УДК 004.421

ДИЗАЙН ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА С ИСПОЛЬЗОВАНИЕМ РЕСУРСА LEETCODE ПРИ ПОДГОТОВКЕ IT СПЕЦИАЛИСТОВ

Глущенко Т.А., Савицкий Ю.В.

Брестский государственный технический университет, г. Брест, yurysav1971@gmail.com

Аннотация. Рассмотрены возможности применения ресурса LeetCode для подготовки IT специалистов, представлены практические аспекты использования данного ресурса на примере курса «Алгоритмы и структуры данных», читаемого для студентов специальности 6-05-0612-01 Программная инженерия на кафедре интеллектуальных информационных технологий Брестского государственного технического университета.

Ключевые слова. Ресурс LeetCode, алгоритм, структура данных, обучение IT специалистов.

Изучение алгоритмов и структур данных является важнейшей и неотъемлемой частью подготовки IT специалистов. В большинстве университетов мира для студентов IT-специальностей читаются отдельные курсы по алгоритмам и структурам данных или же данная проблематика рассматривается в параллельных курсах.

И хотя порой в среде программистов или Интернет-пространстве возникают дискуссии о целесообразности изучения алгоритмов и структур данных, такая необходимость очевидна.

Рассмотрим преимущества изучения алгоритмов и структур данных [1]:

– *повышение эффективности*. Существуют максимально быстрые алгоритмы для обработки данных и эффективные структуры для их организации. Такие паттерны избавляют от необходимости «изобретать велосипед» при решении типовых задач, позволяют прогнозировать производительность ПО и служат основой для разработки новых нетривиальных решений;

– *развитие аналитических способностей и алгоритмического мышления.* Изучение и реализация алгоритмов улучшают не только навыки программирования: привычка разбивать сложные задачи на логически связанные шаги, необходимые для их эффективного решения, пригодится везде – от планирования отпуска до разработки инвестиционной стратегии;

– *успехи в спортивном программировании*. Знание алгоритмов необходимо для успешного решения олимпиадных задач. Стоит заметить, что олимпиадные задачи часто предлагают кандидатам на техническом собеседовании;

– *успешное прохождение собеседований*. Чем сложнее задачи, которые предстоит решать разработчикам компании, тем выше вероятность, что значительная часть вопросов будет посвящена алгоритмам и структурам данных. Работодатели отдают предпочтение кандидатам, которые способны найти самое эффективное и эргономичное решение проблемы.

В крупных IT-компаниях, таких как Google, Amazon, IBM, Oracle, Яндекс, Microsoft и других IT-гигантах, алгоритмическое собеседование – обязательный этап отбора разработчиков. На нём проверяют умение быстро отразить идею в коде. Но знание алгоритмов требуют не только IT-гиганты – для многих компаний это базовый навык хорошего инженера [2].

Зачастую задача успешного прохождение собеседований является основным стимулом к изучению алгоритмов и структур данных для программиста.

Общая характеристика ресурса LeetCode.

Существует множество ресурсов для изучения данной темы. Это и топовые книги по алгоритмам и структурам данных, и различные онлайн-курсы, например, курс Algorithms от Coursera, и специализированные веб-сайты, и Github библиотеки, и готовые подборки ресурсов для изучения алгоритмов и структур данных, публикуемые экспертами [3–5].

Одним из характерных ресурсов для практики и совершенствования навыков по программированию является LeetCode. Это сервис с алгоритмическими задачами, которые помогут подготовиться к собеседованию. