**Львівський національний аграрний університет Факультет механіки та енергетики**

*кафедра інформаційних систем та технологій*

**КУРСОВА РОБОТА**

**з дисципліни «Об’єктно-орієнтоване програмування»**

На тему: Розроблення прикладної програми для сервісу ‘Оренда велосипедів’.

Студент 1-го курсу ІТ-12СП групи денної форми навчання Механічного факультету

Шоп’як Богдан Ігорович

Науковий керівник Укладач: в.о. доц. А.В. Татомир

Дубляни 2021

**Зміст**

[**Розділ 1. Аналіз стану питання та обґрунтування завдання.** 4](#_Toc72085288)

[**1.1Аналіз предмету проектування.** 4](#_Toc72085289)

[**1.2Класифікація об’єктно-орієнтованих мов програмування.** 6](#_Toc72085290)

[**1.3Огляд та аналіз сучасних технологій та засобів проектування програмного забезпечення.** 8](#_Toc72085291)

[**1.4** **Уточнена постановка задачі на розробку програмного забезпечення.** 9](#_Toc72085292)

[**Розділ 2. Побудова загальної структури програми.** 11](#_Toc72085293)

[2.1 Аналіз функцій системи. 11](#_Toc72085294)

[2.2 Розроблення структурної моделі системи. 11](#_Toc72085295)

[**Розділ 3. Розроблення ієрархії класів.** 13](#_Toc72085296)

[3.1 Декомпозиція системи. 13](#_Toc72085297)

[3.2 Аналіз взаємозв’язків між об’єктами. 15](#_Toc72085298)

[3.3 Розроблення інтерфейсів класів. 16](#_Toc72085299)

[3.4 UML – діаграма класів. 17](#_Toc72085300)

[**Розділ 4. Розроблення системи вводу-виводу даних.** 18](#_Toc72085301)

[**Розділ 5. Тестування програмного забезпечення.** 19](#_Toc72085302)

[**Розділ 6. Розроблення документації для супроводження програмного продукту.** 20](#_Toc72085303)

[**Висновки** 23](#_Toc72085304)

[**Список використаних джерел.** 24](#_Toc72085305)

**Вступ**

Курсова робота присвячена моделюванню роботи додатка для оренди велосипедів, у складі: набір об’єктних моделей, програмний код. Виконані усі необхідні роботи з проектування моделювання, аналізуються вимоги до неї, описано особливості реалізації, кодування, тестування програми. При написанні програми використано об’єктно-орієнтовану мову Phyton.

В даній курсовій роботі реалізується моделювання оренди велосипедів. Зміст моделювання полягає у тому, що користувач вибирає потрібну йому дію: оренда чи повернення (після оренди).

Призначення програми – оренда велосипеда за зручним інтерфейсом.

Для кодування програми буде обрана мова Phyton що сприяє написанню надійної програми.

# **Розділ 1. Аналіз стану питання та обґрунтування завдання.**

**1.1Аналіз предмету проектування.**

Сучасні додатки інформаційних технологій дозволити підприємствам конкурувати новаторськими способами. Якщо у минулому фірми повинні були вибрати між стратегіями вартості або диференціювання, то сьогодні інформаційні технології дозволяють фірмам в деяких галузях промисловості конкурувати з низькими цінами і диференціюванням виробів одночасно. Деякі фірми намагаються конкурувати не тільки з низькими цінами і високою якістю, але також здатні робити високо якісні вироби з урахуванням вимог користувача. Названі як «масові настройки» інформаційних технологій використовуються, щоб швидко пов'язати процеси і робочі групи, виготовити конкурентоздатні вироби, які є саме такими, в яких клієнт має потребу.

Однією з найважливіших вимог до методології проектування сучасних інформаційної технології, на думку О. М. Томашевського , є саме забезпечення динамічності її структури і функцій. Тому великого значення набуває визначення інваріантів технологій, тобто опорних структур даних і процесів, які можуть бути досить стабільними властивостями і характеристиками гнучкої інформаційної технології.

**Об'єктно - орієнтоване програмування (ООП)** – це модель програмування яка базується на стверджені того, що програма це сукупність об’єктів які взаємодіють між собою. Кожен об’єкт в цій моделі є незалежним, і він здатний отримувати, обробляти дані та відправляти ці дані іншим об’єктам. В ООП використано моделі успадкування, модульності, поліморфізму та інкапсуляції.

Основним поняттям ООП є об'єкт. **Об'єкт** можна визначити як певну сукупність даних(характеристик об'єкта) та методів роботи з ними. Для класифікації об'єктів у ООП використовують класи. **Клас** служить зразком для створення обєкту, тобто об'єкт є нічим іншим, ніж копією класу.

