Міністерство освіти і науки України

Державний університет “Житомирська політехніка”

Кафедра інженерії програмного забезпечення

Група: ВТ-21-1[1]

Програмування мовою Python

Лабораторна робота № 9

«КЛАСИ. Ч. 3»

Виконав: Бабушко А. С.

Прийняв: Морозов Д. С.

***Мета роботи:*** ознайомитися з ООП, множинним наслідуванням, міксинами в мові Python

***Хід роботи:***

***Завдання на лабораторну роботу:***

1. Створіть клас Alphabet. Його метод \_\_init \_\_ (), буде мати визначені два параметри: lang - мова і letters - список букв. Значення змінних lang і letters будуть визначенні за замовчуванням і міститимуться у вигляді статичних атрибутів для української мови. Клас матиме метод метод print\_alphabet(), який виведе в консоль літери україхнського алфавіту. Метод letters\_num(), повертатиме кількість букв в алфавіті. Метод is\_ua\_lang() прийматиме довільний текст і визначатиме чи відноситься він до української мови (незалежно від регістру). Створіть клас EngAlphabet шляхом успадкування від класу Alphabet. Для його методу \_\_init \_\_(), всередині якого буде викликатися батьківський метод \_\_init \_\_(), в якості параметрів будуть передаватися позначення мови (наприклад, 'En') і рядок, що складається з усіх букв алфавіту. Додайте приватний статичний атрибут \_\_en\_letters\_num, який буде зберігати кількість букв в алфавіті. Створіть метод is\_en\_letter(), який буде приймати строку в якості параметра і визначати, чи відноситься ця строка до англійського алфавіту. Перевизначити метод letters\_num() - нехай в поточному класі класі він буде повертати значення властивості \_\_en\_letters\_num. 6. Створіть статичний метод example(), який буде повертати приклад тексту англійською мовою.

Тести до модуля:

* Створіть об'єкт класу EngAlphabet
* Надрукуйте літери алфавіту для цього об'єкту
* Виведіть кількість букв в алфавіті
* Перевірте, чи відноситься буква J до англійського алфавіту.
* Перевірте, чи відноситься буква Щ до українського алфавіту
* Виведіть приклад тексту англійською мовою

***Лістинг програми:***

***Результат програми:***

1. Створіть клас Human. Визначте для нього два статичних атрибути: default\_name і default\_age. Його метод \_\_init \_\_(), який крім self приймає ще два публічних параметри(name і age) і два приватних (money і house). Параметр money визначатиме кількість грошей, а house – посилання на об’єкт класу House. Метод info(), має виводити поля name, age, house і money. Реалізуйте довідковий статичний метод default\_info(), який буде виводити статичні поля default\_name і default\_age. Реалізуйте приватний метод make\_deal(), який буде відповідати за технічну реалізацію покупки будинку: зменшувати кількість грошей на рахунку і привласнювати посилання на тільки що куплений будинок. В якості аргументів даний метод приймає об'єкт будинку та його ціну. Реалізуйте метод earn\_money(), що збільшує значення поля money. Реалізуйте метод buy\_house(), який буде перевіряти, що у людини достатньо грошей для покупки, і здійснювати операцію. Якщо грошей занадто мало - потрібно вивести попередження в консоль. Параметри методу: посилання на будинок і розмір знижки (за замовчуванням 10%). Створіть клас House. Його метод \_\_init \_\_() містить два динамічних параметри: \_area і \_price, що мають значення за замовчуваннями. Створіть метод final\_price(), який приймає як параметр розмір знижки і повертає ціну з урахуванням даної знижки. Створіть клас SmallHouse, успадкувавши його функціонал від класу House. Всередині класу SmallHouse перевизначите метод \_\_init \_\_() так, щоб він створював об'єкт з площею 40м2

Тести до модуля:

* Викличте довідковий метод default\_info() для класу Human
* Створіть об'єкт класу Human
* Виведіть довідкову інформацію про створений об'єкт (викличте метод info ()).
* Створіть об'єкт класу SmallHouse
* Спробуйте купити створений будинок, переконайтеся в отриманні попередження.
* Виправте фінансове становище об'єкта - викличте метод earn\_money()
* Знову спробуйте купити будинок
* Подивіться, як змінився стан об'єкта класу Human.

