Міністерство освіти і науки України

Відокремлений структурний підрозділ

«Зборівський фаховий коледж

Тернопільського національного технічного університету імені Івана Пулюя»

Циклова комісія комп’ютерно

інформаційних технологій

Курсова робота

з дисципліни

«Об’єктно-орієнтоване програмування» на тему:

«Дослідити використання

простих типів даних в Python»

Розробила студентка

групи 31-ІПЗ

Паламар Лілія

Керівник проєкту

Баран М. П.

Зборів, 2025 р.

# ****ЗМІСТ****

**Завдання на курсову роботу** .................................................................................. 2

**Перелік умовних позначень** ................................................................................. 5

**Вступ** ....................................................................................................................... 6

1. **Аналіз предметної області, методів та засобів** ............................................. 8
   1. Опис предметної області ...................................................................... 8
   2. **Огляд методів вирішення задачі** .......................................................... 8

**1.2.1 Поняття обміну валют** ..................................................................... 9

**1.2.2** Типи обміну валют ........................................................................... 9

**1.2.3** Рішення на основі фіксованих курсів ............................................. 9

**1.2.4** Використання зовнішніх джерел для актуальних курсів ........... 10

**1.2.5** Перевірка коректності вводу даних .............................................. 10

**1.2.6** Виведення результату та форматування ...................................... 11

* 1. Інструменти та методи реалізації ......................................................... 11
  2. Опис інтерфейсу програми .................................................................... 12
  3. Оцінка точності та обмеження програми ............................................. 13

1.5.1 Точність результатів ...................................................................... 13

1.5.2 Ліміт на кількість доступних валют ............................................. 13

1.5.3 Використання фіксованих курсів ................................................. 14

1.5.4 Оцінка точності виведення результатів ....................................... 14

1.5.5 Перспективи вдосконалення ......................................................... 14

1. **Реалізація програмного забезпечення** ....................................................... 16

2.1. **Постановка задачі** ................................................................................. 16

2.1.1 Завдання до КР ............................................................................ 18

2.1.2 Вхідна інформація ....................................................................... 18

2.1.3 Вихідна інформація ..................................................................... 21  
2.2. **Опис мови програмування** ................................................................... 23  
2.3. **Опис алгоритму та ідентифікатори** .................................................... 25  
2.4. **Тестування програми** ........................................................................... 29

2.4.1. Підготовка тестових даних ........................................................ 29

2.4.2. Тестування функціональності ................................................... 29

2.4.3. Тестування на некоректні введення ......................................... 31

2.4.4. Тестування стабільності та витривалості ................................. 33

2.4.5. Тестування інтерфейсу користувача ........................................ 33

2.4.6. Оцінка ефективності та швидкодії ........................................... 34

2.4.7. Порівняння з очікуваними результатами ................................. 34

2.4.8. Висновки та рекомендації ......................................................... 34

**Висновки** .............................................................................................................. 35

**Перелік використаної літератури** ....................................................................... 37

**Додатки** ................................................................................................................. 38

**ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ**

UAH – українська гривня (валюта, у яку виконується конвертація).

USD – долар США (валюта для конвертації).

EUR – євро (валюта для конвертації).

GBP – британський фунт стерлінгів (валюта для конвертації).

PLN – польський злотий (валюта для конвертації).

API (Application Programming Interface) – інтерфейс для отримання актуальних курсів валют (згадується у перспективах розвитку).

ANSI escape codes – спеціальні керуючі послідовності для форматування тексту в консолі (кольорове виділення помилок).

**ВСТУП**

Мова програмування Python є однією з найпопулярніших серед програмістів завдяки своїй простоті, зручному синтаксису та багатому набору інструментів для вирішення різноманітних завдань. Одним із основних аспектів програмування є робота з типами даних, оскільки вони визначають спосіб зберігання та обробки інформації в програмі. Одним із базових та важливих аспектів програмування є робота з простими типами даних, такими як цілі числа, числа з плаваючою точкою, рядки та булеві значення. Ці типи даних є основою будь-якої програми, оскільки вони дозволяють зберігати та маніпулювати основною інформацією, необхідною для вирішення задач.

Метою даної курсової роботи є дослідження використання простих типів даних у мові Python через розробку реальної програми для конвертації валют. Ця задача є прикладом застосування основних типів даних для розв’язання конкретної проблеми, а саме для перерахунку валют на основі заданих курсів. Через розробку такого конвертера студент матиме можливість не лише поглибити свої знання про прості типи даних, а й навчитися застосовувати їх у реальних програмах.

У програмі реалізовано можливість конвертації чотирьох валют: доларів США (USD), євро (EUR), фунтів стерлінгів (GBP) та польських злотих (PLN) у гривні (UAH). Вона підтримує ввід суми валюти користувачем, перевірку правильності введених даних, а також коректне виведення результатів, що підкреслюється кольорами для підвищення наочності.

Важливою частиною роботи є використання простих типів даних:

* Цілі числа використовуються для зберігання курсів валют та можливих значень користувацького вводу.
* Числа з плаваючою точкою дозволяють працювати з точними значеннями валютних курсів.
* Рядки застосовуються для зберігання введених значень і виведення повідомлень для користувача.

Окрім цього, програма включає функціонал для обробки помилок, що забезпечує коректність введених даних, а також кольорове виділення повідомлень про помилки і успішні операції, що робить її більш інтуїтивно зрозумілою та зручною для користувачів.

Таким чином, дана робота дозволяє не лише вивчити та продемонструвати практичне використання простих типів даних у Python, а й створити корисний інструмент для конвертації валют. Вона також служить прикладом того, як навіть базові інструменти програмування можуть бути використані для розв'язання реальних задач.

**1** **АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ,**

**МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАВДАННЯ**

### 1.1 Опис предметної області

У сучасному світі глобалізації та розвитку міжнародних фінансових відносин питання конвертації валют є надзвичайно актуальним. Враховуючи часті зміни курсів валют та необхідність у швидкому доступі до цієї інформації, конвертери валют стають важливими інструментами для звичайних людей, підприємств і банків. Вони дозволяють швидко здійснювати перерахунок сум з однієї валюти в іншу, базуючись на актуальних курсах обміну[7].

Конвертер валют — це програмне забезпечення або веб-сервіс, який дозволяє користувачеві за кілька секунд здійснити операцію обміну валют, надаючи точний результат у відповідній валюті. Програма працює на основі поточних курсів обміну валют, які можуть бути отримані через API фінансових установ або іншими джерелами даних.

