

Форми контролю.**«Метод сліпого рефакторингу»**

Як додатковий метод перевірки, вчитель може надати учню код, згенерований ШІ з навмисними помилками (наприклад, без обробки датчиків або з безкінечним циклом). Завдання учня: «вилікувати» код. Це унеможливорює плагіат і стимулює критичне мислення.

Висновки

Симбіоз логіки та синтаксичної експертності. Python трансформується в інструмент верифікації гіпотез, проте успішна Prompt-інженерія неможлива без фундаментальної мовної підготовки. Фаза «Експерта» вимагає від учня здатності самостійно вносити правки та обґрунтовувати вибір структур даних, тому вивчення синтаксису не скасовується, а переходить у площину експертного аудиту та рефакторингу.

Критичне мислення через аудит. Методика «Тріадного навчання» запобігає бездумному копіюванню. Ідентифікуючи «логічні невідповідності» ШІ та опрацьовуючи граничні випадки, учень стає «технічним лідером», який повністю відповідає за логічну та синтаксичну коректність продукту.

Методична та інклюзивна цінність. Оцінювання зміщується з працездатності коду на глибину когнітивних процесів та вміння аргументувати кожен рядок. Це забезпечує справжню інклюзію, наприклад, для учнів із дислексією ШІ нівелює бар'єр написання тексту, дозволяючи реалізувати інтелектуальний потенціал через логіку декомпозиції задач та аналіз готового коду.

Використані джерела

1. Державний стандарт базової середньої освіти: Постанова Кабінету Міністрів України від 30 вересня 2020 р. № 898. Законодавство України / Верховна Рада України. URL: <https://lnk.ua/0rGC27y8j> (дата звернення: 15.02.2026).

2. Доценко С. О. Штучний інтелект у загальній середній освіті: виклики та можливості. Наукові записки кафедри педагогіки. 2023. № 52. С. 45–53. URL: <http://pedagogy-journal.at.ua/> (дата звернення: 15.02.2026).

3. Маттес Е. Пришвидшений курс Python. Проєктне навчання: ігри, візуалізація даних, веб-додатки / пер. з англ. О. О. Кисіль. Харків: Вид-во «Ранок»: Фабула, 2023. 592 с.

4. Морзе Н. В. Методика навчання інформатики : посіб. для студ. пед. навч. закл.: у 3 ч. / за ред. М. І. Жалдака. Київ: Навчальна книга, 2004. Ч. 2 : Методика навчання інформаційних технологій. 287 с.

5. Mishra P. Practical Prompt Engineering: A Comprehensive Guide to Crafting Strategic and Effective Prompts. Berkeley, CA: Apress, 2023. 218 p.

6. Mollick E. R., Mollick L. Using AI to Implement Intelligent Tutoring Systems: Bridging the Gap Between Technology and Pedagogy. SSRN Electronic Journal. 2023. URL: <https://lnk.ua/IgUTY740j> (дата звернення: 15.02.2026).



Ірина КОВАЛЬЧУК,
вчителька інформатики гімназії №1 м.Славути

Компетентнісний підхід у 8 класі: навчаємо програмувати, приймаючи рішення

Алгоритми та програми — це одна з найскладніших тем шкільного курсу, і для восьмикласника програмування часто видається набором незрозумілих команд, поки він не побачить у коді відображення власного життя. Тема розгалужень — це ідеальний місток між синтаксисом Python та реальністю. У НУШ інформатика має не лише вчити писати код, а й готувати до життя в цифровому суспільстві.

Як зазначено в модельній програмі «Інформатика. 7–9 класи» для закладів загальної середньої освіти (автори Ривкінд Й. Я., Лисенко Т. І., Чернікова Л. А., Шакоцько В. В.), пріоритетом викладання є повага до особистості учня та врахування його інтересів при доборі об'єктів для навчання. Саме тому перехід від сухих консольних розрахунків до життєвих задач із графічним інтерфейсом є ключовим для підтримки пізнавального інтересу. Згідно з програмою,

вивчення інформатики має на меті формування ключових компетентностей, серед яких особливо важливі для 8 класу:

- Математична компетентність: логічне обґрунтування дій для розв'язування задач.
- Підприємливість і фінансова грамотність: через задачі на розрахунок вартості послуг чи товарів.
- Інноваційність: здатність створювати нові інформаційні продукти для розв'язування життєвих проблем.