***Лістинг програми:***

***Результат програми:***

1. Створіть клас Apple. Його статичний атрибут states, яке буде містити всі стадії дозрівання яблука («Відсутнє», «Цвітіння», «Зелене», «Червоне»). Метод \_\_init \_\_(), всередині якого будуть визначені два динамічних protected атрибути: \_index (номер яблука) і \_state (приймає перше значення зі словника states). Створіть метод grow(), який буде переводити яблуко на наступну стадію дозрівання Створіть метод is\_ripe(), який буде перевіряти, що яблуко дозріло (досягло останньої стадії дозрівання). Створіть клас AppleTree. Визначте метод \_\_init \_\_(), який буде приймати як параметр кількість яблук і на його основі буде створювати список об'єктів класу Apple. Даний список буде зберігатися всередині динамічного атрибуту apples. Створіть метод grow\_all(), який буде переводити всі об'єкти зі списку яблук на наступний етап дозрівання. Створіть метод all\_are\_ripe(), який буде повертати True, якщо все яблука зі списку стали стиглими. Створіть метод give\_away\_all(), який буде чистити список яблук після збору врожаю Створіть клас Gardener. Його метод \_\_init \_\_(), міститиме два динамічних атриути: name (ім’я садівника, публічний атрибут) і \_tree (приймає об’єкт класу AppleTree). Створіть метод work(), який змушує садівника працювати, що дозволяє яблукам ставати більш стиглими. Створіть метод harvest(), який перевіряє, чи всі плоди дозріли. Якщо всі - садівник збирає урожай. Якщо і - метод друкує попередження. Створіть статичний метод apple\_base(), який виведе в консоль довідку з кількості яблук і ступені їх стиглості.

Тести до модуля:

* Створіть декілька об’єктів класу Apple.
* Викличте довідку по всім наявним яблукам
* Створіть об'єкти класів AppleTree і Gardener
* Використовуючи об'єкт класу Gardener, попрацювати над яблучним деревом.
* Спробуйте зібрати урожай
* Якщо яблука ще не дозріли, продовжуйте доглядати за деревом
* Зберіть урожай.

***Лістинг програми:***

***Результат програми:***

1. Створіть клас KmrCsv, який має два атрибути класу за замовчуванням: ref (посилання на CSV файл з оцінками) і num (номер КМР), та методи для встановлення і, відповідно, визначення посилання на файл з оцінками, встановлення номеру КМР, читання файлу з оцінками та виведення інформації про файл (номер КМР і кількість студентів, що її виконали).

Створіть клас Statistic, що містить наступні методи:

- avg\_stat() визначає відсотки правильних відповідей на кожне питання серед усіх студентів і повертає результат у вигляді кортежу чисел;

- метод marks\_stat() визначає яку оцінку набрала відповідна кількість студентів і повертає результат у викляді словника формату {оцінка: кількість студентів};

- метод marks\_per\_time() визначає який середній бал за хвилину набирав студент за під час виконання КМР і повертає результат у вигляді словника формату {id студента (це перша колонка csv файлу): середній бал за хвилину} ;

- метод best\_marks\_per\_time(), який приймає два аргументи bottom\_margin і top\_margin (нижня і верхня межа вибірки підсумкових балів за КМР), та формує для цієї вибірки п’ять найкращих результатів середніх балів за хвилину у вигляді кортежу формату (id студента, підсумкова оцінка, середній бал за хвилину).

Створіть клас Plots, що містить наступні методи:

* set\_cat() – встановлює каталог в який зберігатимуться отримані графіки;
* avg\_plot() – приймає кортеж з відсотками правильних відповідей на кожне окреме питання, формує гістограму на його основі і зберігає отриманий графік;
* marks\_plot() – приймає словник з оцінками і кількістю студентів, що їх набрали, формує на його основі гістрограму і зберігає її
* best\_marks\_plot() – формує для п’яти найкращих результатів середніх балів за хвилину гістрограму і зберігає її.

Створіть клас KmrWork, що успадковує класи CsvKmr, Statistic і Plots. В якості аргументів екземпляр класу приймає посилання на csv файл та номер КМР.

Клас KmrWork містить наступні статичні атрибути

* kmrs - в ньому зберігається словник формату {номер КМР: адреса відповідного csv файла}
* cat – каталог для збереження результатів роботи

Крім успадкованих, клас KmrWork містить наступні методи:

* compare\_csv() – виводить на екран і зберігає в txt файл результат порівняння статистики двох КМР (кількість виконаних КМР, середній бал за КМР, середній час виконання КМР);
* compare\_avg\_plots() – виводить на екран і зберігає дві гістограми з відсотками правильних відповідей на кожні окремі питання.

Тести до модуля:

* Створіть об’єкти kmr1 і kmr2 класу KmrWork.
* Використайте для об’єкту kmr2 методи avg\_plot() і marks\_plot()
* Для класу KmrWork використайте методи compare\_csv() і compare\_avg\_plots().

***Лістинг програми:***

***Результат програми:***

***Увесь лістинг програми:***

***Висновок:*** під час виконання лабораторної роботи було отримано навички