Кожен об'єкт має процедури і функції(те що він уміє виконувати, наприклад,завантажувати файл, відображати картинку і т.д.), які служать для роботи з даними об'єкта. Ці процедури і функції називаються **методами**.

Існування ООП можливе завдяки трьом основним парадигмам **на яких базується саме ООП:**

- **Інкапсуляція**. Також відома як приховування даних. Зміст інкапсуляції полягає у приховуванні від зовнішнього користувача деталей реалізації об'єкта, замість цього надаючи інтерфейс взаємодії з ним.

- **Успадкування.** Це означає, що об'єкти (класи) можуть переймати деякі властивості у своїх прабатьків. Як? Це залежить від тієї мови, на якому пишеться програма. Однак у будь-якому випадку картина та ж: це призводить до повторного використання вже написаного одного разу коду. Підкласи успадковують атрибути та поведінку своїх батьківських класів, і можуть мати нові власні атрибути. Тобто утворюється ієрархія з класів, де від основного класу(так званого, предка) походять усі інші класи. Приклад такого розгалуженого "дерева" зображено на схемі.

Поліморфізм означає залежність поведінки від класу, в якому ця поведінка викликається, тобто, два або більше класів можуть реагувати по різному на однакові повідомлення. Це спричинене зміною в одного з класів якогось методу(процедури, функції), шляхом запису іншого алгоритму. Як приклад, деяка комп'ютерна програма при натисканні клавіші Esc завершить роботу, інша ж програма після натискання кнопки Esc тільки відкриє меню даної програми.

Завдання полягає у створенні програми для спрощення роботи з орендою велосипедів.

Вирішення проблеми та створення програми відбувається у мові програмування Python.

**Python** (найчастіше вживане прочитання — «**Па́йтон**») — [інтерпретована](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D1%80%D0%B5%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [об'єктно-орієнтована](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [мова програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) високого рівня зі [строгою](https://uk.wikipedia.org/w/index.php?title=%D0%A1%D1%82%D1%80%D0%BE%D0%B3%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F&action=edit&redlink=1) [динамічною типізацією](https://uk.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D1%96%D1%87%D0%BD%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F).[[8]](https://uk.wikipedia.org/wiki/Python#cite_note-8) Розроблена в [1990](https://uk.wikipedia.org/wiki/1990) році [Гвідо ван Россумом](https://uk.wikipedia.org/wiki/%D0%93%D0%B2%D1%96%D0%B4%D0%BE_%D0%B2%D0%B0%D0%BD_%D0%A0%D0%BE%D1%81%D1%81%D1%83%D0%BC). [Структури даних](https://uk.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднування наявних компонентів. Python підтримує [модулі](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D1%8C_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python та стандартні бібліотеки доступні як у скомпільованій, так і у вихідній формі на всіх основних платформах. В мові програмування Python підтримується кілька [парадигм програмування](https://uk.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F),зокрема: [об'єктноорієнтована](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), [процедурна](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), [функціональна](https://uk.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D1%96%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) та [аспектно-орієнтована](https://uk.wikipedia.org/wiki/%D0%90%D1%81%D0%BF%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F).

Метою завдання є вирішення проблеми з орендою велосипедів а можливо і чогось іншого.

Цей застосунок призначений для підприємств,магазинів,компаній які що небудь орендують на основі цієї програми можна створити аналог іншої оренди чого небудь.

**1.2Класифікація об’єктно-орієнтованих мов програмування.**

Перш ніж ви захочете вивчати якусь мову програмування треба знати трохи про їхню історію і де вони застосовуються.

Я представляю вам короткий огляд 25 відомих мов програмування. Починаючи з самого відомого до менш популярного. Стаття зроблена для новачків в сфері програмування. Ви можете почитати про кожній мові і вибрати той який вам більше сподобатися для вивчення.

Перш ніж приступити до вивчення мов програмування рекомендую вам вивчити курс по.

**1 JavaScript.** Прототипна-орієнтований сценарний мову програмування. JavaScript спочатку створювався для того, щоб зробити web-сторінки «живими». У браузері вони підключаються безпосередньо до HTML і, як тільки завантажується сторінка - тут же виконуються.

коли створювався [мова JavaScript](https://remzhuk.ru/uk/kak-vklyuchit-yazyk-javascript-kak-vklyuchit-javascript-v-brauzere-instrukciya-dlya-vseh/), У нього спочатку була інша назва: «LiveScript». Але тоді був дуже популярний мову Java, і маркетологи вирішили, що схожу назву зробить нову мову більш популярним.

Планувалося, що JavaScript буде таким собі «молодшим братом» Java. Однак, історія розпорядилася по-своєму, JavaScript сильно виріс, і зараз це абсолютно незалежний мову, зі своєю специфікацією і до Java не має ніякого відношення.

