Бондарєв В. М., Черепанова Ю. Ю.
Харківський національний університет радіоелектроніки

АВТОМАТИЧНИЙ ЗАДАЧНИК З ПРОГРАМУВАННЯ

Рішення задач займає центральне місце в процесі навчання майбутнього програміста. Задача - це словесна специфікація деякої програми або її частини. Щоб вирішити задачу, майбутній програміст (далі студент) повинен написати програму, яка даної специфікації цілком відповідає. Перевірку відповідності специфікації можна здійснювати різними способами. Традиційно це робить викладач - візуально, якщо програма маленька, або «тестуючи» її на комп'ютері. Слово «тестуючи» взято в лапки, тому що викладач не тестувальник, і якісної перевірки не влаштує, хоча б через брак часу. Цілком очевидно, що таким способом викладач зможе перевірити відносно невелику кількість завдань. У той же час, завдання, які не перевіряють, студенти вирішувати не стануть (не тому, що не усвідомлюють користі від їх вирішення, просто є більш термінові справи), і рішення задач відкладається з центрального місця в навчанні на далеку периферію. Як наслідок, помітний відсоток дипломованих програмістів програмувати не вміє та змушений ще раз шукати своє місце в житті.

У той же час відносно нескладно створити сервіс, який отримує вихідний код програми, компілює та виконує її, повертаючи результат виконання.

Існують аналоги в області задачників з автоматичною перевіркою рішень, (розглянемо лише декілька). Це системи автоматичної перевірки рішень з архівами задач [1, 2] або без них [3] та опціонально з можливістю участі або проведення змагань з програмування [4, 5]. Є також системи, орієнтовані на конкретні мови програмування [6, 7]. Всіх їх об'єднує спрямованість на змагання з програмування, які вимагають алгоритмічного мислення, а не знання особливостей тієї чи іншої мови програмування. Задачи там абстраговані від мови та уніфіковані за формою - введення зі стандартного потоку, рішення, виведення результатів в стандартний потік. Це відрізняє їх від задач представленої системи, які можуть варіювати від виправлення помилок в заданому фрагменті коду до визначення класу з заданим інтерфейсом або одного тільки методу. Безсумнівно, це ближче до того, що потрібно для різноманітних вправ майбутнього програміста.

Розглянемо, яким чином подібний сервіс може допомогти в перевірці рішення задач студентами. Вихідний код, що відправляється сервісу, повинен бути завершеною програмою, що складається з авторського рішення деякої задачі та серії модульних тестів цього рішення (назвемо цю комбінацію збіркою). Збірку викладач готує заздалегідь і зберігає разом з умовою задачі. Коли студент вирішує завдання, його рішення займає в збірці місце авторського і змінена таким чином збірка відправляється згаданому вище сервісу. Результат компіляції та виконання збірки розглядається як результат перевірки студентського рішення. Це може бути повідомлення про помилки, знайдені при компіляції (задача не вирішена), повідомлення про провал деяких тестів (задача не вирішена), повідомлення про успішне проходження все тестів (завдання виконане). Умова задачі, плюс збірка, плюс ідентифікатор задачі складають власне задачу - одиницю зберігання в базі даних, яку ми назвемо автоматичним задачником.

Як показала практика, наявність автоматичного задачника разом зі службою компіляції та виконання збірок радикально змінює процес початкового навчання програмістів і помітно покращує його результат.

Тепер від викладача не потрібні надмірні зусилля по перевірці студентських контрольних робіт, всі завдання вирішуються студентами на комп'ютері та негайно перевіряються.

Звичайно, необхідно подбати про наповнення задачника, але якщо завдання потрапить в задачник один раз, вирішене воно буде сотні і навіть тисячі разів. До того ж викладачі можуть наповнювати задачник кооперативно.

Переважну більшість задач студенти вирішують самостійно, викладач повинен лише опублікувати їх у мережі. Він може робити це поетапно, по мірі подачі матеріалу на лекціях, або групуючи задачі іншим способом, наприклад, в рамках лабораторних робіт. Не варто ставити студентам всі завдання відразу, краще це робити порціями, обов'язково включаючи в порцію і легкі, і середні задачі. Складні задачі теж потрібні, але їх варто пропонувати тим, хто справляється з легкими і середніми.