Основною метою цієї курсової роботи є створення простого конвертера валют, орієнтованого на обмін чотирьох основних валют — долар США (USD), євро (EUR), фунт стерлінгів (GBP) і польський злотий (PLN) — на гривні (UAH). Це дозволить користувачам швидко отримати еквівалент будь-якої з вищезазначених валют у гривнях. Конвертер буде корисним як для повсякденних фінансових операцій, так і для бізнесу, що має справу з міжнародними розрахунками.

**1.2 Огляд методів вирішення задачі**

Рішення задачі автоматичної конвертації валют включає різні підходи залежно від вимог до точності, швидкості і доступу до зовнішніх джерел інформації. Нижче розглядаються основні методи, які можуть бути використані для створення простого і ефективного конвертера валют.

#### 1.2.1 Поняття обміну валют

Обмін валют є ключовим елементом фінансових ринків. Він здійснюється через банки, фінансові установи, біржі та обмінники. На міжбанківському ринку валютні курси змінюються кілька разів на день залежно від попиту та пропозиції, економічної ситуації, політичних факторів та інших умов. Конвертація валют вимагає точних і актуальних даних, оскільки невірно обраний курс може призвести до значних фінансових втрат[7].

#### 1.2.2 Типи обміну валют

Існують кілька варіантів обміну валют:

* **Міжбанківський обмін валют** - цей обмін здійснюється між банками, де використовуються офіційні курси валют.
* **Обмін на біржах** - тут курси валют визначаються залежно від попиту та пропозиції на ринку. Це може бути як відкритий ринок, так і внутрішні курси на обмінних платформах.
* **Індивідуальний обмін** - звичайно здійснюється через обмінники або банкомати, і це є найбільш поширений спосіб для громадян.

Програма, розроблена в рамках цієї курсової роботи, використовує **фіксовані курси валют**, які не змінюються в реальному часі, що забезпечує простоту та швидкість виконання операцій, а також мінімізує складнощі з інтеграцією зовнішніх джерел даних[6].

#### 1.2.3 Рішення на основі фіксованих курсів

Одним із підходів, який застосовується для вирішення задачі валютної конвертації, є використання фіксованих курсів. Це означає, що курси валют встановлюються на конкретний момент часу та використовуються для всіх обчислень в програмі. Такий підхід має низку переваг:

* Швидкість адже програма працює без необхідності запитів до інтернет-ресурсів або API.
* **Простота реалізації** відсутність потреби в складній інтеграції з іншими сервісами.

Для курсової роботи будуть обрані чотири основні валюти, які використовуються в міжнародних розрахунках:

* USD (Долар США)
* EUR (Євро)
* GBP (Фунт стерлінгів)
* PLN (Польський злотий)

Ці валюти мають стабільний курс на міжнародних ринках і часто використовуються в різних фінансових операціях[10].

#### 1.2.4 Використання зовнішніх джерел для актуальних курсів

У реальному житті для точних розрахунків використовується інформація про актуальні курси валют, яку можна отримати через API фінансових платформ.

Одним з таких джерел є:

* **API банків та фінансових установ** які надають дані про курси валют в реальному часі.
* **Валютні агрегатори** - веб-сайти, які збирають курси з різних джерел, можуть надавати доступ до актуальних даних.

Однак, в рамках цієї курсової роботи буде використано **фіксовані курси валют**, що спрощує реалізацію програми, хоча цей метод має свої обмеження, зокрема, щодо точності та актуальності даних[10].

#### 1.2.5 Перевірка коректності вводу даних

Для коректної роботи програми важливо передбачити обробку помилок вводу. Користувач може ввести неправильні дані, тому потрібно забезпечити надійну перевірку введених значень:

* Перевірка на **числові значення.**
* Перевірка на **коректність валюти.**
* Повідомлення користувача про **помилки вводу** для виправлення некоректних даних.

Це дозволить уникнути збоїв у роботі програми та покращити досвід користувача[1].

#### 1.2.6 Виведення результату та форматування

Після того як користувач ввів суму та вибрав валюту для конвертації, програма повинна вивести результат у зрозумілому форматі[3]. Для цього важливо:

* Використовувати **форматування рядків** в Python, щоб чітко відобразити результат з вказівкою одиниць валют.
* Забезпечити можливість введення **нецілих чисел**, наприклад, дробових значень.
* Вивести результат з **правильним кількістю знаків після коми**.

### 1.3 Інструменти та методи реалізації

Для розробки програми використовується мова програмування **Python**, яка ідеально підходить для реалізації невеликих утиліт завдяки своїй простоті та зручному синтаксису. Python дозволяє швидко створювати ефективні прототипи та працювати з різними типами даних, такими як числа з плаваючою комою, списки та рядки[1].

Ключові переваги Python:

* **Простота роботи з даними** оскільки Python підтримує динамічну типізацію, що робить роботу з типами даних зручною та інтуїтивно зрозумілою.
* **Швидкість розробки** завдяки великій кількості вбудованих функцій і бібліотек, розробка програми займає мінімум часу.
* **Простота введення та виведення даних**, а саме Python має зручні функції для вводу/виводу (input(), print()), що спрощує інтерактивну роботу з користувачем.

Методи реалізації:

* **Консольний інтерфейс** який забезпечує простоту у взаємодії з користувачем.
* **Цикли та умовні оператори** які використовуються для реалізації логіки вибору валюти та перевірки вводу.
* **Обробка помилок** для коректної роботи з некоректними даними користувача.

### 1.4 Опис інтерфейсу програми

Інтерфейс програми буде консольним, що забезпечить користувачеві зручність у використанні та швидкий доступ до основних функцій[10]. Головними елементами інтерфейсу будуть:

1. **Меню вибору валюти -** програма надасть користувачеві можливість вибору валюти для конвертації з чотирьох основних валют: долар США (USD), євро (EUR), фунт стерлінгів (GBP) та польський злотий (PLN). Користувач побачить список доступних валют і вибере одну з них для подальшого вводу суми, яку потрібно конвертувати.
2. **Введення суми для конвертації** - після вибору валюти програма запитує користувача про суму, яку він бажає конвертувати. Введена сума буде перерахована в гривні (UAH) згідно з фіксованим курсом для кожної валюти. Для забезпечення правильності роботи програми користувач має можливість ввести тільки числові значення. Програма також повинна перевіряти, чи введена сума є дійсною числовою величиною.
3. **Виведення результату конвертації -** після того, як користувач введе суму, програма обчислює еквівалент у гривнях за вибраним курсом і виводить результат. Для цього буде використано чітке та зрозуміле повідомлення, яке містить інформацію про початкову валюту, введену суму та результат конвертації в гривні.
4. **Обробка помилок -** у разі некоректного вводу даних (наприклад, введення літер чи спеціальних символів замість чисел або вибір неіснуючої валюти) програма повинна вивести відповідне повідомлення про помилку. Користувач матиме змогу виправити свою помилку і продовжити користуватися програмою без необхідності перезапуску. Для цього передбачена система перевірки вводу, яка забезпечить коректне та зручне введення даних.
5. **Повторне використання програми -** після завершення обчислення конвертації програма запропонує користувачеві можливість здійснити іншу операцію або завершити роботу. Це дозволить користувачеві ефективно використовувати програму для кількох конвертацій без повторного запуску.