**2 Java.** Строго типізований об'єктно-орієнтована мова програмування. Програми Java зазвичай транслюються в спеціальний байт-код, тому вони можуть працювати на будь-якої комп'ютерної архітектурі, за допомогою віртуальної Java-машини.

Перевагою подібного способу виконання програм є повна незалежність байт-коду від операційної системи і устаткування, що дозволяє виконувати Java-додатки на будь-якому пристрої, для якого існує відповідна віртуальна машина. Іншою важливою особливістю технології Java є гнучка система безпеки, в рамках якої виконання програми повністю контролюється віртуальною машиною.

Спочатку мова називалася Oak ( «Дуб») розроблявся Джеймсом Гослінгом для програмування побутових електронних пристроїв. Згодом він був перейменований в Java і став використовуватися для написання клієнтських додатків і серверного [програмного забезпечення](https://remzhuk.ru/uk/programmnoe-obespechenie-sistem-upravleniya-dispetcherizacii-i-asutp-apparatno-programmnyi-kompleks-d/).

**3 PHP.** Є поширеним інтерпретується мовою [загального призначення](https://remzhuk.ru/uk/naznachenie-bluetooth-obshchie-principy-postroeniya-bluetooth-setei-peredacha-dannyh-v-bluetooth/) з відкритим [вихідним кодом](https://remzhuk.ru/uk/skachat-programmu-fonarik-na-russkom-gde-skachat-besplatnyi/) (Скриптова мова). PHP створювався спеціально для ведення web-розробок і код на ньому може впроваджуватися безпосередньо в HTML-код. Синтаксис мови бере початок з C, Java і Perl, і є легким для вивчення.

Основною метою PHP є надання web-розробникам можливості [швидкого створення](https://remzhuk.ru/uk/programma-dlya-sozdaniya-kollekcii-programm-katalogizatory-bystryi/) динамічно генеруються web-сторінок, однак сфера застосування PHP не обмежується тільки цим.

**4.Python.** Високорівнева мова програмування загального призначення, орієнтований на підвищення продуктивності розробника, читання коду і на розробку веб додатків. Синтаксис ядра Python мінімалістичний. Код в Python організовується у функції та класи, які можуть об'єднуватися в модулі.

**1.3Огляд та аналіз сучасних технологій та засобів проектування програмного забезпечення.**

Основні функції операційних систем (ОС) полягають в управлінні ресурсами (фізичними та логічними) і процесами обчислювальних систем. Фізичними ресурсами є: оперативна пам'ять, процесор, монітор, принтер, магнітні та оптичні диски. До логічних ресурсів можна віднести програми, файли, події тощо. Під процесом розуміється деяка послідовність дій, предписана відповідною програмою і використовуваними нею даними.

Мережеве ПЗ призначено для управління загальними ресурсами у розподілених обчислювальних системах. До мережевого ПЗ відносять ОС, що підтримують роботу комп'ютерів в мережевих конфігураціях (так звані мережеві ОС), а також окремі мережеві програми (пакети), що використовуються спільно зі звичайними, не мережевими ОС.

Для розширення можливостей операційних систем та надання набору додаткових послуг використовуються сервісні програми. Їх можна розділити на наступні групи:

* інтерфейсні системи;
* оболонки операційних систем;
* утиліти.
* Інтерфейсні системи є природним продовженням операційної системи і модифікують як користувальницький, так і програмний інтерфейси, а також реалізують додаткові можливості з управління ресурсами комп'ютера. У зв'язку з тим, що розвинена інтерфейсна система може змінити весь призначений для користувача інтерфейс, часто їх також називають операційними системами.
* Оболонки операційних систем, на відміну від інтерфейсних систем, модифікують тільки користувальницький інтерфейс, надаючи користувачеві якісно новий інтерфейс у порівнянні з реалізованим операційною системою. Такі системи істотно спрощують виконання часто запитуваних функцій, наприклад, таких операцій з файлами, як копіювання, перейменування та знищення, а також пропонують користувачеві ряд додаткових послуг. У цілому, програми-оболонки помітно підвищують рівень користувальницького інтерфейсу, найповніше задовольняючи потреби користувача.
* Утиліти надають користувачам засоби обслуговування комп'ютера і його ПЗ. Вони забезпечують реалізацію наступних дій:
* обслуговування магнітних дисків;
* обслуговування файлів і каталогів;
* надання інформації про ресурси комп'ютера;
* шифрування інформації;
* захист від комп'ютерних вірусів;
* архівація файлів тощо.
  1. **Уточнена постановка задачі на розробку програмного забезпечення.**

На цього аналізу та огляду літературних джерел можна сформулювати постановку задачі: розробити ПЗ для оренди велосипедів ,основними функціями якої є: авторизація як адміністратор або користувач, оренда велосипедів , а також їх додавання та видалення їх з головного списку.