У цій статті основний акцент робиться на тому, як перетворити звичайний урок інформатики у 8 класі на симуляцію реального бізнесу. Відійшовши від абстрактних «х та у», навчаю учнів створювати алгоритми для кур'єрських служб, мобільних операторів, розрахунку вартості покупок та підрахунку знижок у магазинах, обчислення суми відсотків у банку за депозит чи кредит.

Розглянемо детальніше.

Задача 1 «Розумна доставка їжі»

Учням пропонуємо уявити, що вони розробляють модуль для додатка доставки їжі у своєму місті. Вартість самої доставки залежить від суми замовлення: що більше людина купує, то дешевша доставка. Також діє бонус для постійних клієнтів.

Умова: Напиши програму, яка запитує:

1. Загальну суму замовлення (у гривнях).

2. Чи є користувач постійним клієнтом (так/ні).

Правила розрахунку вартості доставки:

– Якщо сума замовлення менше 300 грн, доставка коштує 60 грн.

– Якщо сума від 300 до 700 грн (включно), доставка коштує 30 грн.

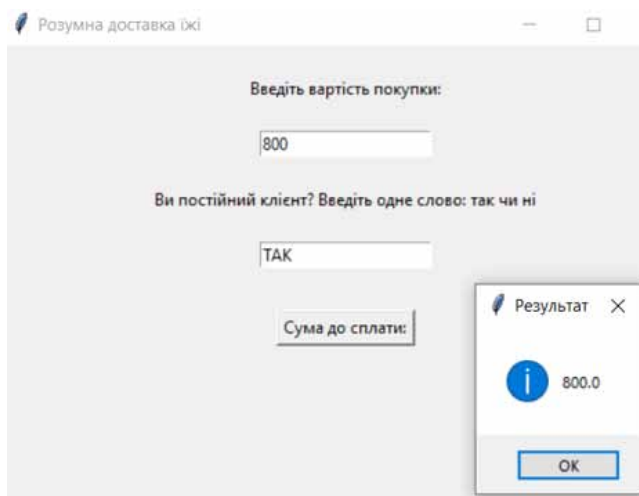
– Якщо сума понад 700 грн, доставка безкоштовна (0 грн).

– Додаткова знижка: Якщо користувач є постійним клієнтом, від фінальної суми всього чека (замовлення + доставка) віднімається ще 10%.

Розв'язок цієї задачі на Python у середовищі Thonny можна зреалізувати таким чином:

```
from tkinter import *
from tkinter.messagebox import *
def click():
    suma=float(entry1.get())
    answer=entry2.get()
    if suma >700:
        dostavka=0
    elif suma>=300 and suma <= 700:
        dostavka=30
    else:
        dostavka=60
    do_splatu = suma+dostavka
    if answer== "так":
        do_splatu=do_splatu* 0.9
    showinfo("Результат", str(do_splatu))
root=Tk()
root.geometry('500x500')
root.title('Розумна доставка їжі')
label1=Label(text='Введіть вартість покупки:')
label1.pack(pady=20)
entry1=Entry()
entry1.pack()
entry1.focus_set()
label2=Label(text='Ви постійний клієнт? Введіть одне слово: так чи ні')
label2.pack(pady=20)
entry2=Entry()
entry2.pack()
button=Button(text='Сума до сплати:', command=click)
button.pack(pady=30)
root.mainloop()
```

Під час тестування цього коду учні стикаються з проблемою: програма не спрацьовує правильно, якщо замість «так» написати «Так» або «да». У реальному програмуванні покладатися на текстове введення користувача («так», «ні», «Да», «ТАК») — це завжди ризик виникнення помилок. Як зазначено в модельній програмі Ривкінда Й.Я., учень має