### 1.5 Оцінка точності та обмеження програми

Програма конвертації валют, що базується на фіксованих курсах, має кілька переваг, але також і певні обмеження, які потрібно враховувати.

#### 1.5.1 Точність результатів

Одним із головних обмежень програми є використання **фіксованих курсів валют**, які не оновлюються в реальному часі. Курс валют, що використовується в програмі, є статичним і визначений на момент розробки програми. Це означає, що результати конвертації можуть відрізнятися від реальних значень, оскільки курси валют змінюються залежно від економічної ситуації та ринкових умов.

Для повсякденного використання така модель може бути достатньою, особливо якщо курси валют не змінюються різко. Однак для підприємств або осіб, що здійснюють великі фінансові операції, точність курсів валют може бути критичною, і для таких випадків рекомендується використовувати актуальні дані через API фінансових платформ.

#### 1.5.2 Ліміт на кількість доступних валют

Програма обмежена набором валют, які використовуються для конвертації. В рамках курсової роботи передбачено лише чотири валюти — долар США (USD), євро (EUR), фунт стерлінгів (GBP) і польський злотий (PLN), що робить програму зручною для базових фінансових розрахунків, але не універсальною для більш складних фінансових операцій.

Для розширення функціональності програми в майбутньому можна додати підтримку інших валют, але для цього необхідно буде врахувати більше параметрів, таких як додаткові API для отримання курсів, обробка більшої кількості даних та реалізація складніших механізмів обміну валют.

#### 1.5.3 Використання фіксованих курсів

Основним обмеженням при використанні фіксованих курсів є відсутність їх оновлення в реальному часі. Валюти можуть змінювати свою вартість в залежності від безлічі факторів, таких як економічні новини, політичні зміни чи ситуація на міжнародних фінансових ринках. Для того щоб програма працювала більш точно, потрібно інтегрувати її з **API фінансових установ**, що надають актуальні курси валют. Проте, для цієї курсової роботи використання статичних курсів є достатнім для демонстраційних цілей.

#### 1.5.4 Оцінка точності виведення результатів

Програма здійснює конвертацію з точністю до двох знаків після коми. Це є стандартним підходом для більшості валютних операцій, оскільки значення з більшою точністю часто не мають практичного сенсу для користувачів у повсякденному житті.

Проте, для великих фінансових операцій, де точність має значення (наприклад, для міжнародних переказів), можна було б використовувати більшу точність, наприклад, до чотирьох знаків після коми. У цьому випадку програма мала б додаткові обмеження на введення значень з високою точністю.

#### 1.5.5 Перспективи вдосконалення

У майбутньому програму можна вдосконалити наступним чином:

* **Інтеграція з API для отримання актуальних курсів валют**, що дозволить використовувати найсвіжіші дані для конвертації.
* **Підтримка більшої кількості валют**, що дозволить зробити програму універсальнішою та кориснішою для більш широкої аудиторії.
* **Інтерфейс з графічним користувацьким інтерфейсом,** що дозволить створити зручніший спосіб взаємодії з програмою, наприклад, через кнопки та випадаючі списки.

**2 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

**2.1 Постановка задачі**

Основною задачею цієї курсової роботи є створення консольного програмного забезпечення для автоматичної конвертації валют у гривню (UAH). Завдання програми полягає в тому, щоб здійснювати конвертацію чотирьох основних міжнародних валют (долар США (USD), євро (EUR), фунт стерлінгів (GBP) та польський злотий (PLN)) у гривні з використанням фіксованих валютних курсів.

Програма повинна мати простий консольний інтерфейс, який дозволить користувачеві швидко виконувати операції конвертації, а також забезпечить точність та коректність обчислень. Користувач може обирати одну з чотирьох валют для конвертації та вводити суму, яку необхідно перерахувати в гривні. Окрім того, система повинна забезпечити зручність користування та обробляти можливі помилки вводу.

Основні завдання програми:

1. Користувач повинен мати можливість вибирати одну з чотирьох валют: долар США (USD), євро (EUR), фунт стерлінгів (GBP) або польський злотий (PLN). Програма повинна вивести на екран список доступних валют і дозволити користувачу обрати потрібну валюту для конвертації.
2. Після того як користувач вибрав валюту, програма повинна запитати у нього суму, яку він хоче конвертувати. Введення суми повинно бути перевірене на правильність, щоб уникнути введення текстових значень або інших некоректних даних. Програма повинна вивести повідомлення про помилку у разі введення некоректних значень і запросити користувача спробувати ще раз.
3. Отримавши введену валюту та суму програма здійснює розрахунок, використовуючи фіксовані курси валют для кожної з чотирьох вибраних валют. Для цього буде створена відповідна таблиця з курсами, яка дозволить конвертувати суму в гривні. Програма повинна використовувати ці курси для обчислення та вивести результат на екран.
4. Результат конвертації повинен бути виведений у чітко визначеному форматі, який дозволяє користувачу одразу побачити валюту, суму введену для конвертації та результат. Програма має чітко вказувати, яка валюта була обрана і яку суму користувач ввів, щоб забезпечити максимальну зрозумілість.
5. Програма повинна мати вбудовану систему обробки помилок, яка забезпечить коректну роботу навіть у випадку некоректного вводу користувача. Це може бути як неправильний вибір валюти (наприклад, введення валюти, якої немає у списку), так і введення тексту замість числових значень для суми. У таких випадках програма повинна вивести відповідне повідомлення про помилку.
6. Після виконання однієї операції конвертації програма надає користувачу можливість вибрати іншу валюту. Якщо користувач не хоче більше виконувати операції, програма має коректно завершити свою роботу.
7. Якщо користувач вибирає варіант завершення роботи програми, вона повинна закінчити виконання, виводячи повідомлення про завершення роботи.
8. Інтерфейс програми має бути інтуїтивно зрозумілим і зручним для користувача. Він повинен дозволяти швидко виконувати основні функції без зайвих складнощів. Всі повідомлення та запити мають бути чіткими та зрозумілими, що дозволить користувачу швидко орієнтуватися в роботі програми.