Створити інтерфейс для можливості швидкого та зручного отримання доступу до потрібних даних у програмі.

Для розробки продукту потрібно орієнтуватися на те, що користувач цієї програми може мати початкові навички роботи з ПК, тому інтерфейс програми повинен бути якомога простішим та зручним.

# **Розділ 2. Побудова загальної структури програми.**

## 2.1 Аналіз функцій системи.

Цей продукт керує даними магазину оренди та надає їх користувачу.

Система використовується для дачі в оренду велосипедної техніки або ж оперування нею.

Вона включає наступні пункти:

* Авторизація
* Перегляд дійсних велосипедів
* Оренда вільних велосипедів
* Повернення оренди
* Додавання велосипедів
* видалення даних

## 2.2 Розроблення структурної моделі системи.

В програмі «Оренда велосипедів» застосовується основний клас Bikerental

Цей клас включає в себе такі функції:

* rent\_bike\_hourly - Видає людині велосипеди погодинно
* rent\_bike\_daily - Видає людині велосипеди на день
* rent\_bike\_weekly – Видає людині велосипед на тиждень
* add – приймає кількість нових велосипедів та додає їх до основної кількості
* min - приймає кількість застарілих або поламаних велосипедів які потрібно списати
* issue\_bill - Видає клієнту рахунок на основі імені клієнта

Структурна схема програми «Оренда велосипедів»

Користувач

Адміністратор

Меню користувача

Меню адміністратора

Перегляд вільних велосипедів

Перегляд вільних велосипедів

Оренда

Оренда

Повернення

Повернення

Кількість нових

Додавання велосипедів

Загальна кількість

Видалення велосипедів

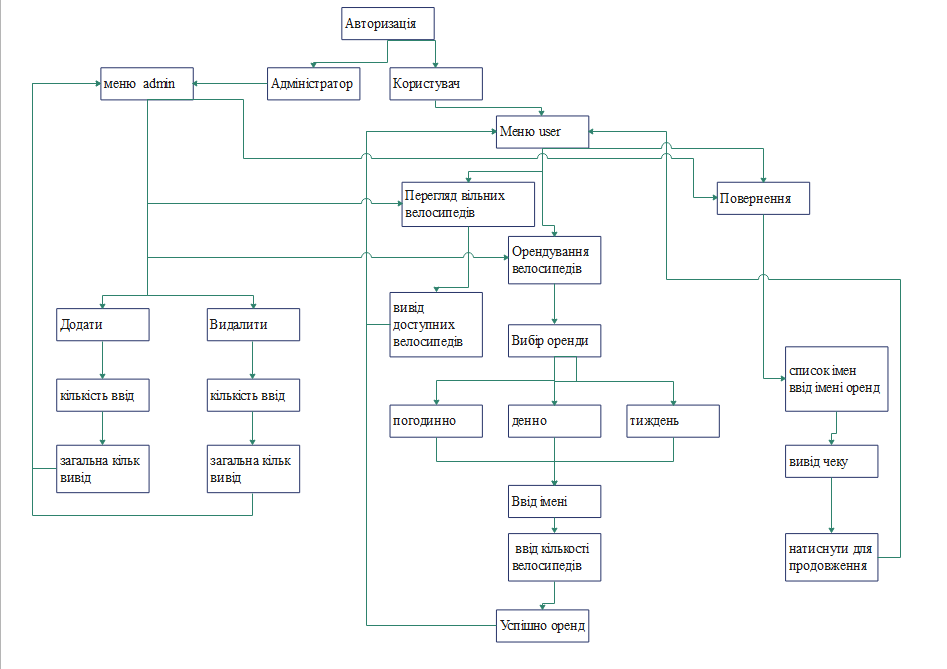
Скільки видалити

Загальна кількість

# **Розділ 3. Розроблення ієрархії класів.**

## 3.1 Декомпозиція системи.

3.1.1 Блок - схема програми «Оренда велосипедів»



Можна виділити два інструменти управління складністю в програмуванні:

* абстракція
* декомпозиція

**Абстрагування** - зосередження на важливих сторонах і відмова від неважливих.

**Декомпозиція** - поділ цілого на частини.

Один з інструментів побудови абстракцій та декомпозиції задач в мові Python - функції. При розробці своїх функцій рекомендується керуватися наступними принципами:

Принцип єдиної відповідальності (The Single Responsibility Principle, SRP). Функція або клас повинні вирішувати тільки одну задачу.