вчитися створювати «дружній до користувача» інтерфейс та забезпечувати коректність опрацювання даних. Коли виникає помилка через некоректне введення тексту, вчитель отримує ідеальний момент для пояснення суті Radiobutton. Варто акцентувати увагу на тому, що перемикачі — це не просто елемент дизайну, а потужний інструмент валідації даних, який дозволяє обрати лише один варіант із чітко визначеного списку, виключаючи людський фактор.

Удосконалюючи задачу, переходимо від пасивного очікування тексту до створення інтелектуального інтерфейсу. Це навчає восьмикласників професійному підходу: замість того, щоб змушувати програму «вгадувати» наміри користувача, ми проєктуємо систему, де помилка стає неможливою заздалегідь. В результаті заміни текстового поля entry2 на перемикачі та заміни однієї умови в підпрограмі answer== "так" на choice.get() == 1 отримаємо програму, яка робить алгоритм розгалуження надійним, лаконічним і по-справжньому професійним. Нижче наведено частину коду, яку потрібно додати замість поля entry2.

```
23 choice=IntVar()
24 p1 = Radiobutton(text="так", variable = choice, value = 1)
25 p1.pack(pady = 10)
26 p2 = Radiobutton(text="ні", variable = choice, value = 2)
27 p2.pack(pady = 10)
```

Наприкінці можна запропонувати учням змінити дизайн вікна: змінити розмір вікна, колір фону або шрифт. Це додасть творчий елемент до завдання.

Сучасні діти полюбляють піцу та суші. Тому їм стане цікавою задача, яка може бути покладена за основу при створенні мобільного додатку, в якому можна обрати піцу, додати до неї на свій смак різні інгредієнти та обрахувати її вартість. Розглянемо на конкретному прикладі.

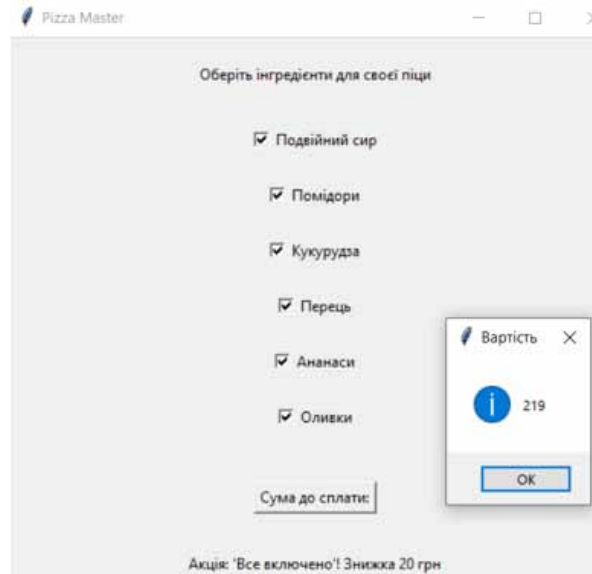
Задача 2 «Піца майстер»

Користувач обирає додатки до піци. Кожен додаток має певну вартість (у грн). Якщо обрано всі інгредієнти, нараховується додатковий бонус – 20 грн знижки. Програма має обрахувати суму до сплати.

Тут важливо, щоб учні зрозуміли, що використання таких елементів керування як прапорці (CheckBox), дає змогу вибрати кілька варіантів одночасно, а якщо потрібно, то й усі. Для цього створюється необхідна кількість змінних логічного типу, кожна з яких набуватиме значення True або False.