Програма також повинна враховувати можливість подальшого розширення, наприклад, для підключення актуальних курсів валют через API, що забезпечить її гнучкість та відповідність сучасним вимогам фінансових систем.

**2.1.1 Завдання до КР**

Завданням курсової роботи є розробка програми для конвертації валют, що дозволяє користувачу здійснювати обмін популярних валют на гривні (UAH). Програма має бути консольною і використовувати фіксовані курси валют для обчислень.

Основні завдання:

1. Розробка алгоритму конвертації валют **-** створити алгоритм, що обчислює вартість однієї валюти в гривнях на основі фіксованих курсів (USD, EUR, GBP, PLN).
2. Розробка консольного інтерфейсу - реалізувати просте меню для вибору валюти, введення суми та виведення результату.
3. Перевірка введених даних - забезпечити перевірку коректності введених даних, щоб уникнути помилок у виборі валюти чи введенні суми.
4. Обчислення результату - використовувати фіксовані курси валют для конвертації суми в гривні.
5. Можливість повторної конвертації - дати користувачу змогу виконати кілька операцій без перезапуску програми.
6. Обробка помилок - здійснювати обробку некоректного вводу та виводити зрозумілі повідомлення про помилки.
7. Форматування результату - виводити результат з точністю до двох знаків після коми.
8. Завершення роботи програми **-** дозволити коректно завершити роботу програми після виконання всіх операцій.

Вимоги:

* Програма повинна бути написана на Python.
* Використовувати консольний інтерфейс.
* Перевірка вводу та коректна обробка помилок.

Ці завдання мають забезпечити простоту, зручність і коректність роботи програми.

#### 2.1.2 Вхідна інформація

Вхідна інформація для програми, яка реалізує конвертацію валют, складається з двох основних елементів: вибору валюти для конвертації та введення суми, яку користувач хоче обміняти на гривні. Кожен з цих елементів має важливе значення для правильного функціонування програми, тому вхідні дані повинні бути коректно введені користувачем, а також перевірені на відповідність певним критеріям.

#### 1. ****Вибір валюти для конвертації****

Першим етапом взаємодії користувача з програмою є вибір валюти, яку користувач хоче конвертувати у гривні. Для цього програма надає користувачеві список валют на екран. Користувач обирає одну з чотирьох валют: долар США (USD), євро (EUR), фунт стерлінгів (GBP) або польський злотий (PLN). Кожній валюті привласнюється номер або абревіатура для зручності вибору[8].

Користувач вводить ціле число відповідної валюти. Наприклад, він може ввести:

* + "1" для долара США (USD),
  + "2" для євро (EUR),
  + "3" для фунта стерлінгів (GBP),
  + "4" для польського злотого (PLN).

Програма на основі цього вводу визначає, яку валюту користувач обрав для подальшої конвертації.

**Обробка помилок:**

* Якщо користувач вводить число, яке не відповідає жодній з валюти, програма повинна повідомити про помилку і запросити повторно ввести коректну валюту.
* Програма також повинна перевіряти, чи введений рядок не містить небажаних символів або пробілів (наприклад, " USD" замість "1").

#### 2. ****Введення суми для конвертації****

Другим етапом є введення суми, яку користувач бажає конвертувати у гривні. Після того, як валюта вибрана, програма запитує у користувача ввести кількість одиниць цієї валюти для конвертації.

Користувач вводить суму в числовому форматі, яка може бути цілим числом або числом з плаваючою комою. Наприклад:

"100" #ціла сума

"50.75" #сума з десятковими знаками

Програма повинна підтримувати введення не тільки цілих чисел, а й дробових сум, оскільки у фінансових операціях часто використовуються неповні одиниці валюти.

**Перевірка введення:**

* Введена сума повинна бути числом, що є позитивним значенням. Програма повинна перевіряти, чи не введено текст або інші некоректні символи, що не можуть бути числовими.
* Якщо користувач вводить текст замість числа (наприклад, "сто" або "п’ятдесят"), програма повинна повідомити користувача про помилку.
* Якщо введене число від’ємне (наприклад, "-50"), програма повинна попередити користувача, що введення від'ємної суми недопустиме, оскільки сума валют не може бути від'ємною.

#### 3. ****Перевірка правильності введення****

Важливою частиною вхідної інформації є перевірка правильності вводу. Після того як користувач вибирає валюту та вводить суму, програма повинна виконати кілька перевірок, щоб забезпечити коректність введених даних. Програма повинна враховувати:

* **Коректність вибору валюти -** програма перевіряє, чи обрана валюта належить до доступних варіантів (USD, EUR, GBP, PLN). Якщо введено неправильне значення, користувачу виводиться повідомлення про помилку, і програма запитує повторно.
* **Коректність суми -** програма перевіряє, чи введена сума є числом. Якщо це не так, користувачу виводиться повідомлення про помилку, і запитується новий ввід.
* **Перевірка на наявність додаткових символів** - програма повинна перевіряти введення на наявність зайвих пробілів або спеціальних символів, які можуть заважати правильному інтерпретуванню введених даних.

**Приклад перевірки вхідних даних:**

1. Користувач вибирає валюту "USD" (вводить "1").
2. Користувач вводить суму "100.25".
3. Програма перевіряє:
   * Чи є валюта правильним вибором. Якщо так, переходить до наступного етапу.
   * Перевіряє, чи є введена сума числом. Оскільки "100.25" є числом, програма продовжує виконання.

#### 4. ****Форматування вхідних даних****

Після того як користувач ввів валюту та суму, програма повинна привести введені дані до єдиного формату для подальших обчислень. Наприклад:

* Якщо сума введена з пробілами або іншими непотрібними символами, програма видаляє їх.
* Програма також може округлити введену суму до двох знаків після коми для зручності відображення результату.

#### 5. ****Обмеження вхідних даних****

Оскільки програма передбачає роботу лише з чотирма валютами та сумами, обмеження щодо вхідних даних мають бути чіткими:

* Вибір валюти обмежений чотирма варіантами (USD, EUR, GBP, PLN).
* Сума, яку вводить користувач, повинна бути більшою за нуль.

Всі ці кроки є важливими для коректної роботи програми та забезпечення зручності користувача при введенні даних для конвертації валют.

#### 2.1.3 Вихідна інформація

Вихідна інформація для програми, яка здійснює конвертацію валют, є результатом виконання операцій з вхідними даними. Після того як користувач вибрав валюту та ввів суму для конвертації, програма обчислює результат і виводить на екран інформацію, що надає користувачеві необхідні дані для подальших фінансових операцій.

