ";

x++;

if (x > 4) {

x--;

gotoXY(0, x); cout << "->";

}

else {

gotoXY(0, x); cout << "->";

menu\_item++;

}

continue;

}

if (GetAsyncKeyState(VK\_UP) && x != 7) //up button pressed

{

gotoXY(0, x); cout << " ";

x--;

if (x < 1) {

x++;

gotoXY(0, x); cout << "->";

}

else {

gotoXY(0, x); cout << "->";

menu\_item--;

}

continue;

}

if (GetAsyncKeyState(VK\_RETURN)) { // Enter key pressed

switch (menu\_item) {

case 0: {

gotoXY(0, 7);

obj.showFood();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 1: {

gotoXY(0, 7);

obj.addFood();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 2: {

gotoXY(0, 7);

obj.sellFood();

x = 1;

menu\_item = 0;

break;

}

case 3: {

running = false;

system("CLS");

}

}

}

}

}

void lessonMenu(School & obj) {

system("CLS");

int menu\_item = 0, run, x = 1;

bool running = true;

gotoXY(0, 1); cout << "->";

while (running)

{

setlocale(LC\_CTYPE, "ukr");

gotoXY(2, 0); cout << "Меню Укрокiв";

gotoXY(2, 1); cout << "1)Показати Уроки";

gotoXY(2, 2); cout << "2) Пiйти на урок";

gotoXY(2, 3); cout << "Закрити пiд меню";

system("pause>nul"); // the >nul bit causes it the print no message

if (GetAsyncKeyState(VK\_DOWN) && x != 11 && x >= 0) //down button pressed

{

gotoXY(0, x); cout << " ";

x++;

if (x > 3) {

x--;

gotoXY(0, x); cout << "->";

}

else {

gotoXY(0, x); cout << "->";

menu\_item++;

}

continue;

}

if (GetAsyncKeyState(VK\_UP) && x != 7) //up button pressed

{

gotoXY(0, x); cout << " ";

x--;

if (x < 1) {

x++;

gotoXY(0, x); cout << "->";

}

else {

gotoXY(0, x); cout << "->";

menu\_item--;

}

continue;

}

if (GetAsyncKeyState(VK\_RETURN)) { // Enter key pressed

switch (menu\_item) {

case 0: {

gotoXY(0, 7);

obj.showLessons();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 1: {

gotoXY(0, 7);

obj.takeLesson();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 2: {

running = false;

system("CLS");

}

}

}

}

}

int run()

{

School obj;

int menu\_item = 0, run, x = 1;

bool running = true;

gotoXY(0, 1); cout << "->";

while (running)

{

setlocale(LC\_CTYPE, "ukr");

//gotoXY(0, 1); cout << "->";

gotoXY(2, 0); cout << "Головне Меню";

gotoXY(2, 1); cout << "1) Про Школу";

gotoXY(2, 2); cout << "2) Уроки";

gotoXY(2, 3); cout << "3) Вчителi";

gotoXY(2, 4); cout << "4) Школярi";

gotoXY(2, 5); cout << "5) Їдальня";

gotoXY(2, 6); cout << "Вийти";

system("pause>nul"); // the >nul bit causes it the print no message

if (GetAsyncKeyState(VK\_DOWN) && x != 11 && x >= 0) //down button pressed

{

gotoXY(0, x); cout << " ";

x++;

if (x > 6) {

x--;

gotoXY(0, x); cout << "->";

}

else {

gotoXY(0, x); cout << "->";

menu\_item++;

}

continue;

}

if (GetAsyncKeyState(VK\_UP) && x != 7) //up button pressed

{

gotoXY(0, x); cout << " ";

x--;

if (x < 1) {

x++;

gotoXY(0, x); cout << "->";

}

else {

gotoXY(0, x); cout << "->";

menu\_item--;

}

continue;

}

if (GetAsyncKeyState(VK\_RETURN)) { // Enter key pressed

switch (menu\_item) {

case 0: {

gotoXY(0, 7);

obj.showSchool();

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 1: {

gotoXY(0, 7);

lessonMenu(obj);

x = 1;

menu\_item = 0;

gotoXY(0, 1); cout << "->";

break;

}

case 2: {

gotoXY(0, 7);

teacherMenu(obj);

x = 1;

menu\_item = 0;

gotoXY(0, 1); cout << "->";

break;

}

case 3: {

gotoXY(0, 7);

studentMenu(obj);

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 4: {

gotoXY(0, 7);

canteenMenu(obj);

cleanConsole(x);

x = 1;

menu\_item = 0;

break;

}

case 5: {

running = false;

}

}

}

}

return 0;

}

};

/////////////////////////////////////////////////////////////////////////////////////////

#pragma once

#include <vector>

#include <fstream>

#include "Student.h"

#include "Teacher.h"

#include<clocale>

using namespace std;

class Canteen : virtual public Student, virtual public Teacher {

protected:

vector <string> food;

vector <int> price;

int profit;

fstream foodName;

fstream foodPrice;

public:

Canteen();

void readFiles();

void addFood();

void sellFood();

void showFood();

~Canteen();

};

**////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////**

#pragma once

#include "Teacher.h"

#include<clocale>

class Lessons : virtual public Teacher {

string lessons[7] = { "Англ мова", "Укр мова", "Математика", "Фiзика", "Хiмiя", "Iсторiя", "Фiз вих" };

public:

Lessons();

void takeLesson();

void showLessons();

~Lessons();

};

////////////////////////////////////////////////////////////////////////////////////////

#ifndef PCH\_H

#define PCH\_H

// TODO: add headers that you want to pre-compile here

#endif //PCH\_H

/////////////////////////////////////////////////////////////////////////////////////////

#pragma once

#include "Canteen.h"

#include "Lessons.h"

#include<clocale>

class School : virtual public Canteen, virtual public Lessons {

protected:

string name;

public:

School();

void showSchool();

~School();

};

/////////////////////////////////////////////////////////////////////////////////////////

#pragma once

#include<clocale>

#include "Human.h"

#include <vector>

class Student : virtual public Human{

protected:

int grade,

avgMark;

vector <Student> students;

vector <Student>::iterator it;

int studentCount;

public:

Student();

Student(int grade, int avgMark);

void setGrade(int grade);

int getGrade();

void setAvgMark(int avgMark);

int getAvgMark();

void setAllStudent();

void addStudent();

int pickStudent();

bool showStudents();

~Student();

};

/////////////////////////////////////////////////////////////////////////////////////////

#include <vector>

#include<clocale>

class Teacher : virtual public Human{

protected:

int experince,

teacherCount;

string object;

vector <Teacher> teachers;

vector <Teacher>::iterator it;

public:

Teacher();

Teacher(int experience, string object);

void setExperience(int experience);

int getExperience();

void setObject(string object);

string getObject();

void addTeacher();

int pickTeacher();

bool showTeacher();

void setAllTeacher();

int searchTeacher(string lesson);

int getTeacherCount();

~Teacher();

};

/////////////////////////////////////////////////////////////////////////////////////////