По можливості намагайтеся розробляти чисті функції.

Чистий функція - це функція, результат роботи якої залежить тільки від вхідних аргументів.

Функція не є чистою, якщо в ній присутній введення-виведення або звернення до глобальних змінних.

Do not repeat yourself (DRY). Кожна частина знання повинна мати єдине, несуперечливе і авторитетне уявлення в рамках системи. У простих випадках порушення цього принципу проявляється в дублюванні коду, використанні різних неназваних констант (magic numbers).

## 

## 3.2 Аналіз взаємозв’язків між об’єктами.

У об’єктно-орієнтованому програмуванні існує ієрархія класів та об’єктів.

Об’єкти описуються за допомогою класів. Об’єкт це сутність зі своїми властивостями та характеристиками, яка може реагувати на повідомлення за допомогою своїх існуючих даних та з’являється при створення екземпляра класу або прототипу.

Клас – це список характеристик та можливостей об’єкта.

У об’єктно-орієнтованому програмуванні існують такі зв’язки між класами та об’єктами:

* + асоціація;
  + успадкування;
  + агрегація;
  + використання;

Успадкування класів –  механізм утворення нових [класів](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) на основі використання вже існуючих. При цьому властивості та функціональність батьківського класу переходять до класу нащадка (дочірнього). Таким чином буде зекономлено маса часу на написання та налагодження коду нової програми. Об'єкти похідного класу вільно можуть використовувати все, що створено і налагоджено в базовому класі. При цьому, є можливість в похідний клас дописати необхідний код для удосконалення програми: додати нові елементи, методи тощо. Базовий клас залишиться недоторканим.

## 

## 3.3 Розроблення інтерфейсів класів.

В проекті реалізовано клас BikeRental.

Код класу BikeRental:

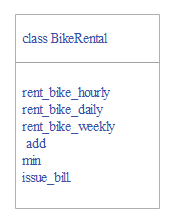


## 

В даному класі присутній конструктор, який створює об’єкт на основі заданих параметрів( ім’я, весипед, користувач). А також функції rent\_bike\_hourly, rent\_bike\_daily, rent\_bike\_weekly , add, min, issue\_bill.

## 3.4 UML – діаграма класів.

Діагра́ма кла́сів — статичне представлення структури моделі. Відображає статичні елементи, такі як: класи, типи даних, їх зміст та відношення. Діаграма класів може містити позначення для пакетів та може містити позначення для вкладених пакетів. Діаграма класів (classdiagram) служить для представлення статичної структури моделі системи в термінології класів об'єктно-орієнтованого програмування. На цій діаграмі показують класи, інтерфейси, об'єкти й кооперації, а також їхні відносини.

**

# **Розділ 4. Розроблення системи вводу-виводу даних.**

Всі програмні продукти обробляють і виводять дані. Тільки в деяких інформація надається спочатку, а в інших вона надходить ззовні. Отримувати дані можна з файлів або безпосередньо з вводиться користувачем тексту. Виходить, користувач вводить інформацію, програма її обробляє і виводить на екран. У такому випадку говорять, що код не замкнутий на собі, а може «спілкуватися» з зовнішнім середовищем. Так, для введення інформації в останніх версіях Python використовується команда input(). Для друку будь-яких даних на екрані використовується стандартна вбудована функція print(). У версії Python 3 після слова print йдуть круглі дужки, в яких вказується, що виводиться інформація.

Система Введення-виводу, тобто комплекс ЗАСОБІВ обміну інформацією Із зовнішнімі прилаштувати, є найважлівішою частина архітектури процесора и машини в цілому. До системи Введення-виводу можна Віднести І спосіб Підключення до сістемної шини різного устаткування, и процедури взаємодії процесора з ЦІМ устаткуванням, и командіровку процесора, прізначені для обміну данімі Із зовнішнімі прилаштувати.

Безперервна Вдосконалення мікропроцесорів и Прагнення максимально підвіщіті продуктивність всієї обчіслювальної системи призвело до істотного ускладнення внутрішньої организации комп'ютерів: підвіщенню розрядності шин, появі внутрішніх швідкодіючіх магістралей обміну данімі, Використання кеш-буферів для прискореного обміну з пам'яттю и дисками и інш. Если, навпаки, відвернутіся від важлівіх з подивимось продуктівності, но неістотніх для програміста деталей, то логічну схему сучасного комп'ютера можна представіті традіційнім чином, у виде сістемної шини (магістралі), до якої підключаються сам мікропроцесор і всі пристрої комп'ютера.

# **Розділ 5. Тестування програмного забезпечення.**

Тестування програми “Оренда велосипедів ” на основі функціонального тестування.