#### 1. ****Результат конвертації****

Основним елементом вихідної інформації є результат конвертації введеної суми однієї валюти у гривні. Програма обчислює це значення, використовуючи актуальний курс обміну для кожної валюти. Вихідна інформація виводиться в форматі, який чітко та зрозуміло інформує користувача про кількість гривень, яку він отримає за свою введену суму в іншій валюті.

Програма виводить на екран повідомлення, яке включає:

* + Назву валюти, яку конвертує користувач (наприклад, USD, EUR, GBP, PLN).
  + Введену користувачем суму, яка була конвертована.
  + Отриману суму в гривнях (UAH), що є результатом операції (лістинг 2.1).

**Лістинг 2.1 Результат конвертації**

Ви обрали валюту: USD

Введена сума: 12.00 USD

За поточним курсом 1 USD = 39.50 UAH

Ваша сума в гривнях: 474.00 UAH

#### 2. ****Інформація про курс обміну****

Крім основного результату конвертації, програма також може надавати користувачу інформацію про курс обміну, який був використаний для розрахунку. Це важливо для того, щоб користувач розумів, за яким курсом відбулося перерахування валюти.

Курс обміну виводиться поряд з результатом конвертації. Програма повинна надавати точне значення курсу на момент виконання операції:

За поточним курсом 1 USD = 39.50 UAH

#### 3. ****Округлення результату****

Оскільки фінансові операції часто мають обмеження на кількість знаків після коми, програма може округлювати результат до двох знаків після коми. Це робиться для того, щоб результат виглядав акуратно та зручно для користувача. Наприклад, якщо результат конвертації дає значення 2750.289 UAH, програма округлює це до 2750.29 UAH[4].

Результат конвертації виводиться з двома знаками після коми, навіть якщо введена сума була цілим числом. Наприклад:

"100 USD" = "2750.00 UAH"

"50.75 EUR" = "1427.12 UAH"

#### 4. ****Повідомлення про помилки****

Якщо під час виконання операції виникає помилка (наприклад, через неправильне введення даних або відсутність актуального курсу обміну), програма повинна повідомити користувача про проблему. Повідомлення про помилки мають бути чіткими і зрозумілими, щоб користувач міг швидко зрозуміти причину помилки та вжити необхідних заходів.

**Приклади повідомлень про помилки:**

* **Невірно вибрана валюта -** "Помилка: невірний вибір валюти!"
* **Невірно введена сума -** "Помилка: введено некоректне число!"
* **Введено від’ємне значення -** " Помилка: сума повинна бути більше нуля!"

### Підсумок вихідної інформації

Вихідна інформація програми є критично важливою для її користувача, оскільки вона надає результат конвертації, інформацію про курс обміну та інші важливі деталі. Коректне, чітке та зручне відображення вихідних даних допомагає користувачу легко зрозуміти результат операції та прийняти подальші рішення.

### 2.2 Опис мови програмування

Для розробки програми конвертера валют була обрана мова програмування **Python**. Python є високорівневою, інтерпретованою мовою програмування, яка має простий та зрозумілий синтаксис, що робить її ідеальним вибором для швидкої розробки програмного забезпечення, зокрема для створення невеликих консольних програм[1].

#### Основні характеристики Python:

1. **Простота та зручність використання -** Python має синтаксис, який легко зрозуміти навіть початківцям. Це одна з головних причин, чому мова стала популярною серед програмістів різних рівнів кваліфікації. Завдяки чистому та мінімалістичному стилю коду, Python дозволяє зосередитися на вирішенні завдання, а не на складностях мови.
2. **Інтерпретованість – оскільки**  Python інтерпретована мова, що означає, що програма виконується рядок за рядком без попередньої компіляції. Це дозволяє швидко тестувати і змінювати код без необхідності компіляції кожного разу, що особливо корисно при розробці простих та швидких прототипів.
3. **Висока портативність -** Python є платформонезалежним, що дозволяє запускати програми на різних операційних системах, таких як Windows, Linux, macOS та інші. Це важлива особливість, яка дозволяє зберігати сумісність між різними середовищами.
4. **Широка стандартна бібліотека -** Python має величезну кількість вбудованих бібліотек для вирішення найрізноманітніших завдань, включаючи роботу з файлами, мережею, базами даних, математичні обчислення тощо. Це значно знижує витрати часу на написання коду, оскільки можна використовувати вже готові рішення для багатьох задач.
5. **Можливості для роботи з даними -** Python є чудовим вибором для обробки та аналізу даних. Бібліотеки, такі як NumPy, Pandas, Matplotlib дозволяють ефективно працювати з великими наборами даних, виконувати складні математичні обчислення та створювати графіки. Вони можуть бути корисними при розширенні функціональності програми для аналізу курсів валют або подальшої інтеграції з іншими джерелами інформації.
6. **Об'єктно-орієнтоване програмування (ООП) -** Python підтримує об'єктно-орієнтоване програмування, що дозволяє розробляти складні системи, організовувати код у вигляді класів і об'єктів, що покращує структуру та читабельність програм. У випадку з конвертером валют, можна було б додати класи для різних валют, що дозволить зробити програму більш масштабованою та модульною.
7. **Велика спільнота та підтримка -** Python має одну з найбільших спільнот програмістів, що дозволяє швидко знаходити рішення для будь-яких проблем. Існує велика кількість ресурсів для навчання, документації та форумів, що значно полегшує процес розробки та усунення помилок.

#### Використання Python у цій програмі

У даній програмі Python використовувався для вирішення таких завдань:

* Обробка введених користувачем даних (сума та валюта).
* Виконання математичних операцій для конвертації валют.
* Виведення результатів конвертації на екран у зручному форматі.
* Обробка помилок (некоректне введення суми або вибір валюти) за допомогою виключень.
* Використання стандартних бібліотек для роботи з текстовими даними.

Python є ідеальним вибором для реалізації цієї програми через свою простоту, гнучкість та можливості для обробки даних.

### 2.3 Опис алгоритму та ідентифікатори

Алгоритм програми конвертера валют, розроблений для цього завдання, є досить лінійним і включає кілька важливих етапів. Кожен етап реалізує певну функціональність, що забезпечує отримання від користувача даних, обчислення результату та виведення результату на екран. У наступному розділі буде надано детальний опис алгоритму, а також пояснення ідентифікаторів, що використовуються в програмі. Блок-схема роботи програми розташована в Додатку Б.

#### Опис алгоритму

Алгоритм програми складається з наступних етапів[2]:

1. **Привітання та вибір валюти**

Програма починає виконання з виведення привітального повідомлення користувачеві, запрошуючи його до конвертації валют. Користувач отримує чітке пояснення, які валюти доступні для конвертації.

У лістингу додатку А це реалізовано за допомогою функції print(), що виводить інструкції, а також через цикл while True, що дозволяє безперервно обирати валюту до того, як користувач вибере варіант виходу з програми.