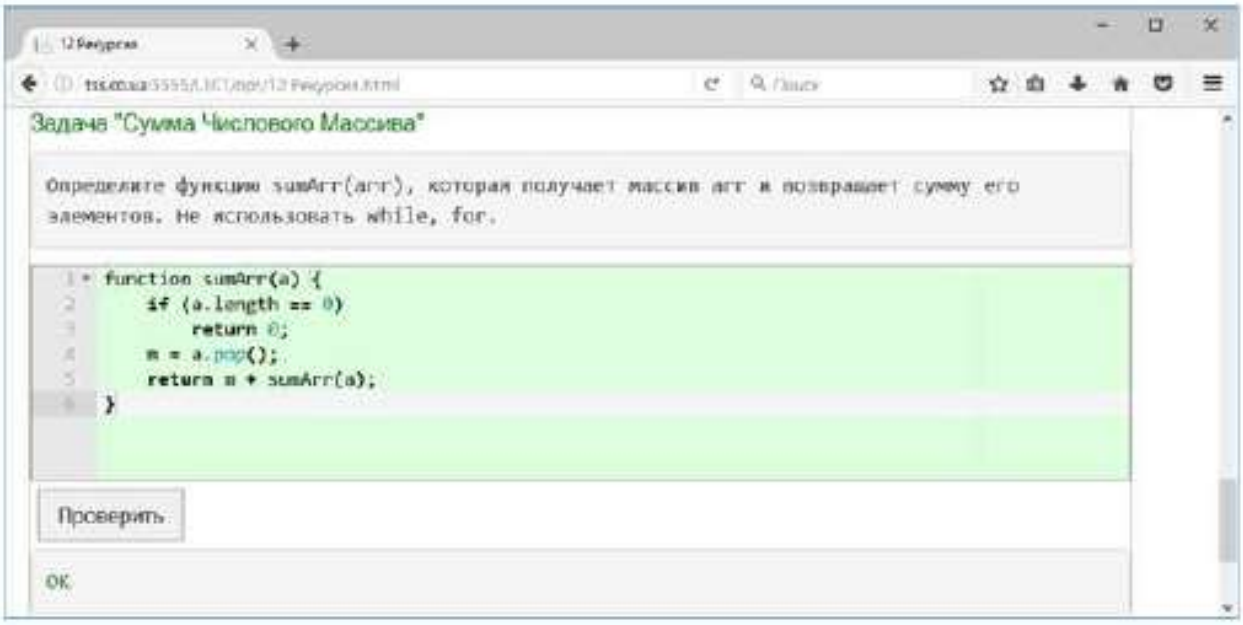

Рисунок 1 – Задача з автоматичною перевіркою

Коли перевірка рішень автоматизована, неважко організувати статистику і враховувати, хто скільки задач вирішив і яким чином. Викладач може спостерігати за процесом, аналізувати, але, очевидно, не повинен оцінювати успішність студентів за цими даними. Для оцінювання знань (для мотивації студентів) необхідно використовувати зв'язку задачник-служба у іншому режимі - у режимі іспиту. На «іспиті» студенти вирішують задачі за обмежений час і у присутності викладача, який повинен забезпечити аутентифікацію студентів і стежити, щоб вони працювали самостійно. Не можна сказати, що у епоху інтернету та мобільних пристроїв це дуже просто, але проблема вирішувана і справи йдуть не гірше, ніж на традиційних іспитах. Зауважимо, що свою мотиваційну функцію іспити виконають, якщо здійснювати їх регулярно, хоча б раз на два тижні. Так студентам стає зрозумілою необхідність вирішувати задачі самостійно.

Описана вище система у різних версіях експлуатується авторами вже третій рік та показала свою корисність та, як тепер їм здається, незамінимість. Помітно підвищилися залучення студентів у процес навчання, впевненість у об'єктивності оцінок, а головне, вміння програмувати.

Литература:

1. Online Judge – архів задач Вальядолідського університету в Іспанії. [Електронний ресурс]. Режим доступу: https://uva.onlinejudge.org/
2. Timus Online Judge - архів задач з перевіряючею системою. [Електронний ресурс]. Режим доступу: http://acm.timus.ru/
3. ejudge - система для проведення різних заходів, в яких необхідна автоматична перевірка програм. . [Електронний ресурс]. Режим доступу: http://ejudge.ru/
4. TopCoder – корпорація, яка проводить змагання зі спортивного програмування. [Електронний ресурс]. Режим доступу: http://www.topcoder.com/
5. CS Academy - освітній проект в сфері програмування. [Електронний ресурс]. Режим доступу: https://csacademy.com/
6. Абрамян М. Э., Михалкович С. С. Веб-среда разработки и обучения // Открытые системы. СУБД. 2012, No. 10. С. 56–59.
7. Programming Taskbook. Електронний задачник з програмування. [Електронний ресурс]. Режим доступу: http://ptaskbook.com/ru/.