**Резюме (Summary)1:** Перевірка авторизації.

**Середовище виконання**: Windows 10.

**Опис (Description) –** Перевіряємо вхід з сторони адміністратора .

Етапи для відтворення проблеми:

1. Відкриваю програму.

2. Авторизуюся під адміністратором.

**Очікуваний результат виконання(Expected result)**: На екран повинна виводитись список з 5ти дій.

**Поточний результат виконання(Observed result)**:На екран виводиться список з 5ти дій.

**Складність(Severity):** рівень легкий

Тест пройшов успішно.

**Резюме (Summary)2:** Перевірка авторизації.

**Середовище виконання**: Windows 10.

**Опис (Description)** Перевіряємо вхід з сторони користувача .

Етапи для відтворення проблеми:

1. Відкриваю програму.

2. Авторизуюся під користувачем.

**Очікуваний результат виконання(Expected result)**: На екран повинна виводитись список з 3 дій.

**Поточний результат виконання(Observed result)**: На екран виводиться список з 3 дій.

**Складність(Severity):** рівень легкий

Тест пройшов успішно.

# **Розділ 6. Розроблення документації для супроводження програмного продукту.**

Запустивши програму відкривається вікно з авторизацією де потрібно обрати як ви авторизовуєтесь.

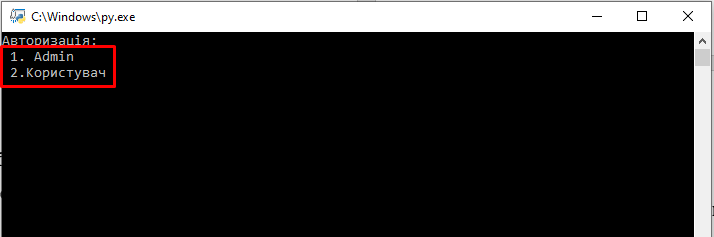
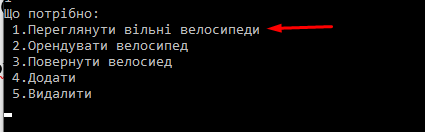


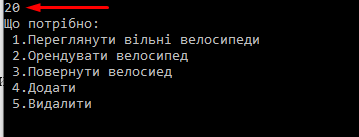
Рисунок 1(Обираємо 1-адмін)

Обравши авторизацію як адміністратор вам виведеться список з 5ти дій.

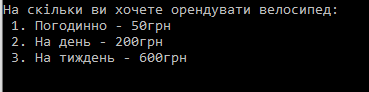
Обираємо 1.



Обравши 1 нам виводить кількість вільних велосипедів та наш список з 5ти дій.



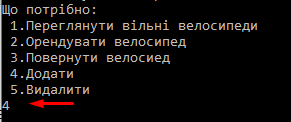
Обравши 2 нам потрібно буде обрати певний тип оренди.



Оберемо оренду на день. Потрібно виконати дві дії вписати своє ім’я та кількість орендованих велосипедів. Після чого нам виведеться повідомлення про успішну оренду.



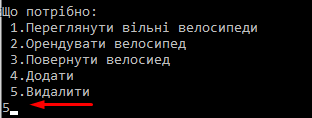
І також ми можем надалі користуватись меню. Оберемо пункт 4 додати.



Додаємо 6 нових велосипедів після чого нам виводить весь список вільних велосипедів. Оскільки ми орендувал 2 велосипеди та додали 6 нових вільних велосипедів 24.

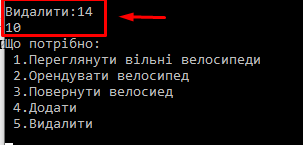


Обираємо пункт 5.



Вводимо кількість велосипедів які ми хочемо виключити з оренди 14.

Після чого нам виводиться загальна кількість велосипедів та наше меню.



Обираємо пункт 3.Нам виводить список імен які орендували велосипеди.

Нам потрібно вписати своє ім’я для повернення.

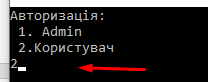


Після чого нам виводиться наш чек скільки ми обрали велосипедів та на яку суму.

Натиснувши Enter ми перейдемо до нашого меню.

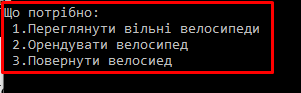


Авторизація як користувач.



У меню користувач всього лиш 3 пункти на відміну від адміністратора.

Користувач може лише орендовувати велосипеди.



# **Висновки**

Персональні комп’ютери за призначенням, фактично не мають обмежувань, а їхні можливості визначені програмами, які опрацьовують будь-яку інформацію. Щоб одержати за допомогою комп’ютера корисний результат, треба мати відповідну програму.