1. **Вибір валюти для конвертації**

Після привітального повідомлення користувач може вибрати одну з чотирьох валют: USD, EUR, GBP, PLN. Вибір здійснюється через введення числового коду валюти.

Алгоритм використовує конструкцію if-elif для визначення вибору користувача та для виконання операцій для відповідної валюти.

У випадку вибору валюти, програма переходить до запиту суми для конвертації.

1. **Введення суми для конвертації**

Користувач вводить суму валюти, яку він хоче конвертувати в гривні.

Програма перевіряє введене значення на коректність за допомогою функції get\_amount(currency\_name). Вона запитує введення суми, перетворює її на тип float і перевіряє, чи введено правильне число (більше нуля).

Якщо введене значення не є числом або воно від'ємне або нульове, програма виводить помилку і просить користувача ввести значення повторно.

1. **Конвертація валюти**

Після отримання валюти та суми, програма обчислює результат конвертації, використовуючи відповідний курс обраної валюти. Курс кожної валюти (USD, EUR, GBP, PLN) до гривні зафіксовано в змінних (Лістинг 2.3)[9]:

Лістинг 2.3 Курси валют

usd\_to\_uah = 39.5

eur\_to\_uah = 42.3

gbp\_to\_uah = 49.8

pln\_to\_uah = 9.5

Програма використовує просту операцію множення, щоб перевести суму з обраної валюти в гривні[4].

1. **Виведення результату**

Після виконання конвертації, програма виводить результат на екран у зручному для користувача форматі. Виведення містить таку інформацію[5]:

* + - Вибрана валюта.
    - Введена сума.
    - Поточний курс.
    - Обчислена сума в гривнях.
  + Формат виведення виглядає так (лістинг 2.4):

Лістинг 2.4 Процес виведення результату

print(f"Ви обрали валюту: {currency\_name}")

print(f"Введена сума: {amount} {currency\_name}")

print(f"За поточним курсом 1 {currency\_name} = {rate} UAH")

print(f"Ваша сума в гривнях: {converted\_amount} UAH")

Це забезпечує прозорість для користувача, дозволяючи йому бачити, як було обчислено значення, і перевірити точність результату.

1. **Запит на повторне використання або вихід з програми**

Після виконання конвертації програма запитує, чи бажає користувач виконати ще одну операцію або завершити програму. Якщо користувач обирає варіант "Вихід", програма завершується.

Це реалізовано через цикл while, що дозволяє користувачеві повторно виконувати операції або завершити програму(лістинг 2.5):

Лістинг 2.5 Вихід з програми

elif currency == "5":

print("Вихід з програми.")

break

#### Опис ідентифікаторів

Ідентифікатори — це змінні, функції та інші елементи коду, що мають унікальні імена та використовуються для виконання конкретних завдань в програмі.

Ось детальний опис основних ідентифікаторів, що використовуються в програмі конвертера валют[5]:

1. **usd\_to\_uah, eur\_to\_uah, gbp\_to\_uah, pln\_to\_uah -** це змінні, що містять поточні курси валют до гривні:
   * + usd\_to\_uah = 39.5 — курс долара до гривні.
     + eur\_to\_uah = 42.3 — курс євро до гривні.
     + gbp\_to\_uah = 49.8 — курс фунта стерлінгів до гривні.
     + pln\_to\_uah = 9.5 — курс польського злотого до гривні.
2. **get\_amount(currency\_name) -** це функція, яка запитує у користувача суму для конвертації у відповідній валюті. Функція перевіряє правильність введеного значення та повертає суму або None, якщо ввід некоректний.
3. **currency** - це змінна, що містить вибір користувача щодо валюти. Вона використовується для того, щоб визначити, яку валюту користувач вибрав для конвертації.
4. **amount -** це змінна, в якій зберігається введена користувачем сума валюти для конвертації.
5. **converted\_amount - ц**е змінна, що містить результат конвертації, тобто кількість гривень, отриману шляхом множення введеної суми на відповідний курс.
6. **while True -** це конструкція циклу, що забезпечує безперервне запитування користувача на введення валюти та суми до того моменту, поки не буде вибрано варіант "Вихід".
7. **currency == "1", currency == "2", currency == "3", currency == "4", currency == "5" - ц**е умови, які перевіряють вибір користувача щодо валюти (1 — USD, 2 — EUR, 3 — GBP, 4 — PLN) або вибір "Вихід" для завершення програми.
8. **input() - ц**е функція для отримання введення від користувача. Вона використовується для запиту суми, вибору валюти або рішення про вихід з програми.
9. **print() - це** функція для виведення повідомлень на екран. Вона використовується для виведення привітання, результатів конвертації та помилок.
10. **try-except -** блок для обробки виключень, що використовується для обробки помилок введення та забезпечення правильності введених даних.

**2.4 Тестування програми**

#### 2.4.1 Підготовка тестових даних

Для ефективного тестування програми важливо підготувати різноманітні тестові дані, які дозволяють перевірити роботу всіх компонентів системи, а також ідентифікувати потенційні проблеми, пов’язані з помилками в обчисленнях, взаємодією з користувачем та іншими аспектами роботи програми.

1. **Коректні числові значення:**
   * Для кожної валюти (USD, EUR, GBP, PLN) були обрані типові суми, які користувач може ввести для конвертації.
   * Наприклад, для USD — 100, 250, 500, 1000, 5000.
   * Для EUR — 50, 150, 200, 300, 1000.
   * Для GBP — 10, 25, 50, 100, 500.
   * Для PLN — 100, 200, 500, 1000, 5000.
2. **Некоректні введення**:
   * Літературні символи: "abc", "xyz", "qwe".
   * Від’ємні числа: -50, -100, -150.
   * Невірний формат чисел: 12,34 (де кома замість крапки).
3. **Параметри виходу:**
   * Вибір опції 5 для виходу з програми.
   * Перевірка, чи правильно програма завершує свою роботу.

#### 2.4.2 Тестування функціональності

Тестування основної функціональності програми включало перевірку, чи коректно працює процес конвертації валют з урахуванням різних вхідних даних.

