УДК 004.5
Дьячук Т.С.[1]
[1] старш. викл. НУ «Запорізька політехніка»

СИСТЕМА АВТОМАТИЗОВАНОЇ ПЕРЕВІРКИ ЗАВДАНЬ ДЛЯ НАВЧАЛЬНИХ КУРСІВ З ПРОГРАМУВАННЯ

Система автоматизованої перевірки завдань для курсів з програмування – це потужний інструмент оптимізації освітнього процесу, підвищення ефективності викладання та полегшення навчання студентам. Що забезпечує перевірку правильності коду, якості реалізації та відповідності вимогам завдань без потреби в постійній участі викладача.

Розглянемо основні переваги, які надає така система особливо в умовах дистанційної освіти. По перше автоматизація допоможе впоратися з первинною перевіркою великої кількості типових завдань та заощадити час як викладача так і студента. Студенти можуть отримати результати перевірки та зворотній зв’язок одразу після здачі роботи, швидше виправити свої помилки, що мотивує до самостійного навчання, а також змушує виконувати завдання згідно з чіткими вимогами. Такий підхід є доречним для великих груп студентів, де викладачам важко вручну перевірити кожне завдання. Це знімає навантаження з викладачів і дозволяє більше зосередитися на індивідуальній допомозі студентам. Також виключається фактор людської суб'єктивності в оцінюванні, оскільки автоматизована система забезпечує стандартизовану перевірку.

Однак у подібного підходу є і свої недоліки. Розробка і підтримка такої системи потребує часу, а також кваліфікації для налаштування тестів та критеріїв оцінювання. Система може перевіряти лише чітко задані показники

(наприклад, правильність вихідних даних), але не може оцінити творчі або нестандартні підходи до вирішення. Для задач, які вимагають креативних рішень або незвичних підходів, автоматизована система може бути використана тільки для початкового аналізу коректності роботи. Також якщо система має помилки або неправильно налаштовані параметри, це може призвести до некоректного оцінювання студентських робіт.

Система автоматизованої перевірки завдань була розроблена та апробована у рамках дисципліни «Основи програмування на Kotlin» (базовий репозиторій для навчання доступний за посиланням [1]). Система побудована на можливостях GitHub, таких як Pull requests (PR) та GitHub Actions. Автоматизована перевірка є частиною більш складної системи для дистанційної освіти, тому завдання, які надаються студентам також генеруються автоматично, придатні для подальшої автоматизованої перевірки та мають чітко визначену специфікацію [2].

Для прикладу розглянемо перевірку завдання з теми «Змінні та типи даних, рядки, умови та цикли, функції». Для об'єднання та формалізації варіантів формул були використані елементи принципу One-Hot Encoding [3]. Оскільки при формуванні завдання *res* (формула (1)) використовується головна *mainFnc* та вторинна функції *secondaryFnc* [2], то для перевірки також потрібно враховувати всі наявні варіанти.

$$res = mainFnc\big(secondaryFnc(X_0, X_1, \dots, X_n)\big), \tag{1}$$

де $\bar{X}$ - вектор вхідних аргументів розмірністю *n*, які для тестування система перевірки підбирає за допомогою генератора випадкових чисел.

Наведемо формулу (2) для розрахунку *secRes* - вектора результатів вторинної функції для відповідних вхідних аргументів X_i.

$$secRes_0 = A_0 \cdot X_0 + A_1 \cdot X_0^2 + A_2 \cdot X_0^3 + A_3 \cdot X_0 + A_4 \cdot X_0 + A_5 \cdot |X_0| + \\ + A_6 \cdot |X_0| + A_7 \cdot |X_0|$$

$$secRes_i = secRes_{i-1} \cdot sum + secRes_{i-1}^{mult} \cdot \big(A_0 \cdot X_i + A_1 \cdot X_i^2 + A_2 \cdot X_i^3 + \\ + A_3 \cdot min(secRes_{i-1}, X_i) + A_4 \cdot max(secRes_{i-1}, X_i) + A_5 \cdot |X_i| + \\ + A_6 \cdot min(secRes_{i-1}, |X_i|) + A_7 \cdot max(secRes_{i-1}, |X_i|)\big), \tag{2}$$

де *A* - вектор, в якому є лише один не нульовий елемент, який буде у позиції відповідно до завдання студента. Наприклад, якщо є завдання розрахувати суму квадратів, то вектор буде наступним *A = [0, 1, 0, 0, 0, 0, 0];*
sum - буде мати значення *1*, якщо в завданні потрібно розрахувати суму всіх компонентів, інакше *0;*

mult - буде мати значення *1,* якщо в завданні потрібно перемножити всі компоненти, інакше *0;*
i – приймає значення від *1* до *n.*

Головна функція *result* розраховується аналогічно (3).

$$data = [\sqrt{secRes_n}, \sqrt[3]{secRes_n}, cos(secRes_n), tan(secRes_n), sin(secRes_n), tanh(secRes_n), ln(secRes_n)]$$

$$result = \overrightarrow{data} \cdot \overrightarrow{B}\ , \tag{3}$$

де *B* - вектор, в якому є лише один не нульовий елемент, він буде у позиції відповідно до головного завдання студента. Наприклад, якщо студент отримав завдання розрахувати корінь квадратний з суми квадратів, то вектор буде наступним *B = [1,0,0,0,0,0,0]*;
$secRes_n$ - останній елемент вектору *secRes*, тобто результат ітеративного розрахунку значення вторинної функції завдання;
data - вектор всіх можливих головних функцій завдання.

На всі завдання система генерує набори тестових даних. Далі виконується розрахунок результату за кожним елементом тестових даних за програмним кодом від студента та з використанням вищенаведених формул. Якщо результати не співпадають, то завдання студента не зараховується, система видає звіт про помилку з певним набором тестових даних.

Таким чином, автоматизована перевірка завдань студентів здатна суттєво полегшити процес викладання та оцінювання знань. Завдяки цьому підходу можна автоматизувати численні аспекти навчання та забезпечити об’єктивність оцінювання.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. DiachT/KotlinLabsNUZP – Репозиторій з кодом. URL: https://github.com/DiachT/KotlinLabsNUZP (дата звернення: 10.11.2024).

2. Дьячук Т.С. Автоматизована система генерації завдань в навчальних курсах з програмування / Т.С. Дьячук, В.І Шкрябець, А.В. Тіменко, Т.В. Голуб// Вчені записки Таврійського національного університету імені В.І. Вернадського, Серія: Технічні науки, Видавничий дім «Гельветика», 2024 – Том 35 (74) № 2 2024 – С.85-90.

3. Data Science in 5 Minutes: What is One Hot Encoding? URL: https://www.educative.io/blog/one-hot-encoding (дата звернення: 10.11.2024).