Сьогодні є дуже багато готових систем та пакетів. Однак будь-який пакет чи програма з моменту своєї появи морально старіє, тобто з’являються нові пропозиції щодо поліпшення їхніх можливостей, а самі задачі трансформується у нові, наперед не передбачені. Відповідно до цього модифікують пакети та програми – з’являються їхні нові версії. Крім того, сучасні пакети тяжіють до універсалізму, їхні розробки намагаються забезпечити підтримку якомога ширшого кола задач. Саме по собі це позитивне, однак коли потрібно сьогодні розв’язати одну задачу, завтра – іншу, а післязавтра – ще іншу, і всі вони неодмірні, то виникає потреба використовувати декілька різноманітних пакетів. Це може виявитись досить складним.

У цій роботі була виконана поставлена задача - створення програми за темою Моделювання оренди велосипедів. За допомогою цієї програми можна здійснити оренду велосипеда. Програма не займає багато місця, не вимоглива до встановленого програмного забезпечення. Алгоритм простий та зручний у користуванні.

# **Список використаних джерел.**

1. Дейтел, X. М. Як програмувати на С / X. М. Дейтел, П. Дж. Дейтел. - М.: Біном-Пресс, 2013. - 910 с.

2. Керниган, Б. У. Мова програмування С // Б. У. Керниган, Д. М. Рітчі. - СПб. : Вільямс, 2013. - 304 с.

3. Полубенцева, М. C / C ++. Процедурне програмування // М. Полубенцева. - СПб. : БХВ-Петербург, 2014. - 432 с.

4. Пратта, С. Мова програмування С. Лекції і вправи // С. Пратта. - СПб. : Вільямс, 2013. - 960 с.

5. Пупков, А. Н. Інформатика та програмування: навч, посіб. / А. Н. Пупков, В. В. Самарін, Р. Ю. Царьов. - Красноярськ: ІПК СФУ, 2012. - 156 с.

6. Царьов, Р. Ю. Програмування на мові Сі. - [Електронний ресурс] / Р. Ю.

7.Редько М.М. Інформатика та комп’ютерна техніка. Навчально-методичний посібник. Вінниця : Нова книга, 2014.-568с.

8.Тарасюк Г.М. Планування комерційної діяльності : Навч.посіб.-К.:Каравела, 2014.-400с.

9.Ярмуш О.В., Редько М.М. Інформатика і комп’ютерна техніка: Навчальний посібник. - К.:Вища освіта, 2013. – 359 с.

**Додаток**

**Код програми**

class BikeRental:

def \_\_init\_\_(self, name, no\_of\_bikes) -> None:

self.name = name

self.no\_of\_bikes = no\_of\_bikes

self.users = {}

def rent\_bike\_hourly(self, name:str, bikes:int):

"""

Видає людині велосипеди щогодини.

В якості вхідних даних приймає ім’я людини та кількість велосипедів для прокату.

"""

if bikes > self.no\_of\_bikes:

print(f"Дефіцит велосипедів! Ви можете мати макс {self.no\_of\_bikes} велосипедів.")

else:

self.users.update({name:{"hourly":bikes}})

print("Велосипед успішно орендований!")

self.no\_of\_bikes -= bikes

def rent\_bike\_daily(self, name:str, bikes:int):

"""

Щодня дає людині велосипеди.

В якості вхідних даних приймає ім’я людини та кількість велосипедів, які можна взяти в оренду

"""

if bikes > self.no\_of\_bikes:

print(f"Дефіцит велосипедів! Ви можете мати макс {self.no\_of\_bikes} велосипедів.")

else:

self.users.update({name:{"daily":bikes}})

print("Велосипед успішно орендований!")

self.no\_of\_bikes -= bikes

def rent\_bike\_weekly(self, name:str, bikes:int):

"""

Виділяє людині велосипеди щотижня.

В якості вхідних даних приймає ім’я людини та кількість велосипедів для прокату.

"""

if bikes > self.no\_of\_bikes:

print(f"Дефіцит велосипедів! Ви можете мати мак {self.no\_of\_bikes} велосипедів.")

else:

self.users.update({name:{"weekly":bikes}})

print("Велосипед успішно орендований!")

self.no\_of\_bikes -= bikes

def add(self, num:int):

self.no\_of\_bikes = self.no\_of\_bikes + num

def min(self, num:int):

if num > self.no\_of\_bikes:

print("Error")

else:

self.no\_of\_bikes = self.no\_of\_bikes - num

def issue\_bill(self, name:str):

"""

Видає клієнту рахунок на основі імені клієнта

"""

if name in self.users.keys():

for i in self.users[name].keys():

if i == 'hourly':

print(f"{name.capitalize()}, ви орендували {self.users.get(name).get('hourly')} велосипеди на погодинній основі.")

bill = 50 \* self.users[name]['hourly']

print(f"Отже, загальна сума до сплати: {bill}грн")

self.no\_of\_bikes += self.users[name]['hourly']

check = input("Натисніть Enter для продовження. ")

print("Повернення успішне!")

self.users.pop(name)

elif i == 'daily':

print(f"{name.capitalize()}, ви орендували {self.users.get(name).get('daily')} велосипед на день.")

bill = 200 \* self.users[name]['daily']

print(f"Отже, загальна сума до сплати: {bill}грн")

self.no\_of\_bikes += self.users[name]['daily']

check = input("Натисніть Enter для продовження. ")

print("Повернення успішне!")

self.users.pop(name)

elif i == 'weekly':

print(f"{name.capitalize()}, ви орендували {self.users.get(name).get('weekly')} велосипед на тиждень.")

bill = 600 \* self.users[name]['weekly']

print(f"Отже, загальна сума до сплати: {bill}грн")

self.no\_of\_bikes += self.users[name]['weekly']

check = input("Натисніть Enter для продовження. ")

print("Повернення успішне!")

self.users.pop(name)

else:

print("ERROR")

else:

print("Введіть ваше ім'я!")

if \_\_name\_\_ == "\_\_main\_\_":

Nagpur\_Bikes = BikeRental("Nagpur Bike Rental Service", 20)

user = int(input("Авторизація: \n 1. Admin \n 2.Користувач\n"))

if user == 1:

while 1:

task = int(input("Що потрібно:\n 1.Переглянути вільні велосипеди\n 2.Орендувати велосипед\n 3.Повернути велосиед\n 4.Додати\n 5.Видалити\n"))

if task == 1:

print(Nagpur\_Bikes.no\_of\_bikes)

elif task == 2:

type = int(input("На скільки ви хочете орендувати велосипед:\n 1. Погодинно - 50грн \n 2. На день - 200грн \n 3. На тиждень - 600грн\n"))

name = input("Введіть ваше ім'я: ")

if name in Nagpur\_Bikes.users.keys():

print("Це ім'я недоступне! Спробуйте з невеликою адаптацією.")

continue

bikes = int(input("Скільки велосипедів ви хочете орендувати? "))

if type == 1:

Nagpur\_Bikes.rent\_bike\_hourly(name, bikes)

elif type == 2:

Nagpur\_Bikes.rent\_bike\_daily(name, bikes)

elif type == 3:

Nagpur\_Bikes.rent\_bike\_weekly(name, bikes)

else:

print("Invalid input!")

elif task == 3:

if len(Nagpur\_Bikes.users) != 0:

for i in Nagpur\_Bikes.users.keys():

print(f"Ім'я користувача: {i}")

name = input("Хто ви з списку? ")

Nagpur\_Bikes.issue\_bill(name)

else:

print("No one has rented a bike yet!")

elif task == 4:

x = int(input("Додати:"))

Nagpur\_Bikes.add(x)

print(Nagpur\_Bikes.no\_of\_bikes)

elif task == 5:

x = int(input("Видалити:"))

Nagpur\_Bikes.min(x)

print(Nagpur\_Bikes.no\_of\_bikes)

else:

print("Invalid input!")

print("\n----------------------------------------------------\n")

elif user == 2:

while 1:

task = int(

input("Що потрібно:\n 1.Переглянути вільні велосипеди\n 2.Орендувати велосипед\n 3.Повернути велосиед\n "))

if task == 1:

print(Nagpur\_Bikes.no\_of\_bikes)

elif task == 2:

type = int(input(

"На скільки ви хочете орендувати велосипед:\n 1. Погодинно - 50грн \n 2. На день - 200грн \n 3. На тиждень - 600грн\n"))

name = input("Введіть своє ім'я: ")

if name in Nagpur\_Bikes.users.keys():

print("Це ім'я недоступне! Спробуйте з невеликою адаптацією.")

continue

bikes = int(input("Скільки велосипедів ви хочете орендувти? "))

if type == 1:

Nagpur\_Bikes.rent\_bike\_hourly(name, bikes)

elif type == 2:

Nagpur\_Bikes.rent\_bike\_daily(name, bikes)

elif type == 3:

Nagpur\_Bikes.rent\_bike\_weekly(name, bikes)

else:

print("Invalid input!")

elif task == 3:

if len(Nagpur\_Bikes.users) != 0:

for i in Nagpur\_Bikes.users.keys():

print(f"Ім'я користувача: {i}")

name = input("Хто ви з списку? ")

Nagpur\_Bikes.issue\_bill(name)

else:

print("Ще ніхто не орендував велосипед!")

else:

print("Invalid input!")

print("\n----------------------------------------------------\n")