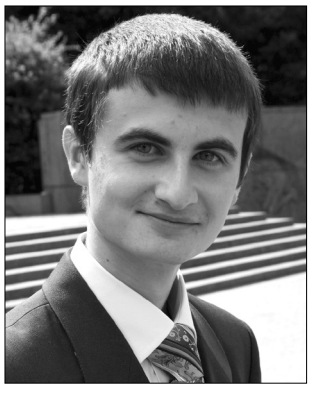
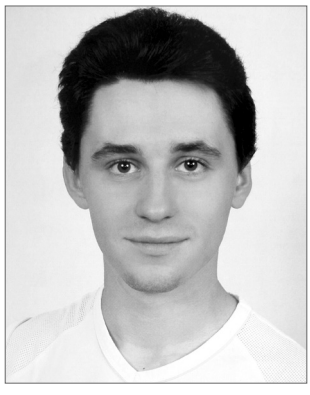
СИСТЕМА АВТОМАТИЧНОГО ТЕСТУВАННЯ «АСМ КОНТЕСТЕР»

Семен Ю., Бабіля Р.





Нині у світі і в Україні серед учнів та студентів проводиться багато олімпіад із програмування. Перевірка правильності розв'язання задач на більшості з них проводиться в автоматичному режимі за допомогою тестуючих програм. Участь у таких олімпіадах допомогла нам розробити систему автоматичного тестування «АСМ Контестер». Зупинимося спочатку на сутності чемпіонату за версією АСМ.

Чемпіонати світу з програмування за версією АСМ проводяться з 1977 року за багаторівневою системою (спочатку відбіркові змагання, потім — фінальні). У 1977–1985 рр. в змаганнях брали участь тільки північноамериканські команди. Першою командою не з Північної Америки, що взяла участь у змаганнях у 1986 році, була команда з Нової Зеландії. Генеральним спонсором змагань з сезону 1997–1998 років є компанія IBM.

Згідно правил чемпіонату світу в змаганнях можуть брати участь команди університетів у складі 3-х учасників. Кожен учасник повинен бути студентом або аспірантом відповідного університету. Водночас одній і тій же людині заборонено виступати більш ніж у двох фіналах. На час змагань (зазвичай, він становить п'ять годин) кожній команді з трьох учасників надається один комп'ютер з установленими на ньому засобами розробки програм і виходом на систему автоматичного (або напівавтоматичного, як це прийнято в фіналі) тестування. Для розв'язування учасникам, зазвичай, пропонується від 8-и до 12-и задач. Розв'язком кожної задачі повинна бути програма на одній з прийнятих на змаганнях мов програмування (переважно це Java, C, C++). Якщо команда вважає, що розв'язок готовий, вона відправляє його на тестування. Система тестує розв'язок на заздалегідь заданому автором задачі наборі тестів. Якщо на якомусь тесті програма некоректно завершилася, перевищила заданий в умові задачі час роботи на одному тесті або дала неправильну відповідь, то команда отримує коротке повідомлення про помилку. Водночас сам тест, на якому виникла помилка, команді не повідомляється. Якщо ж програма правильно відпрацювала на всіх тестах, то задача команді зараховується. Крім того, до «штрафного часу» додається час від початку змагань до моменту здачі задачі, а також по 20 хвилин за кожну невдалу спробу здати цю задачу. Перемагає команда, яка здає найбільше задач, а якщо таких команд кілька — та з них, яка отримала найменший «штрафний час».

Місця з 1-го по 4-е отримують золоті медалі, з 5-го по 8-е — срібні, місця з 9-го по 12-е — бронзові. Відбір на фінальні змагання проводиться за багатоступеневою схемою: усі університети світу розбиті на регіони (за територіальною ознакою), у яких проводяться півфінали. Найбільші регіони розбиті на підрегіони, а відбір у півфінал проводиться за результатами чвертьфіналів. Більше того, у деяких регіонах вже проводяться і 1/8 фіналу. Для кожного півфіналу кожен рік визначається квота команд, які виходять від даного півфіналу у фінал, при цьому з одного вузу в фіналі не може брати участь більше однієї команди. Квота, яка виділяється регіону, залежить від кількості команд, що брали участь у відбіркових змаганнях від даного регіону, результатів, показаних регіоном, а також від деяких інших факторів. За результатами фіналу також визначаються чемпіони континентів, тобто краща команда серед усіх, що вийшли у фінал з приписаних даному континенту регіонів.

Команда Львівського національного університету 2008 року в складі Василя Білецького, Остапа Коркуни та Руслана Бабілі, зайнявши 4-е місце, стала першою українською командою, що завоювала в фіналі чемпіонату світу з програмування золоті медалі.

Як визначається коректність розв'язків задачі на змаганнях? Існує багато систем тестування для проведення змагань і тренувань. Відповідно кожна із цих систем повинна відповідати на запитання: чи правильний розв'язок. Завдання — це, в основному, прикладні алгоритмічні задачі і для їх правильного розв'язання не достатньо досконало знати синтаксис мови програмування, знати базові типи мови та їх властивості і т. д. Бувають випадки, коли людина вміє визначити результати виконання конкретного програмного коду. Але у разі самостійного написання коду, який виконуватиме поставлену задачу, виникають труднощі. Отже, щоб оцінити вміння програмувати — необхідно перевірити вміння створювати програми. Тобто здатність будувати алгоритми.

Очевидно, що це не так просто. Варіантів програмного коду, який виконує одну і ту ж задачу, безліч. Ми не можемо навчити звичайну систему тестування відрізнити правильний код від неправильного.

Цю, на перший погляд, досить складну проблему можна вирішити так: введемо деякі обмеження та уточнення:

- розв'язок — це програмний код консольної програми;
- набір тестів — початкові дані задачі разом із еталонними результатами;

- якщо програма не знаходить результат вчасно — розв'язок неправельний;
- якщо в обчисленнях програма використовує забагато оперативної пам'яті — розв'язок теж неправильний;
- якщо в обчисленнях програма користувача хоча б один раз дала неправильну відповідь — розв'язок, знову ж таки, неправильний.

У цьому випадку можемо поставити перед користувачем завдання — розробити певний алгоритм, який буде отримувати початкові дані, і повертати результати роботи алгоритму. Отриманий результат система може автоматично перевірити на відповідність еталонному результату.

Для розвитку олімпіадного руху серед студентів та школярів України нами розроблена система автоматичного тестування «ACM Контестер». Дана система працює в режимі 24/7 і зарекомендувала себе як надійна, стабільна система, яка не потребує ніяких втручань у її роботу. Завдяки цьому користувачі системи мають змогу у будь-який зручний для них час перевірити, чи правильно побудований алгоритм.

Щоб почати користуватися системою, потрібно зайти на Інтернет-сторінку http://acm.lviv.ua, та пройти стандартну процедуру реєстрування. Завдання задач, які доступні учасникам, можна подивитися в електронному вигляді. На умови можна подивитись, використавши меню (пункт «Завдання»). У кожній умові вказується обмеження на час і пам'ять та посилання, за яким можна відіслати на перевірку розв'язок цієї задачі. Також задається формат вхідних та вихідних даних, їхні приклади. Формат вхідних та вихідних даних вашого розв'язку повинен точно співпадати із даними в умові. Інакше система перевірки сприйме це як неправильну відповідь. Для того щоб перевірити правильність формату даних, можна відіслати на перевірку лише на першому тесті.

Після ознайомлення з умовою завдання можна приступати до відшукання і написання розв'язку. Розв'язком вважається текст програми, написаний на одній з дозволених мов програмування, який у подальшому можна відкомпілювати відповідним компілятором та перевірити на всіх тестах. Якщо Ви написали розв'язок, необхідно відправити його на перевірку. Для цього необхідно перейти за посиланням в кінці умови, вибрати відповідний пункт меню. Водночас потрібно ввести логін та пароль учасника або команди, мову програмування, на якій написаний розв'язок, номер задачі, текст розв'язку задачі та вибрати режим перевірки (повністю чи лише на першому тесті).

Щоб перевірити правильність формату вихідних даних або у випадку виникнення труднощів із компіляцією програми, можна скористатися вже згаданим режимом «лише перший тест». Ваш розв'язок буде протестовано лише на одному тесті (перший тест з умови, вказаний у прикладах вхідних та вихідних даних). Результати цієї перевірки ні на що не впливають.

Після отримання розв'язку система працює в автоматичному режимі. Спочатку розв'язок компілюється у виконуваний файл. Для цього використовуються наступні компілятори: Delphi7, Microsoft Visual Studio C++ Express Edition. Потім отримана програма запускається на кожному тесті. На вхід даються відповідні вхідні дані, а програма повинна повернути результат, який порівнюється з еталонним. Якщо результати співпали і програма вклалася у відповідні обмеження, тоді цей тест зарахований і тестування продовжується на наступному тесті. Розв'язок вважається правильним, якщо всі тести зараховано.

Якщо алгоритм складений правильно, то про це повідомляється учасника, а також додатково кількість використаної пам'яті, час проходження по тестах програми, а також кількість тестів для цього розв'язку. У випадку невдалої спроби надається додатково інформація залежно від типу помилки:

- **Помилка компіляції** (Compilation Error) — допущена синтаксична помилка в тексті алгоритму і система не змогла скомпілювати розв'язок, або він не відповідає стандарту компілятора, який використовується в системі;
- **Помилка виконання** (Runtime Error) — під час виконання алгоритму відбулася критична помилка, за якої подальше виконання неможливе. Типові помилки: ділення на 0 (нуль), доступ до неіснуючого елементу в масиві. Також можлива ця помилка у випадку, якщо код завершення програми не дорівнює нулю;
- **Неправильна відповідь** (Wrong Answer) — алгоритм хоча б на одному з тестів подав неправильну відповідь, або формат виведення не відповідає умові;
- **Ліміт часу** (Time limit) — алгоритм не вклався у відведений для знаходження відповіді час. Цей ліміт подається в умові задачі і рахується для кожного тесту окремо. Час виконання програми — це загальний час від моменту запуску програми і аж до її повного завершення;
- **Ліміт пам'яті** (Memory limit) — алгоритм використовує більше пам'яті, ніж було для цього відведено;
- **Заборонена функція** — під час створення алгоритму використовується заборонена функція. Якщо це буде повторюватись досить часто — учасника буде дискваліфіковано за порушення правил.

Крім цього постійно підтримується база даних, у якій зберігаються всі спроби разом із результатами. Це дає змогу відображати на Інтернет-сторінці динамічний рейтинг, який додає певного стимулу. А також у разі необхідності провести «он-лайн» змагання. Досить вибрати час початку, тривалість та завдання, які будуть включені до цього змагання і воно розпочнеться автоматично в заданий час.

На даний момент у системі зареєстровано понад 970 користувачів та понад 150 завдань олімпіадного рівня різної складності. Зареєстровано понад 40 тисяч запитів на перевірку алгоритму та проведено 14 відкритих Інтернет-олімпіад і понад 20 олімпіад факультетського рівня.