Зразок коду, який можуть створити учні використовуючи прапорці:

```
from tkinter import*
from tkinter.messagebox import*
def click():
    varartist = 150
    zag_varartist=varartist
    # Перевірка кожного прапорця окремо
    if cheese.get():
        zag_varartist += 20
    if tomato.get():
        zag_varartist += 15
    if corn.get():
        zag_varartist += 13
    if pepper.get():
        zag_varartist += 10
    if pineapple.get():
        zag_varartist += 17
    if olives.get():
        zag_varartist += 14
    # Складна умова: якщо обрано все, надаємо
    знижку 20 грн
    if cheese.get() and tomato.get() and corn.get() and
    pineapple.get() and olives.get() and pepper.get():
        zag_varartist -= 20
        label2=Label(text="Акція: 'Все включено!' Знижка
        20 грн")
        label2.pack()
        showinfo("Сума до сплати",zag_varartist)
    else:
        showinfo("Сума до сплати", zag_varartist)
root=Tk()
root.geometry("500x500")
root.title("Pizza Master")
label1=Label(text="Оберіть інгредієнти для своєї піци")
label1.pack(pady=20)
cheese=BooleanVar()
tomato=BooleanVar()
corn=BooleanVar()
pepper=BooleanVar()
pineapple=BooleanVar()
olives=BooleanVar()
prap1 = Checkbutton(text="Подвійний сир", variable
=cheese)
prap1.pack(pady = 10)
prap2 = Checkbutton(text="Помідори", variable = tomato)
prap2.pack(pady = 10)
prap3 = Checkbutton(text="Кукурудза", variable = corn)
prap3.pack(pady = 10)
prap4 = Checkbutton(text="Перець", variable = pepper)
prap4.pack(pady = 10)
prap5 = Checkbutton(text="Ананаси", variable =
pineapple)
prap5.pack(pady = 10)
prap6 = Checkbutton(text="Оливки", variable = olives)
prap6.pack(pady = 10)
button=Button(text="Сума до сплати", command=click)
button.pack(pady=30)
root.mainloop()
Результат виконання програми зображено на рисунку.
Наступні задачі можуть бути запропоновані учням як
домашнє завдання або для самостійної роботи.
```



Задача 3 «Квиток у кіно»

Кінотеатр проводить акцію. Стандартна ціна квитка — 150 грн. Проте квиток коштує 100 грн, якщо виконується хоча б одна з умов:

- Глядачу менше 12 років.
- Глядачу більше 60 років.
- Сьогодні середа (день глядача).

Завдання: Напиши програму, яка запитує вік та день тижня, а потім виводить ціну квитка.

Задача 4 «Квест-кімната»

Групу підлітків пустять у квест-кімнату «Таємниці Python», лише якщо виконуються такі 2 умови: їх від 2 до 5 осіб (включно), всім учасникам виповнилося 14 років.

Завдання: Запитай кількість людей та вік наймолодшого учасника. Програма має вивести «Доступ дозволено», якщо обидві умови виконуються, або «Доступ заборонено» з поясненням причини.

Задача 5 «Кур'єрська доставка (Розрахунок вартості)»

Сервіс доставки розраховує вартість так:

- Якщо сума замовлення понад 500 грн ТА відстань до будинку менше 5 км — доставка безкоштовна.
- В інших випадках доставка коштує 70 грн.
- Якщо ж зараз «Година пік» (так/ні), до будь-якої вартості доставки додається ще 30 грн.

Завдання: Запитай яка сума замовлення та відстань до будинку. «Годину пік» зроби перемикачем.

Використання бібліотеки tkinter дозволяє реалізувати об'єктний підхід, де кожен елемент інтерфейсу — це зрозумілий об'єкт із властивостями, які дитина може контролювати. Це перетворює програмування з «мови формул» на «конструктор рішень», що відповідає вимогам програми щодо створення безпечного та розвивального освітнього середовища. За допомогою конструкторів if-elif-else та операторів and і or у школярів розвиваються не лише навички кодингу, а й критичне мислення, фінансова грамотність та вміння розглядати, аналізувати і прогнозувати наслідки та перевіряти всі можливі варіанти розвитку подій. А задачі «з життя» стимулюють учнів до вивчення уже не такої складної, як на перший погляд здавалося, теми.