**Міністерство освіти і науки України**

**Національний університет «Львівська політехніка»**

**Кафедра ЕОМ**



**Курсова робота**

з дисципліни «Програмування, частина 2 (Об'єктно-орієнтоване програмування)»

на тему:

**«Базові принципи об’єктно-орієнтованого програмування»**

Індивідуальне завдання: « Пошта »

**Виконав: Марков Л.Ю.**

**Прийняв: Козак Н.Б.**

**Львів 2020**

***Анотація***

У цій курсовій роботі було виконано написання програмного забезпечення на мові програмування С++ з використанням основних принципів об’єктно-орієнтованого програмування.

Програма являє собою спрощену модель функціонування поштового відділення та включає у себе роботу з текстовими файлами, а саме: запис, зчитування, видалення та додавання нових даних. Консольний інтерфейс програмного забезпечення написаний українською мовою.

***Метою виконання*** цієї курсової роботи є засвоєння основних методів та алгоритмів об’єктно-орієнтованого програмування мови С++, та практичне застосування їх у цій сфері.

***Зміст***

**Завдання на курсову роботу …………………………………………………….. 3**

**Вступ ……………………………………………………………………………….. 4**

**1. Визначення поняття ООП (Об'єктно-орієнтоване програмування ), його основні принципи та обґрунтування вибору його технологій …………….... 5**

1.1 ООП як основа сучасного програмування …………………………..……...... 5

1.2 Визначення ООП ……………………………………………………..…….. 6

1.3 Фундаментальні поняття ООП ………………………………………...….. 7

1.4 Переваги вибраного методу ……………………………………………..… 8

**2. Аналіз та розробка алгоритму згідно індивідуального завдання .…….… 9**

2.1 Загальний опис поставленої задачі ……………………………………....... 9

2.2 Визначення основних задач покладених в основу програми …..……..... 10

2.3 Алгоритм виконання роботи програми …………………………...............12

**3. Реалізація алгоритму у програмному коді ……………………….………...14**

**4. Тестування програми ……………………………………………………........15**

**Висновок ………………………………………………………………………….. 19**

**Література ………………………………………………………………………... 20**

**Додаток. Код програми ……..…….…………………………………………….. 21**

***Завдання на курсову роботу***

У даній курсовій роботі повинно бути реалізовано на мові С++ таку предметну галузь як “Пошта”. У ній повинні бути використані основні принципи об’єктно-орієнтованого програмування. Також програма має підтримувати роботу та операції з файлами, серед яких запис, зчитування, видалення та додавання нових даних. Для користувача програми потрібно розробити систему підказок, яка допомогла б йому зорієнтуватися у використанні даної програми. Схема програми (Рис.1).

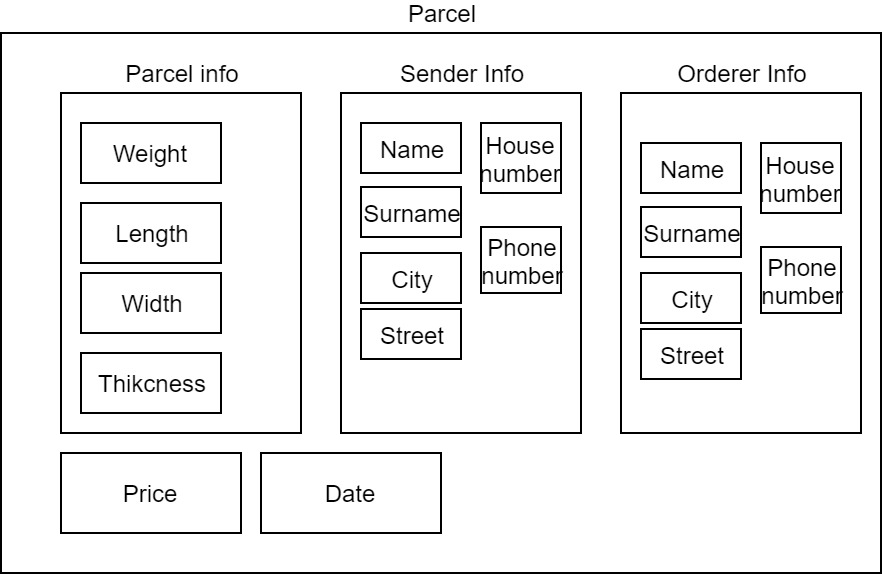


Рис. 1 Схема предметної галузі мого завдання

***Вступ***

**Об'єктно-орієнтоване програмування** (ООП) — одна з парадигм програмування, яка розглядає програму як множину «об'єктів», що взаємодіють між собою. Основу ООП складають три основні концепції: інкапсуляція, успадкування та поліморфізм. Одною з переваг ООП є краща модульність програмного забезпечення (тисячу функцій процедурної мови, в ООП можна замінити кількома десятками класів із своїми методами). Попри те, що ця парадигма з'явилась в 1960-тих роках, вона не мала широкого застосування до 1990-тих, коли розвиток комп'ютерів та комп'ютерних мереж дозволив писати надзвичайно об'ємне і складне програмне забезпечення, що змусило переглянути підходи до написання програм. Однією з таких мов програмування стала мова С++. У 1990-х р. дана мова стала однією з найуживаніших мов програмування загального призначення. ЇЇ розробником є професор Техаського університету А&М Б'ярн Страуструп. На початку свого існування ця мова дістала назву «Сі з класами». Згодом Страуструп перейменував мову на C++ у 1983 р. У 1998 році ратифіковано міжнародний стандарт мови С++: ISO/IEC 14882 «Standard for the C++ Programming Language». Поточна версія цього стандарту — ISO/IEC 14882:2011. На сьогоднішній день цю мову використовують для системного програмування, розробки програмного забезпечення, написання драйверів, потужних серверних та клієнтських програм, а також для розробки розважальних програм таких як відеоігри. Також мова С++ суттєво вплинула на інші, популярні сьогодні, мови програмування такі як С# та Java. [3, 6, 8]

**1. Визначення поняття об’єктно-орієнтованого програмування(ООП), його основні принципи та обґрунтування його технологій.**

**1.1 ООП як основа сучасного програмування**

Із розвитком комп’ютерних технологій, коли програми поступово почали ускладнюватися, виникла потреба для нових засобів програмування, які б могли впорядкувати та спростити сприйняття та написання програм, які могли б містити тисячі стрічок коду. Для подолання цієї проблеми було розроблено структурне програмування. Структурне програмування, на відміну від його попередника — стихійного програмування, має певні правила та засоби, які суттєво вдосконалили процес написання програм. Але за допомогою структурного програмування дуже важко писати програми ще більшої складності, які вимагає сучасність. Тому була створена парадигма Об’єктно-орієнтованого програмування. Першою мовою програмування, в якій були запропоновані основні поняття, які згодом склалися в парадигму, була Симула, але термін «об'єктна орієнтованість" не використовувався в контексті використання цієї мови. У момент його появи в 1967 році в ньому були запропоновані революційні ідеї: об'єкти, класи, віртуальні методи і ін., проте це все не було сприйнято сучасниками як щось грандіозне. Фактично, Симула була «Алголом з класами», що спрощує вираз в процедурному програмуванні багатьох складних концепцій. Поняття класу в Симулі може бути повністю визначено через композицію конструкцій Алгола (тобто клас в Симулі - це щось складне, що описується за допомогою примітивів).

Погляд на програмування «під новим кутом» (відмінним від процедурного) запропонували Алан Кей і Ден Інгаллс в мові Smalltalk. Тут поняття класу стало основотвірною ідеєю для всіх інших конструкцій мови (тобто клас в Smalltalk є примітивом, за допомогою якого описані більш складні конструкції). Саме ця мова стала першою широко поширеною об'єктно-орієнтованою мовою програмування.

В даний час кількість прикладних мов програмування (список мов), що реалізують об'єктно-орієнтовану парадигму, є найбільшою по відношенню до інших парадигм. Найбільш поширені в промисловості такі мови як С ++, Delphi, C #, Java які втілюють об'єктну модель Симула. Прикладами мов, що спираються на модель Smalltalk, є Objective-C, Python, Ruby. [5, 7, 8 - 10]

**1.2 Визначення ООП**

Об'єктно-орієнтоване програмування (ООП) - методологія програмування, заснована на представленні програми у вигляді сукупності об'єктів, кожен з яких є екземпляром певного класу, а класи утворюють ієрархію спадкування. Лука Карделла і Мартін Абаді побудували теоретичне обґрунтування ООП і класифікацію на основі цього обґрунтування. Вони відзначають, що виділені ними поняття і категорії разом зустрічаються далеко не у всіх ОО-мовах, більшість мов підтримують лише підмножини теорії, а часом і своєрідні відхилення від неї.

Ключові поняття:

* квантифікація змінних типу (універсальна, екзистенційна, обмежена);
* підтипізація (англ. subtyping - відносини «супертип-підтип»);
* включення (англ. subsumption) - окремий випадок підтипізації;
* об'єкт;
* об'єктний тип (фактично, спеціальна форма «запису», в число полів якої входять функції першого класу);
* клас.

Клас описує абстрактну поведінку. Об'єктні типи будуються на основі класу за допомогою додавання різних приватних полів і методів. Об'єкт (тобто значення об'єктного типу, традиційно називається «екземпляром класу») породжується конструктором на основі початкових параметрів.[1, 5, 8, 9]

**1.3 Фундаментальні поняття ООП**

До фундаментальних понять (принципів), використаних у переважній більшості визначень об'єктно-орієнтованого програмування належать:

* *Абстракція даних* – виділення значущої інформації і виключення її з розгляду незначною. В ООП розглядають лише абстракцію даних (нерідко називаючи її просто «абстракцією»), маючи на увазі набір значущих характеристик об'єкта, доступних решті частини програми.
* *Інкапсуляція –* властивість системи, що дозволяє об'єднати дані і методи, що працюють з ними, в класі. Одні мови (наприклад, С ++, Java або Ruby) ототожнюють інкапсуляцію з приховуванням, але інші (Smalltalk, Eiffel, OCaml) розрізняють ці поняття.
* *Спадкування –* властивість системи, що дозволяє описати новий клас на основі вже існуючого з частково або повністю запозиченою функціональністю. Клас, від якого відбувається спадкування, називається базовим, батьківським або суперкласом. Новий клас – нащадком, спадкоємцем, дочірнім або похідним класом.
* *Поліморфізм підтипів –* властивість системи, що дозволяє використовувати об'єкти з однаковим інтерфейсом без інформації про тип і внутрішню структуру об'єкта. Інший вид поліморфізму – параметричний - в ООП називають узагальненим програмуванням.
* *Клас –* абстрактні характеристики деякої сутності, включаючи характеристики самої сутності (її атрибути або властивості) та дії, які вона здатна виконувати (її поведінки, методи або можливості). Класи вносять модульність та структурованість в об'єктно-орієнтовану програму.
* *Об’єкт –* сутність в адресному просторі обчислювальної системи, що з'являється при створенні екземпляра класу (наприклад, після запуску результатів компіляції і зв'язування вихідного коду на виконання).   
  [2, 5, 7, 8]

**1.4 Переваги вибраного методу**

Від будь-якого методу програмування ми чекаємо, що він допоможе нам у вирішенні наших проблем. Але однією з найбільш значних проблем у програмуванні є складність. Чим більша і складніша програма, тим важливіше стає розбити її на невеликі, чітко окреслені частини. Щоб побороти складність, ми повинні абстрагуватися від дрібних деталей. У цьому сенсі класи являють собою дуже зручний інструмент. Класи дозволяють проводити конструювання з корисних компонент, що володіють простими інструментами, що дає можливість абстрагуватися від деталей реалізації. Також ООП дає можливість створювати розширювані системи. Це одна з найбільш значних переваг ООП і саме вона відрізняє даний підхід від традиційних методів програмування. Розширюваність означає, що існуючу систему можна змусити працювати з новими компонентами, причому без внесення до неї будь-яких змін. Компоненти можуть бути додані на етапі виконання. Багаторазового використання програмного забезпечення на практиці досягти не вдається з-за того, що існуючі компоненти вже не відповідають новим вимогам. ООП допомагає досягти цього без порушення роботи вже наявних клієнтів, що дозволяє нам витягти максимум з багаторазового використання компонент. Завдяки чому ми скорочуємо час на розробку, яке з вигодою може бути віддано іншим проектам. Також компоненти багаторазового використання звичайно містять набагато менше помилок, ніж знову розроблені, адже вони вже не раз піддавалися перевірці. [6]

**2. Аналіз та розробка алгоритму згідно індивідуального завдання**

**2.1 Загальний опис поставленої задачі**

Задачею мого курсового проекту є пошта. Перед усім його слід розглядати як взаємодію кількох важливих його складових, зокрема інформації про: відправника, отримувача та безпосередньо посилку. Усі ці частини є невід’ємними для мінімального опису її роботи. Розглянемо кожну частину поштового відділення окремо та докладніше.

**Посилка.** Посилка повинна доставлятись між населеними пунктами, для цього ми повинні знати такі дані: габарити, вагу та відстань між пунктом відправки та пунктом призначення щоб мати змогу розрахувати її вартість та час, який буде затрачено на доставку

**Відправник/отримувач.** Кожна посилка має як відправника так і отримувача, і для того, щоб знати хто посилку відправив і хто повинен її отримати, нам потрібно мати такі дані: ПІБ, адресу та номер телефону. Хочу зазначити, що хоч властивості що для отримувача, що для відправника однакові, проте вони містять зовсім різні дані, які в жодному випадку не повинні сплутатись, інакше посилка приїде не туди, куди потрібно, а в гіршому випадку взагалі загубиться.

**Поштове відділення.** Основна задача поштового відділення правильно відправити посилку використовуючи зазначені відправником дані при цьому переконатись що всі дані зазначено вірно, а також розрахувати приблизний час доставки і вартість доставки.

**2.2** **Визначення основних задач покладених в основу програми**

Дана програма написана для вирішення певного роду задач які повністю пов’язані з такими об’єктами поштового відділення як інформація про посилку, отримувача, відправника, та безпосередньо саме поштове відділення. Основною задачею, яку повинна виконувати ця програма є операції доставки посилки, розрахунок ціни доставки та часу затраченого на доставку. В результаті я отримав набір функцій, які мені потрібно реалізувати для виконання поставленої перед мною задачі:

* Запис і зберігання даних у файлах.
* Зчитування даних з файлу.
* Додавання або видалення даних.
* Розрахунок вартості доставки.
* Розрахунок часу який буде затрачено на доставку.

Проаналізувавши мою область програмування, можна вивести основні об’єкти, відносно яких можна сформувати властивості та методи:

* Інформація про посилку.
* Поштове відділення.
* Інформація про відправника.
* Інформація про отримувача.

В класі (Інформація про посилку) описані всі дані про посилку такі як:

* Довжина.
* Ширина.
* Товщина.
* Вага.
* Відстань між населеними пунктами.

В класі (Поштове відділення) описані такі методи:

* Зміна габаритів посилки з накладанням на них певних обмежень.
* Зміна даних про відправника/отримувача.
* Розрахунок часу який буде затрачено на доставку в залежності від відстані вказаної в класі (Інформація про посилку).

Також тут реалізовано:

* Метод розрахунку ціни.
* Зберігання інформації.
* Вводу та виводу необхідних даних.

До властивостей можна віднести:

* Ціну.
* Дату відправки.

В класі (Інформація про відправника) виділю такі властивості:

* ПІБ.
* Номер телефону.
* Місто, вулиця та номер будинку.

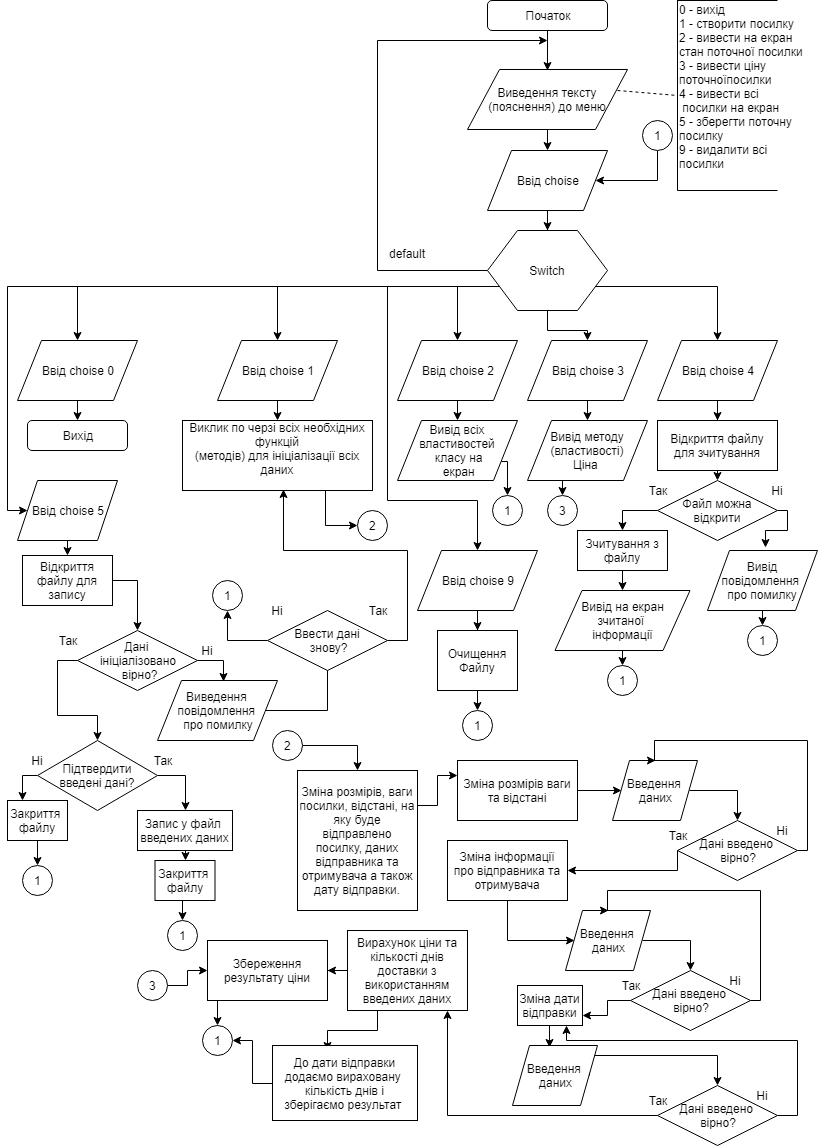
В класі (Інформація про отримувача) дані ідентичні вище зазначеному класу (Інформація про відправника).

**2.3 Алгоритм виконання роботи програми**

**Програма повинна давати користувачу такі можливості:**

* Внести всі дані для «створення» посилки.
* Переглянути введену інформацію після заповнення необхідних даних.
* Виводити(розраховану програмою після введення даних) вартість доставки.
* Переглядати весь каталог створених посилок, в т.ч і поточної.
* Збереження введених даних(після багатократної перевірки на правильність введених даних).
* Успішне видалення каталогу посилок.
* Вихід з програми.

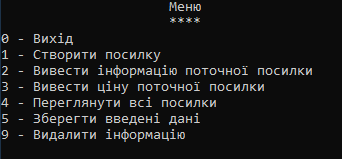
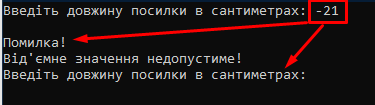
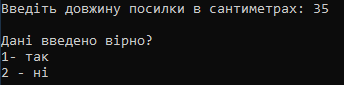
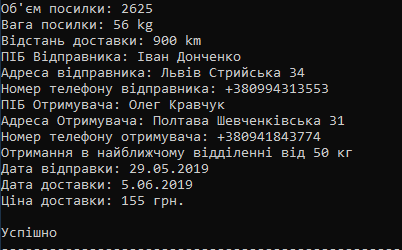
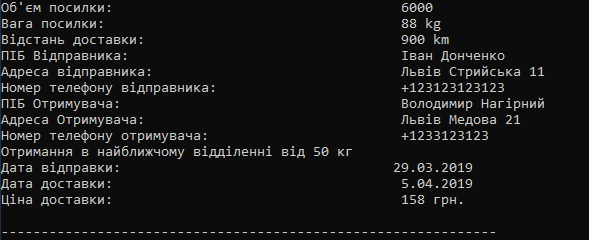
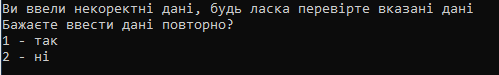
Хотів би також виділити багатократну перевірку вводу правильності даних, а також встановлених максимально допустимих значень, що майже унеможливлює допуску помилки зі сторони поштового відділення.



**3.** **Реалізація алгоритму у програмному коді**

Після складання алгоритму роботи моєї програми, я приступив до її реалізації програмним кодом. У своїй програмі я використовую 4 класи, а саме: Info\_Sender(містить інформацію про відправника а також методи доступу до цих даних), Info\_Orderer(містить інформацію про отримувача а також методи доступу до цих даних), Parcel\_info(містить інформацію безпосередньо про посилку, а також методи доступу до цих даних), Post(Містить методи обробки перевірки та встановлення обмежень на дані, які вводить користувач, а також методи розрахунку вартості та часу доставки також метод виклику всіх методів внесення даних, що унеможливлює пропуск одного чи кількох введень даних, також містить метод виводу на екран інформації її запис та зчитування з файлу). Виконання запису нових даних у файл відбувається таким чином: відкривається файл для запису, поелементно записуються дані, файл закривається, а при потребі можна переглянути записані дані у файл, одним із методів програми.

**4. Тестування програми**

1. При запуску програми вас зустірчає меню (Рис. 2):  
     
   Рис. 2
2. Вибравши 0 програма закриється (Рис. 3)  
     
   Рис. 3
3. Вибравши 1 запускається метод який викликає почергово методи класів для заповнення всіх необхідних даних (Рис. 4):  
     
   Рис. 4  
   В цьому випадку,( а також у всіх інших) програма не дозволить вам ввести некоректні дані, і в разі введення таких даних сповістить що дані невірні і буде просити ввести їх повторно (Рис. 5)  
     
   Рис. 5  
   Якщо дані введено вірно (для програми), програма попросить підтвердити правильність введених даних (Рис. 6)   
   /  
   Рис. 6  
   Якщо все введено вірно натискаємо 1 якщо ні – 2 (програма попросить ввести дані ще раз), в разі натиснення будь-якої іншої клавіші програма знову попросить підтвердити правильність вводу даних  
   Коли всі дані про посилку заповнені, паочинається ввід даних про відправника, а також про отримувача, яка в цілому аналогічна введенню даних про посилку.  
     
   Останній крок – введення дати відправки (Рис. 7)  
      
   Рис. 7  
   Хочу виділити, що програма враховує кількість днів у кожному місяці, і не дозволить ввести вам більше значення ніж можливе, також не дозволить ввести від’ємне або нульове значення. Після коректного введення дати програма поверне користувача до меню.
4. Вибравши 2 програма враховує дві ситуації 1 – якщо ми не вводили поки що нічого результат буде такий (Рис. 8):  
     
   Рис. 8  
   2 - Якщо ж ми ввели дані, то результат буде наступним (Рис. 9):   
     
   Рис. 9  
   Хочу зауважити, що вже на цьому етапі програма вирахувала ціну, а також кінцеву дату прибуття, яка залежить від відстані між населеними пунктами. Також програма в залежості від ваги посилки вказує, куди доставлять посилку.
5. Натиснувши 3 прогама теж відреагує двома можливими способами 1 – дані ще не введено (Рис. 10):  
     
   Рис. 10  
   2 – Всі дані введено і програма може опрацювати і вирахувати ціну (Рис 11):  
     
   Рис. 11  
   Хочу зазначити, що ціна буде різною для різних значень, оскільки вона вираховується автоматично в залежності від ваги, розміріві та відстані.
6. Якщо ми натиснемо 4 і у файлі є якісь дані, то програма виведе їх на екран (Рис. 12)   
     
   Рис. 12  
   Проте якщо у файлі немає нічого, то, відповідно, програма нічого не виведе.
7. Натиснувши 5 програма теж відреагує двояко: 1 – якщо ми ще нічого не ввели, результат виводу буде такий (Рис. 13):  
     
   Рис. 13  
   Натиснувши 2 ми повернемось до меню, а натиснувши 1 запуститься метод, що в меню знаходиться під номером 1, тобто ми почнемо вводити дані. В разі натиснення будь-якої іншої клавіші програма повторно виведе це ж повідомлення.  
   2 - Якщо ж ми ввели всі дані, то результат буде такий (Рис. 14):   
     
   Рис. 14  
   Натиснувши в цій ситуації 1 – програма запише введені дані у файл і поверне нас до меню, натиснувши 2 програма запустить метод, який в меню знаходиться під 1-м номером. В разі натиснення будь-якої іншої клавіші програма повторно виведе це ж повідомлення.
8. Натиснувши 9 ми очистимо каталог посилок, тобто очистимо файл.

**Висновок**

У процесі виконання поставленого перед мною завдання, а саме реалізація такої предметної галузі як поштове відділення, я закріпив свої знання з вивченого за один семестр курсу ООП. Закріпив на практиці знання із створення класів та утворення множинного наслідування.

З плюсів моєї програми можна виділити те , що вона має високу швидкодію, наявність простого у використанні консольного інтерфейсу який полегшує керування нею. А також максимального рівня перевірки правильності вводу даних, що майже унеможливлює допуск помилки при їх введенні.

Недоліком моєї програми можна вважати те, що вона не охоплює усіх аспектів керування поштового відділення.

В результаті виконання даної курсової роботи можна сказати, що на сьогоднішній день поштове відділення є потужним механізмом, який здійснює обслуговування великої кількості клієнтів. І саме тому правильне його функціонування є запорукою успішної доставки пошти великої кількості людей. Тому взявши всі плюси і мінуси можна сказати, що моя програма спрямована для підтримки правильності його функціонування.

**Література**

1. Armstrong, Deborah J. (February 2006). The Quarks of Object-Oriented Development. Communications of the ACM 49 (2). с. 123–128. ISSN 0001-0782.
2. Meyer, Bertrand (1997). Object-Oriented Software Construction. Prentice Hall. ISBN 0-13-629155-4.
3. Основы информатики и программирования: Учебное пособие// Е. А. Роганов — М.: МГИУ, 2001. — Вузол VI.(рос.)
4. Лавріщева К.М. Програмна інженерія—К.: Академперіодика, 2008.- 319 с.
5. http://programming.in.ua/programming/basisprogramming/25-oop.html
6. https://uk.wikipedia.org/wiki/Об%27єктноорієнтоване\_програмування#.D0.8D1.81.D1.82.D0.BE.D1.80.D1.96.D1.8F
7. https://uk.wikipedia.org/wiki/C%2B%2B
8. https://ru.wikipedia.org/wiki/Объектноориентированное\_программировани #.D0.98.D1.81.D1.82.D0.BE.D1.80.D0.B8.D1.8F
9. https://uk.wikipedia.org/wiki/Об%27єктно-орієнтоване\_програмування

10)http://damp.biz/plyusi-i-minusi-obyektno-oriyentovanogo-programuvannya/

**Додаток. Код програми.**

#include <iostream>

#include <string>

#include <windows.h>

#include <fstream>

using namespace std;

void succesfull() {

cout << "\nУспiшно\n--------------------------------------------------\n";

}

void deleteData()

{

ofstream fout("File.txt", ios::trunc);

}

void Show\_info() {

ifstream fin;

fin.open("File.txt");

if (!fin.is\_open()) {

cout << "Cannot open file fin" << endl;

}

else {

while (!fin.eof()) {

string str = "";

getline(fin, str);

cout << str << endl;

} }

fin.close();

}

class Info\_Sender {

private:

string sender\_name;

string sender\_surname;

string sender\_city;

string sender\_street;

string sender\_phone;

int sender\_house\_num;

public:

string Get\_sender\_name() {

return sender\_name;

}

string Get\_sender\_surname() {

return sender\_surname;

}

string Get\_sender\_city() {

return sender\_city;

}

string Get\_sender\_street() {

return sender\_street;

}

string Get\_sender\_phone() {

return sender\_phone;

}

int Get\_sender\_house\_num() {

return sender\_house\_num;

}

void Set\_sender\_name(string sn) {

sender\_name = sn;

}

void Set\_sender\_surname(string ss) {

sender\_surname = ss;

}

void Set\_sender\_city(string sc) {

sender\_city = sc;

}

void Set\_sender\_street(string ssr) {

sender\_street = ssr;

}

void Set\_sender\_house\_num(int shn) {

sender\_house\_num = shn;

}

void Set\_sender\_phone(string sp) {

sender\_phone = sp;

} };

class Info\_Orderer{

private:

string orderer\_name;

string orderer\_surname;

string orderer\_city;

string orderer\_street;

string orderer\_phone;

int orderer\_house\_num;

public:

string Get\_orderer\_name() {

return orderer\_name;

}

string Get\_orderer\_surname() {

return orderer\_surname;

}

string Get\_orderer\_city() {

return orderer\_city;

}

string Get\_orderer\_street() {

return orderer\_street;

}

string Get\_orderer\_phone() {

return orderer\_phone;

}

int Get\_orderer\_house\_num() {

return orderer\_house\_num;

}

void Set\_orderer\_name(string on) {

orderer\_name = on;

}

void Set\_orderer\_surname(string os) {

orderer\_surname = os;

}

void Set\_orderer\_city(string oc) {

orderer\_city = oc;

}

void Set\_orderer\_street(string osr) {

orderer\_street = osr;

}

void Set\_orderer\_house\_num(int ohn) {

orderer\_house\_num = ohn;

}

void Set\_orderer\_phone(string op) {

orderer\_phone = op;

} };

class Parcel\_info

{

private:

int weight;

int x;

int y;

int z;

int distance;

public:

int Get\_weight() {

return weight;

}

int Get\_x() {

return x;

}

int Get\_y() {

return y;

}

int Get\_z() {

return z;

}

int Get\_distance() {

return distance;

}

void Set\_weight(int value\_weight) {

weight = value\_weight;

}

void Set\_x(int value\_x) {

x = value\_x;

}

void Set\_y(int value\_y) {

y = value\_y;

}

void Set\_z(int value\_z) {

z = value\_z;

}

void Set\_distance(int value\_distnace) {

distance = value\_distnace;

} };

class Post : public Parcel\_info , public Info\_Sender, public Info\_Orderer

{

public:

Post() {

Set\_x(0);

Set\_y(0);

Set\_z(0);

Set\_distance(0);

Set\_weight(0);

Set\_sender\_name("");

Set\_sender\_surname("");

Set\_sender\_city("");

Set\_sender\_street("");

Set\_sender\_phone("");

Set\_sender\_house\_num(0);

Set\_orderer\_name("");

Set\_orderer\_surname("");

Set\_orderer\_city("");

Set\_orderer\_street("");

Set\_orderer\_phone("");

Set\_orderer\_house\_num(0);

}

int s\_day ;

int o\_day;

int o\_year;

int s\_year;

string month;

string o\_month;

int addtime;

////////////////////////////////////// Parcel info

void Change\_x() {

long int x;

cout << "Введiть довжину посилки в сантиметрах: ";

cin >> x;

cout << endl;

if (x <= 0) {

cout << "Помилка!\nВiд'ємне значення недопустиме!\n";

Change\_x();

}

else if (x >= 1000) {

cout << "\nМаксимальна довжина посилки 10м\n";

Change\_x();

}

else {

Set\_x(x);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1: succesfull(); break;

default: Change\_x(); break;

} } }

void Change\_y() {

long int y;

cout << "Введiть висоту посилки в сантиметрах: ";

cin >> y;

cout << endl;

if (y <= 0) {

cout << "Помилка!\nВiд'ємне значення недопустиме!\n";

Change\_y();

}

else if (y >= 300) {

cout << "\nМаксимальна висота посилки 3м\n";

Change\_y();

}

else {

Set\_y(y);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1: succesfull(); break;

default: Change\_y(); break;

} } }

void Change\_z() {

long int z;

cout << "Введiть ширину посилки в сантиметрах: ";

cin >> z;

cout << endl;

if (z <= 0) {

cout << "Помилка!\nВiд'ємне значення недопустиме!\n";

Change\_z();

}

else if (z >= 500) {

cout << "\nМаксимальна ширина посилки 5м\n";

Change\_z();

}

else {

Set\_z(z);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1: succesfull(); break;

default: Change\_z(); break;

} } }

void Change\_weight() {

long int w;

cout << "Введiть вагу посилки в кг: ";

cin >> w;

cout << endl;

if (w <= 0) {

cout << "Помилка!\nВiд'ємне значення недопустиме!\n";

Change\_weight();

}

else if (w >= 5000) {

cout << "\nМаксимальна вага посилки 5т(5000кг)\n";

Change\_weight();

}

else {

Set\_weight(w);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1:succesfull(); break;

default: Change\_weight(); break;

} } }

void Change\_distance() {

long int d;

cout << "Введiть вiдстань мiж пунктами призначення в км: ";

cin >> d;

cout << endl;

if (d <= 0) {

cout << "Помилка!\nВiд'ємне значення недопустиме!\n";

Change\_distance();

}

else {

Set\_distance(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1: succesfull(); break;

default: Change\_distance(); break;

} } }

///////////////////////////////////// Sender info

void Change\_sender\_name() {

string d;

cout << "Введiть ваше iм'я: ";

cin >> d;

cout << endl;

Set\_sender\_name(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1: succesfull(); break;

default: Change\_sender\_name(); break;

} }

void Change\_sender\_phone() {

string d;

cout << "Введiть ваш номер телефону: ";

cin >> d;

cout << endl;

Set\_sender\_phone(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1:succesfull(); break;

default: Change\_sender\_phone(); break;

} }

void Change\_sender\_surname() {

string d;

cout << "ВВедiть ваше прiзвище: ";

cin >> d;

cout << endl;

Set\_sender\_surname(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1: succesfull(); break;

default: Change\_sender\_surname(); break;

} }

void Change\_sender\_city() {

string d;

cout << "Введiть ваше мiсто: ";

cin >> d;

cout << endl;

Set\_sender\_city(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1:succesfull(); break;

default: Change\_sender\_city(); break;

} }

void Change\_sender\_street() {

string d;

cout << "Введiть назву вашої вулицi:";

cin >> d;

cout << endl;

Set\_sender\_street(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1:succesfull(); break;

default: Change\_sender\_street(); break;

} }

void Change\_sender\_house\_num() {

long int d;

cout << "Введiть номер вашого будинку:";

cin >> d;

cout << endl;

if (d < 0) {

cout << "Помилка!\nВiд'ємне значення недопустиме!\n";

Change\_sender\_house\_num();

}

else {

Set\_sender\_house\_num(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1: succesfull(); break;

default: Change\_sender\_house\_num(); break;

} } }

////////////////////////////////////// Date

void Date() {

cout << "Введiть день, мiсяць, рiк вiдправки: ";

cin >> s\_day;

cin >> month;

cin >> s\_year;

o\_year = s\_year;

o\_month = month;

int b = stoi(month);

if ((s\_day <= 0) || b <= 0 || b > 12) {

cout << "\nПомилка!\n";

Date();

}

else {

if (Get\_distance() <= 300) {

addtime = 2;

}

else if ((Get\_distance() > 300) && Get\_distance() <= 600) {

addtime = 4;

}

else if ((Get\_distance() > 600) && Get\_distance() <= 1000) {

addtime = 7;

}

else {

addtime = 10;

}

if (b == 1) {

int max = 31;

if (s\_day <= max) {

if ((s\_day + addtime) > max) {

o\_day = (s\_day + addtime) - max;

o\_month = "02";

}

else {

o\_day = s\_day + addtime;

} }

else {

cout << "\nПомилка!\n";

Date();

} }

else if (b == 2) {

int max = 28;

if (s\_day <= max) {

if ((s\_day + addtime) > max) {

o\_day = (s\_day + addtime) - max;

o\_month = "03";

}

else {

o\_day = s\_day + addtime;

} }

else

{

cout << "\nПомилка!\n";

Date();

} }

else if (b == 3) {

int max = 31;

if (s\_day <= max) {

if ((s\_day + addtime) > max) {

o\_day = (s\_day + addtime) - max;

o\_month = "04";

}

else {

o\_day = s\_day + addtime;

} }

else

{

cout << "\nПомилка!\n";

Date();

} }

else if (b == 4) {

int max = 30;

if (s\_day <= max) {

if ((s\_day + addtime) > max) {

o\_day = (s\_day + addtime) - max;

o\_month = "05";

}

else {

o\_day = s\_day + addtime;

} }

else

{

cout << "\nПомилка!\n";

Date();

} }

else if (b == 5) {

int max = 31;

if (s\_day <= max) {

if ((s\_day + addtime) > max) {

o\_day = (s\_day + addtime) - max;

o\_month = "06";

}

else {

o\_day = s\_day + addtime;

} }

else

{

cout << "\nПомилка!\n";

Date();

} }

else if (b == 6) {

int max = 30;

if (s\_day <= max) {

if ((s\_day + addtime) > max) {

o\_day = (s\_day + addtime) - max;

o\_month = "07";

}

else {

o\_day = s\_day + addtime;

} }

else

{

cout << "\nПомилка!\n";

Date();

} }

else if (b == 7) {

int max = 31;

if (s\_day <= max) {

if ((s\_day + addtime) > max) {

o\_day = (s\_day + addtime) - max;

o\_month = "08";

}

else {

o\_day = s\_day + addtime;

}

}

else

{

cout << "\nПомилка!\n";

Date();

} }

else if (b == 8) {

int max = 31;

if (s\_day <= max) {

if ((s\_day + addtime) > max) {

o\_day = (s\_day + addtime) - max;

o\_month = "09";

}

else {

o\_day = s\_day + addtime;

} }

else

{

cout << "\nПомилка!\n";

Date();

} }

else if (b == 9) {

int max = 30;

if (s\_day <= max) {

if ((s\_day + addtime) > max) {

o\_day = (s\_day + addtime) - max;

o\_month = "10";

}

else {

o\_day = s\_day + addtime;

} }

else

{

cout << "\nПомилка!\n";

Date();

} }

else if (b == 10) {

int max = 31;

if (s\_day <= max) {

if ((s\_day + addtime) > max) {

o\_day = (s\_day + addtime) - max;

o\_month = "11";

}

else {

o\_day = s\_day + addtime;

} }

else

{

cout << "\nПомилка!\n";

Date();

} }

else if (b == 11) {

int max = 30;

if (s\_day <= max) {

if ((s\_day + addtime) > max) {

o\_day = (s\_day + addtime) - max;

o\_month = "12";

}

else {

o\_day = s\_day + addtime;

} }

else

{

cout << "\nПомилка!\n";

Date();

} }

else if (b == 12) {

int max = 31;

if (s\_day <= max) {

if ((s\_day + addtime) > max) {

o\_day = (s\_day + addtime) - max;

o\_month = "01";

o\_year = s\_year + 1;

}

else {

o\_day = s\_day + addtime;

} }

else

{ cout << "\nПомилка!\n";

Date(); } }

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1: succesfull(); break;

default: Date(); break;

} } }

////////////////////////////////////// Orderer info

void Change\_orderer\_name() {

string d;

cout << "Введiть iм'я отримувача: ";

cin >> d;

cout << endl;

Set\_orderer\_name(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1:succesfull(); break;

default: Change\_orderer\_name(); break;

} }

void Change\_orderer\_phone() {

string d;

cout << "Введiть номер телефону отримувача: ";

cin >> d;

cout << endl;

Set\_orderer\_phone(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1:succesfull(); break;

default: Change\_orderer\_phone(); break;

} }

void Change\_orderer\_surname() {

string d;

cout << "Введiть прiзвище отримувача: ";

cin >> d;

cout << endl;

Set\_orderer\_surname(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1:succesfull(); break;

default: Change\_orderer\_surname(); break;

} }

void Change\_orderer\_city() {

string d;

cout << "Введiть мiсто отримувача:";

cin >> d;

cout << endl;

Set\_orderer\_city(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1:succesfull(); break;

default: Change\_orderer\_city(); break;

} }

void Change\_orderer\_street() {

string d;

cout << "Введiть назву вулицi отримувача:";

cin >> d;

cout << endl;

Set\_orderer\_street(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1:succesfull(); break;

default: Change\_orderer\_street(); break;

} }

void Change\_orderer\_house\_num() {

long int d;

cout << "Введiть номер будинку отримувача:";

cin >> d;

cout << endl;

if (d < 0) {

cout << "Помилка!\nВiд'ємне значення недопустиме!\n";

Change\_orderer\_house\_num();

}

else {

Set\_orderer\_house\_num(d);

cout << "Данi введено вiрно?\n1- так\n2 - нi" << endl;

int c;

cin >> c;

switch (c) {

case 1: succesfull(); break;

default: Change\_orderer\_house\_num(); break;

} } }

///////////////////////////////////////// Func

int Price() {

int price;

int a;

if (Get\_x() <= 0 || Get\_y() <= 0 || Get\_z() <= 0 || Get\_distance() <= 0 || Get\_weight() <= 0 || Get\_sender\_name() == "" || Get\_sender\_surname() == "" || Get\_sender\_city() == "" || Get\_sender\_street() == "" || Get\_sender\_phone() == "" || Get\_sender\_house\_num() <= 0 || Get\_orderer\_name() == "" || Get\_orderer\_surname() == "" || Get\_orderer\_city() == "" || Get\_orderer\_street() == "" || Get\_orderer\_phone() == "" || Get\_orderer\_house\_num() <= 0) {

return 0;

}

else

{

if (Get\_distance() <= 300) {

a = 40;

}

else if ((Get\_distance() > 300) && Get\_distance() <= 600) {

a = 90;

}

else if ((Get\_distance() > 600) && Get\_distance() <= 1000) {

a = 150;

}

else {

a = 250;

}

price = a + (Get\_weight() / 10);

return price;

} }

void Create() {

Change\_x();

system("cls");

Change\_y();

system("cls");

Change\_z();

system("cls");

Change\_weight();

system("cls");

Change\_distance();

system("cls");

Change\_sender\_name();

system("cls");

Change\_sender\_surname();

system("cls");

Change\_sender\_city();

system("cls");

Change\_sender\_street();

system("cls");

Change\_sender\_house\_num();

system("cls");

Change\_sender\_phone();

system("cls");

Change\_orderer\_name();

system("cls");

Change\_orderer\_surname();

system("cls");

Change\_orderer\_city();

system("cls");

Change\_orderer\_street();

system("cls");

Change\_orderer\_house\_num();

system("cls");

Change\_orderer\_phone();

system("cls");

Date();

}

void Print() {

if (Get\_x() <= 0 || Get\_y() <= 0 || Get\_z() <= 0 || Get\_distance() <= 0 || Get\_weight() <= 0 || Get\_sender\_name() == "" || Get\_sender\_surname() == "" || Get\_sender\_city() == "" || Get\_sender\_street() == "" || Get\_sender\_phone() == "" || Get\_sender\_house\_num() <= 0 || Get\_orderer\_name() == "" || Get\_orderer\_surname() == "" || Get\_orderer\_city() == ""

|| Get\_orderer\_street() == "" || Get\_orderer\_phone() == "" || Get\_orderer\_house\_num() <= 0)

{

cout << "Немає поточної посилки або дані введено невірно!\n";

}

else

{

cout << "Об'єм посилки: " << (((Get\_x())\*(Get\_y())\*(Get\_z())) / 100) << endl;

cout << "Вага посилки: " << Get\_weight() << " kg" << endl;

cout << "Вiдстань доставки: " << Get\_distance() << " km" << endl;

cout << "ПIБ Вiдправника: " << Get\_sender\_name() << " " << Get\_sender\_surname() << endl;

cout << "Адреса вiдправника: " << Get\_sender\_city() << " " << Get\_sender\_street() << " " << Get\_sender\_house\_num() << endl;

cout << "Номер телефону вiдправника: " << Get\_sender\_phone() << endl;

cout << "ПIБ Отримувача: " << Get\_orderer\_name() << " " << Get\_orderer\_surname() << endl;

cout << "Адреса Отримувача: " << Get\_orderer\_city() << " " << Get\_orderer\_street() << " " << Get\_orderer\_house\_num() << endl;

cout << "Номер телефону отримувача: " << Get\_orderer\_phone() << endl;

if (Get\_weight() <= 50) {

cout << "Отримання в найближчому вiддiленнi до 50 кг";

}

else { cout << "Отримання в найближчому вiддiленнi вiд 50 кг"; }

cout << "\nДата вiдправки: " << s\_day << "." << month << "." << s\_year << endl;

cout << "Дата доставки: " << o\_day << "." << o\_month << "." << o\_year << endl;

cout << "Цiна доставки: " << Price() << " грн." << endl;

succesfull();

} }

void SaveData()

{

ofstream fout("File.txt", ofstream::app);

if (Get\_x() <= 0 || Get\_y() <= 0 || Get\_z() <= 0 || Get\_distance() <= 0 || Get\_weight() <= 0 || Get\_sender\_name() == "" || Get\_sender\_surname() == "" || Get\_sender\_city() == ""

|| Get\_sender\_street() == "" || Get\_sender\_phone() == "" || Get\_sender\_house\_num() <= 0

|| Get\_orderer\_name() == "" || Get\_orderer\_surname() == "" || Get\_orderer\_city() == ""

|| Get\_orderer\_street() == "" || Get\_orderer\_phone() == "" || Get\_orderer\_house\_num() <= 0) {

cout << "Ви ввели некоректні дані, будь ласка перевірте вказані дані" << endl;

fout.close();

cout << "Бажаєте ввести дані повторно?\n1 - так\n2 - ні" << endl;

int tem;

cin >> tem;

switch (tem) {

case 1:system("cls"); Create(); break;

case 2:system("cls"); break;

default:system("cls"); SaveData(); break;

} }

else

{

cout << "Ви дійсно хочете зберегти введені дані? Після підтвердження змінити їх не можливо" << endl;

cout << "1 - Підтвердити зміни(скасувати не можливо)\n2 - Ввести дані повторно" << endl;

int tem;

cin >> tem;

switch (tem)

{ case 1:

fout.setf(ios::left);

fout << "\n--------------------------------------------------------------\n";

fout.width(50);

fout << "Об'єм посилки: " << (((Get\_x())\*(Get\_y())\*(Get\_z())) / 100) << endl;

fout.width(50);

fout << "Вага посилки: " << Get\_weight() << " kg" << endl;

fout.width(50);

fout << "Вiдстань доставки: " << Get\_distance() << " km" << endl;

fout.width(50);

fout << "ПIБ Вiдправника: " << Get\_sender\_name() << " " << Get\_sender\_surname() << endl;

fout.width(50);

fout << "Адреса вiдправника: " << Get\_sender\_city() << " " << Get\_sender\_street() << " " << Get\_sender\_house\_num() << endl;

fout.width(50);

fout << "Номер телефону вiдправника: " << Get\_sender\_phone() << endl;

fout.width(50);

fout << "ПIБ Отримувача: " << Get\_orderer\_name() << " " << Get\_orderer\_surname() << endl;

fout.width(50);

fout << "Адреса Отримувача: " << Get\_orderer\_city() << " " << Get\_orderer\_street() << " " << Get\_orderer\_house\_num() << endl;

fout.width(50);

fout << "Номер телефону отримувача: " << Get\_orderer\_phone() << endl;

if (Get\_weight() <= 50)

{ fout << "Отримання в найближчому вiддiленнi до 50 кг"; }

else

{ fout << "Отримання в найближчому вiддiленнi вiд 50 кг"; }

fout.width(50);

fout << "\nДата вiдправки: " << s\_day << "." << month << "." << s\_year << endl;

fout.width(50);

fout << "Дата доставки: " << o\_day << "." << o\_month << "." << o\_year << endl;

fout.width(50);

fout << "Цiна доставки: " << Price() << " грн." << endl;

fout.close();

break;

case 2: fout.close(); break;

default: SaveData(); break;

} } } };

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Post a;

int c;

bool b = true;

while (b)

{

cout.width(25);

cout << "Меню" << endl;

cout.width(25);

cout << "\*\*\*\*" << endl;

cout << "0 - Вихiд\n1 - Створити посилку\n2 - Вивести iнформацiю поточної посилки\n3 - Вивести цiну поточної посилки\n4 - Переглянути всi посилки\n5 - Зберегти введені дані\n9 - Видалити iнформацiю\n";

cin >> c;

cout << endl;

switch (c)

{ case 0:b = false; system("cls"); break;

case 1:system("cls"); a.Create(); system("cls"); break;

case 2: system("cls"); a.Print(); break;

case 3:system("cls"); cout <<"Ціна: "<< a.Price() << endl; break;

case 4:system("cls"); Show\_info(); break;

case 5:system("cls"); a.SaveData(); break;

case 9:system("cls"); deleteData(); break;

default: break; }

}

system("pause");

return 0;

}