УДК 519.8

МОДУЛЬНАЯ СТРУКТУРА ЭЛЕКТРОННОЙ СИСТЕМЫ ДЛЯ ОБУЧЕНИЯ ПРОГРАММИРОВАНИЮ

Цветцих Д.В.,
научный руководитель канд.техн. наук Редькина А.В.
Сибирский Федеральный Университет

Сегодня большая часть технических специальностей вузов предусматривает обучение программированию, где студентов учат написанию компьютерных программ. Важной особенностью проверки компьютерных программ является то, что сами тексты созданных программ или алгоритмов — решений задач, проверять гораздо труднее, чем решения задач по математике или физике (трудоемкость можно оценить не менее чем в 5 раз выше). Для проверки решений задач требуется большое число проверяющих — специалистов по программированию высокой квалификации. При очной форме обучения преподаватель, проверяющий прикладные компьютерные программы, может попросить студента объяснить соответствие алгоритма работы его программы с исходным текстом и, таким образом, несколько облегчить процесс проверки. На практике же обычно работоспособность программы оценивается не по ее внутренней структуре, а по результатам ее работы. При заочной форме обучения преподаватель, проверяющий подобную прикладную программу, часто не имеет возможности услышать комментарии студента по алгоритму ее выполнения и проверяет ее только по результатам работы программы.

Исходя из сказанного выше, видно, что выходом из данной ситуации могло бы стать тестирование результатов работы программы с возможным последующим анализом кода программы. Также процесс проверки результатов работы компьютерных программ можно автоматизировать, в отличие от процесса анализа исходных текстов. Таким образом, возникает проблема автоматизированного тестирования программ. А все существующие средства электронного обучения не предоставляют средств для решений данной проблемы.

Требования к структуре системы.

Сформулируем требования к структуре системы электронного обучения с поддержкой автоматической проверки решений.

Для обеспечения переносимости системы между платформами, облегчения процесса доработки нового функционала и исправления ошибок, а также для разграничения прав доступа, всю функциональность системы необходимо разделить на следующие подсистемы:

1) Подсистема администрирования. Обеспечивает регистрацию пользователей в системе, а также настройку прав и ролей.
2) Подсистема управления архивом задач. Для студентов обеспечивает просмотр задач, распределение задач по тематическим блокам при помощи рубрикатора. Для преподавателей предоставляет возможность расширения состава предлагаемых для решения задач, а также возможности по доработке условий задач и тестов.
3) Подсистема проверки решений. Всем участникам курса обеспечивает возможность отправки решений любой задачи на проверку и просмотра протокола тестирования решения. Также предоставляется возможность просмотра списка всех своих посылок, результатов их тестирования и исходного кода самих решений.
4) Подсистема интеграции в учебный процесс. Обеспечивает закрепление за студентами задач из архива в качестве заданий на лабораторные работы и

выставление в электронный журнал оценок по результатам решения студентами данных задач.

5) Подсистема мотивации. Для студентов обеспечивает возможность просмотра своего положения в рейтинге участников курса согласно количеству набранных баллов при решении задач.

Подсистема администрирования.

Подсистема администрирования осуществляет поддержку следующих ролей:

Студент. Осуществляет просмотр списка задач, отправляет решения задач на проверку, просматривает протокол тестирования задач, а также свой положение в рейтинге согласно количеству баллов, набранному при решении задач.

Преподаватель. Осуществляет наполнение архива задачами и тестовыми примерами, назначает студентам задания на лабораторные работы, выставляет в журнал оценки за решения задач.

Администратор. Осуществляет регистрацию студентов, учебных групп и преподавателей в системе и назначает им соответствующие роли.

Пользователи, имеющие в системе роль студента или преподавателя, не имеют доступа к функциям системы администрирования. Кроме того, в связи с тем, что система предназначена для использования при обучении студентов Сибирского Федерального Университета, гостевого доступа в системе не предусмотрено.

Подсистема управления архивом задач.

Просмотр общего списка задач. Подсистема управления архивом задач должна обеспечивать возможность просмотра списка всех заданий. Семестровый курс дисциплины «Программирование» может содержать до десяти лабораторных работ и до тридцати заданий на каждую лабораторную работу. Таким образом, общее количество задач в архиве будет около трехсот и для отображения данного списка необходимо использовать постраничную выборку.

Просмотр текста задачи. Любой студент, имеющий доступ к курсу, имеет возможность просматривать условия всех задач курса.

Просмотр задач по тематическим блокам. Задачи курса разбиваются по тематическим блокам (например, «Вычисление суммы ряда», «Одномерные массивы», «Строки символов»). Каждая задача должна принадлежать одному или нескольким тематическим блокам. Не допускается существование задач, не относящихся ни к одному тематическому блоку. Рубрикатор должен быть иерархическим для обеспечения удобного распределения задач по тематическим блокам (например, тематический блок «Графы» может иметь подразделы «Поиск в глубину» и «Поиск в ширину»).

Добавление, удаление, изменение задач. Позволяет преподавателям редактировать список заданий, предлагаемых студентам для решения.

Добавление, изменение, удаление тестов к задачам. Данная функциональность доступна только преподавателям. Не допускается удаление всех тестов к задаче.

Подсистема проверки решений.

Выбор компилятора (опциональная возможность). Предоставляет участнику курса возможность выбора компилятора, который будет использоваться при компиляции исходного кода решения. Так как практически все студенты пользуются одним и тем же компилятором (в СФУ это компилятор MicrosoftVisual C++), данная возможность не является необходимой. Однако реализация данной возможности позволит использовать систему при обучении разным языкам программирования, а также предоставит участникам курса относительную свободу выбора средств разработки лабораторных работ (выбор будет ограничен набором компиляторов, поддерживаемых системой).