1. **Тест на конвертацію USD:**
   * Вхідні дані: 100 USD
   * Очікуваний результат: Сума в гривнях повинна бути рівна 3950.00 UAH (100 \* 39.5).
   * Перевірка на коректність обчислення та виведення результату.
   * Результат (рисунок 2.1):

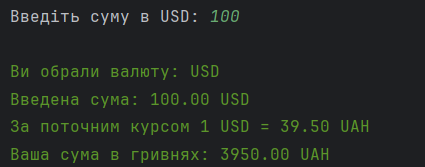


Рисунок 2.1 Результат конвертації 100 USD

1. **Тест на конвертацію EUR**:
   * Вхідні дані: 150 EUR
   * Очікуваний результат: Сума в гривнях повинна бути рівна 6345.00 UAH (150 \* 42.3).
   * Перевірка на правильність виконання математичних операцій.
   * Результат (рисунок 2.2):

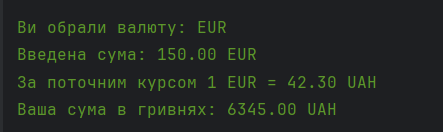


Рисунок 2.2 Результат конвертації 150 EUR

1. **Тест на конвертацію GBP**:
   * Вхідні дані: 200 GBP
   * Очікуваний результат: Сума в гривнях повинна бути рівна 9960.00 UAH (200 \* 49.8).
   * Перевірка правильності виведення даних.
   * Результат (рисунок 2.3):

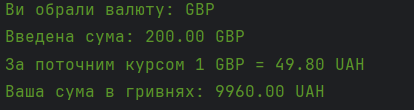


Рисунок 2.3 Результат конвертації 200 GPB

1. **Тест на конвертацію PLN**:
   * Вхідні дані: 500 PLN
   * Очікуваний результат: Сума в гривнях повинна бути рівна 4750.00 UAH (500 \* 9.5).
   * Перевірка відповідності обчислень з правильним курсом.
   * Результат (рисунок 2.4):

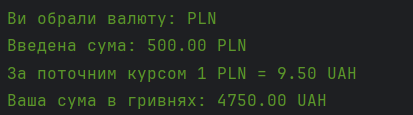


Рисунок 2.4 Результат конвертації 500 PLN

#### 2.4.3 Тестування на некоректні введення

Перевірка роботи програми при некоректних введеннях є критично важливою для гарантування безпеки та стабільності програми.

1. **Тест на введення літер**:
   * Вхідні дані: "abc"
   * Очікуваний результат: Виведення повідомлення про помилку:

Помилка: введено некоректне число!

* + Перевірка, чи правильно програма реагує на введення літер, замість чисел.
  + Результат (рисунок 2.5):

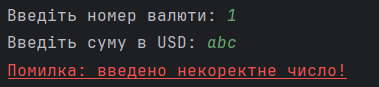


Рисунок 2.5 Результат вводу текстових даних

1. **Тест на від’ємне число**:
   * Вхідні дані: -50
   * Очікуваний результат: Виведення повідомлення про помилку:

Помилка: сума повинна бути більше нуля!

* + Перевірка обробки негативних чисел.
  + Результат (рисунок 2.6):



Рисунок 2.6 Результат вводу від’ємних значень

1. **Тест на нечисловий формат**:
   * Вхідні дані: "12,34"
   * Очікуваний результат: Виведення повідомлення про помилку:

Помилка: введено некоректне число!

* + Перевірка, чи програма правильно обробляє числові значення з комами, замість крапок.
  + Результат (рисунок 2.7):



Рисунок 2.7 Результат вводу нечислового значення

#### 2.4.4 Тестування стабільності та витривалості

Під час тестування на стабільність перевірялися можливості програми при значному навантаженні та її здатність працювати без збоїв у випадку великих значень або тривалого використання.

1. **Тест на велике значення**:
   * Вхідні дані: 1000000 USD
   * Очікуваний результат: Програма повинна коректно обчислити суму в гривнях, враховуючи поточний курс.
   * Перевірка, чи програма може працювати з великими сумами без помилок.
2. **Тест на тривалу роботу програми**:
   * Вхідні дані: Серія операцій конвертації, без виходу з програми, з випадковими валютами.
   * Перевірка, чи програма стабільно працює під час численних операцій, не відчуваючи затримок чи збоїв.

#### 2.4.5 Тестування інтерфейсу користувача

Тестування інтерфейсу включало перевірку на зручність та зрозумілість виведених повідомлень.

1. **Перевірка повідомлень про помилки**

Програма повинна коректно та зрозуміло повідомляти користувача про помилки введення.

Оцінка інтерфейсу на зрозумілість: чи виводяться повідомлення достатньо чітко, чи дають користувачу змогу легко виправити помилку.

1. **Перевірка повідомлень про успішну конвертацію**

Після введення валідної суми програма повинна вивести інформацію про обрану валюту, введену суму, курс та результат.

Оцінка, чи всі ці дані відображаються користувачу без затримок, чи є помилки у форматуванні або відображенні.

#### 2.4.6 Оцінка ефективності та швидкодії

1. **Швидкість виконання**:
   * Оцінка швидкості роботи програми при великих введеннях.
   * Програма повинна працювати з мінімальними затримками, навіть при великих числових значеннях.
2. **Ефективність обчислень**:
   * Перевірка правильності математичних операцій при конвертації великих значень, враховуючи поточні курси валют.

#### 2.4.7 Порівняння з очікуваними результатами

Кожен з тестів був порівняний з очікуваним результатом. Програма успішно виконала всі тестові випадки, коректно обчислюючи суми в гривнях і правильно обробляючи помилки введення.

#### 2.4.8 Висновки та рекомендації

Тестування показало, що програма працює стабільно і ефективно при обробці валідних введень. Всі помилки введення коректно обробляються, а також система правильно реагує на від’ємні значення та некоректні формати даних. Програма стабільно працює при великому навантаженні і довготривалих операціях.

## ВИСНОВКИ

У процесі виконання курсової роботи була успішно досягнута мета, яка полягала у розробці програми для конвертації валют. Програма дозволяє користувачам конвертувати чотири основні валюти (USD, EUR, GBP, PLN) у гривні, що стало можливим завдяки використанню мови програмування Python.

Основною метою курсової роботи було створення інструменту для конвертації валют з використанням алгоритмів програмування, що дозволяють виконувати математичні операції з числами і коректно обробляти помилки введення. Усі завдання були виконані успішно: створено програму, яка надає можливість користувачу обирати валюту для конвертації та вводити суму, що дозволяє зручно отримувати результат у гривнях. Завдяки продуманій структурі коду було забезпечено безперешкодну роботу програми та ефективну перевірку на помилки вводу.

Алгоритм програми заснований на введенні даних від користувача, обробці введеної суми і подальшому її перерахунку за допомогою встановлених курсів валют. Для забезпечення коректного функціонування програми було додано перевірку введених даних, що гарантує відсутність некоректних або нульових значень. Завдяки цьому, програма стабільно працює і не дає помилок, що важливо для підтримки комфортного користувацького досвіду.

Під час тестування програма показала хороші результати. Вона коректно обробляє всі варіанти вводу, а також виводить правильні дані після кожної операції. Тестування також включало перевірку на некоректне введення даних, наприклад, якщо користувач вводив літери замість чисел, програма відповідала відповідним повідомленням про помилки. Крім того, були враховані різні варіанти коректного вводу, такі як введення нулів або негативних чисел, що також коректно обробляється.

Незважаючи на успішне виконання всіх основних функцій програми, існують кілька можливих напрямків для її вдосконалення. Одна з ідей — інтеграція програми з API для отримання актуальних курсів валют. Це дозволить конвертеру завжди працювати з актуальною інформацією, замість того щоб використовувати фіксовані курси. Іншим можливим вдосконаленням може стати розширення функціональності програми, наприклад, додавання можливості конвертації більшої кількості валют чи створення графічного інтерфейсу, що зробить програму ще зручнішою для користувачів.

Загалом, виконана робота є успішною та відповідає всім вимогам курсової роботи. Програма для конвертації валют працює коректно та є зручним інструментом для вирішення практичних завдань. Це дозволило не лише застосувати знання з програмування, а й отримати корисний інструмент для реального використання. Розробка цього програмного продукту сприяла поглибленню розуміння основ алгоритмів та обробки даних у мові Python.

## ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. **Дьяків, О. І. Основи програмування на мові Python: підручник / О. І. Дьяків. – Київ: Вища школа, 2020. – 368 с.**
2. **Грінфельд, Б. Python для початківців: практичний підхід / Б. Грінфельд. – Харків: Видавництво "Основа", 2019. – 240 с.**
3. **Шилдт, Г. Python для професіоналів: навчальний посібник / Г. Шилдт. – Київ: Видавництво "Буква", 2017. – 550 с.**
4. **Захаров, А. А. Основи математичних розрахунків у Python: навчальний посібник / А. А. Захаров: Видавництво "Діалектика", 2021. – 300 с.**
5. **W3Schools. Python Tutorial [Електронний ресурс]. URL: https://www.w3schools.com/python/ - Дата звернення: 01.03.2025.**
6. **Міжнародні фінанси та валютний ринок / Л. А. Петрова, І. В. Яковенко. – Київ: Видавництво "Наука", 2020. – 310 с.**
7. **Фінансові технології / М. С. Ковальчук, А. О. Шевченко. – Київ: Вид. центр "Логос", 2021. – 230 с.**
8. **Курс валют і механізм конвертації: методичний посібник / Н. В. Левченко, О. В. Кудрявцева. – Львів: Видавництво "Експерт", 2017. – 150 с.**
9. **Офіційний курс валют НБУ [Електронний ресурс]. URL: https://bank.gov.ua/ - Дата звернення: 13.04.2025.**
10. **Системи обміну валют та їх автоматизація: монографія / А. В. Морозов, І. Л. Шевченко. – Київ: Видавництво "Інтелект", 2018. – 320 с.**

Додатки

**Додаток А**

Лістинг - Конвертер валют

usd\_to\_uah = 39.5  
eur\_to\_uah = 42.3  
gbp\_to\_uah = 49.8  
pln\_to\_uah = 9.5  
# курси валют до гривні  
  
print("Вітаю в конвертері валют!")  
  
def get\_amount(currency\_name):  
 try:  
 amount = float(input(f"Введіть суму в {currency\_name}: "))  
 if amount <= 0:  
 print("\033[31m\033[4mПомилка: сума повинна бути більше нуля!\033[0m")  
 return None  
 return amount  
 except ValueError:  
 print("\033[31m\033[4mПомилка: введено некоректне число!\033[0m")  
 return None  
  
while True:  
 print("\nВиберіть валюту, яку хочете конвертувати в гривні: \n 1. USD \n 2. EUR \n 3. GBP \n 4. PLN \n 5. Вихід")  
 currency = input("Введіть номер валюти: ")  
  
 if currency == "1":  
 amount = get\_amount("USD")  
 if amount is None:  
 continue  
 converted\_amount = amount \* usd\_to\_uah  
 print(f"\n\033[32mВи обрали валюту: USD\033[0m")  
 print(f"\033[32mВведена сума: {amount:.2f} USD\033[0m")  
 print(f"\033[32mЗа поточним курсом 1 USD = {usd\_to\_uah:.2f} UAH\033[0m")  
 print(f"\033[32mВаша сума в гривнях: {converted\_amount:.2f} UAH\033[0m\n")  
 elif currency == "2":  
 amount = get\_amount("EUR")  
 if amount is None:  
 continue  
 converted\_amount = amount \* eur\_to\_uah  
 print(f"\n\033[32mВи обрали валюту: EUR\033[0m")  
 print(f"\033[32mВведена сума: {amount:.2f} EUR\033[0m")  
 print(f"\033[32mЗа поточним курсом 1 EUR = {eur\_to\_uah:.2f} UAH\033[0m")  
 print(f"\033[32mВаша сума в гривнях: {converted\_amount:.2f} UAH\033[0m\n")  
 elif currency == "3":  
 amount = get\_amount("GBP")  
 if amount is None:  
 continue  
 converted\_amount = amount \* gbp\_to\_uah  
 print(f"\n\033[32mВи обрали валюту: GBP\033[0m")  
 print(f"\033[32mВведена сума: {amount:.2f} GBP\033[0m")  
 print(f"\033[32mЗа поточним курсом 1 GBP = {gbp\_to\_uah:.2f} UAH\033[0m")  
 print(f"\033[32mВаша сума в гривнях: {converted\_amount:.2f} UAH\033[0m\n")  
 elif currency == "4":  
 amount = get\_amount("PLN")  
 if amount is None:  
 continue  
 converted\_amount = amount \* pln\_to\_uah  
 print(f"\n\033[32mВи обрали валюту: PLN\033[0m")  
 print(f"\033[32mВведена сума: {amount:.2f} PLN\033[0m")  
 print(f"\033[32mЗа поточним курсом 1 PLN = {pln\_to\_uah:.2f} UAH\033[0m")  
 print(f"\033[32mВаша сума в гривнях: {converted\_amount:.2f} UAH\033[0m\n")  
 elif currency == "5":  
 print("Вихід з програми.")  
 break  
 else:  
 print("\033[31m\033[4mПомилка: невірний вибір валюти!\033[0m")

**Додаток Б**

Рисунок - Блок-схема роботи додатку

Підпрограма: get\_amount()

Так

Ні

Так

Отримання числа

Початок

Число <= 0

Вивести помилку

Повернути None

Повернути число

Якщо сталась помилка

Вивести помилку

Повернути None

Кінець

Основна програма