Отправка решения. Любой участник курса может отправить на проверку любую задачу, независимо от того, закреплена эта задача за участником в качестве лабораторной, или нет. На проверку отправляется файл с исходным кодом решения на изучаемом языке программирования. Система осуществляет компиляцию исходного кода решений и запуск исполняемого файла решения на наборе тестов. При выполнении решения на каждом тесте измеряется время его работы решения и объем потребляемой памяти. Выходные данные решения сравниваются с ответом, и на основании этого сравнения делается вывод о корректности решения. Решение последовательно запускается на всех тестах. Если на одном из тестов решение выдаёт неправильный ответ, его тестирование не прекращается для формирования полного итогового протокола тестирования и начисления баллов за задачу.

Просмотр протокола тестирования. Участникам курса предоставляется возможность просмотра протокола тестирования решения на всех тестах. Для каждого теста указывается:

- Входные данные
- Выходные данные решения
- Корректные выходные данные
- Заключение о корректности работы решения на тесте
- Время работы решения
- Объем потребляемой памяти

Просмотр списка отправленных на проверку решений. Для всех участников курса, имеющих роль «Студент», обеспечивается возможность просмотра списка отправленных на проверку решений. Для каждого решения указывается задача, которую решал участник курса, количество баллов, начисленное за решение, максимальное время работы и наибольший объем использованной памяти. Кроме того, для всех решений доступен просмотр исходного кода и протокола тестирования.

Просмотр списка решений всех участников. Данная возможность доступна только участникам курса, имеющим роль «Преподаватель». От описанной выше возможности отличается только тем, что в списке доступны для просмотра не только свои решения, но и решения всех остальных участников курса.

Перепроверка решений. Данная возможность доступна только преподавателям и применяется после корректировки условий задачи или изменения тестов к задаче. Функция заключается в последовательном перезапуске проверки всех решений данной задачи в порядке отправки их на проверку студентом. При этом заново происходит формирование новых протоколов тестирования и начисление баллов за все посылки решения.

Валидация тестов. Функция заключается в запуске на тестах эталонного авторского решения и применяется для проверки корректности тестов к задаче после их изменения. Доступна только преподавателям.

Подсистема интеграции в учебный процесс.

Все функции подсистемы интеграции доступны только участникам курса, имеющим роль «Преподаватель».

Закрепление задач из архива в качестве заданий на лабораторные работы. У студентов есть возможность отправки на проверку любой задачи из архива, и за решение любой из задач студенту будут начислены баллы. Однако в электронный журнал будут выставлены баллы только за те задачи, которые закреплены за студентом как лабораторные заботы. Назначение задач на лабораторные работы происходит либо на всю группу, либо на каждого студента в отдельности. Назначение на группу используется, когда новая учебная группа начинает обучение дисциплине «Программирование». В этом случае на каждую лабораторную работу студентам

назначаются задачи из определённого тематического блока в соответствии с их порядковым номером в группе. Например, студенту с первым номером в списке группы на все лабораторные назначается первая задача из каждого тематического блока, студенту со вторым номером – вторая. Если в течение семестра возникает необходимость в смене заданий на лабораторные работы, преподаватель может изменить задачу на одну работу, при этом не изменяя задачи на остальные работы. Для назначения работ студентам, которые выходят в середине семестра или для быстрой смены вариантов должна быть предусмотрена функция переназначения заданий на все лабораторные работы по определённому номеру.

Выставление в журнал оценки по количеству пройденных решением тестов. По результатам проверки решения на тестах студент получает за задачу определённое количество баллов. Количество баллов, выставляемое за решение, определяется по формуле:

Passed*Score - Attempts*Penalty

Passed – количество тестов, на которых решение дало верный ответ

Score – количество баллов, начисляемое за один тест (вычисляется исходя из максимального количества баллов, которое можно набрать за решение задачи, и количества тестов)

Attempts – количество попыток решения данной задачи студентов

Penalty – штрафные баллы за дополнительные попытки

Таким образом, если решение задачи с первой попытки проходит все тесты, он получает максимальное количество баллов за задачу. В случае, когда студент предпринимает дополнительные попытки решения задачи, за каждую попытку начисляются штрафные баллы, которые вычитаются из результата. Результативной (той, за которую выставляются баллы в журнал) считается последняя попытка. Отрицательное количество баллов за задачу не может быть начислено. В случае, когда размер штраф превышает количество набранным решением баллов, за решение выставляется ноль баллов.

Подсистема мотивации

Предназначена для повышения интереса студентов к решению большего количества задач. Общие правила построения рейтинга таковы: за решение любой задачи, независимо от того, закреплена ли она за студентом в качестве лабораторной работы, начисляются баллы. Рейтинг – список студентов, упорядоченный по убыванию количества набранных баллов.

Функции подсистемы рейтингов доступны всем участникам курса. Любой студент может отслеживать свой прогресс одновременно в трёх рейтингах:

- Рейтинг студентов своей группы
- Рейтинг студентов, начавших обучение в текущем учебном году
- Абсолютный рейтинг всех участников курса

В рейтинговой таблице указывается фамилия студента, учебная группа, год поступления, количество решенных задач, количество набранных баллов. За первые места в соответствующих рейтингах напротив фамилий студентов должны отображаться следующие надписи:

- Лучший программист группы
- Лучший программист по итогам года
- Лучший программист ИКИТ СФУ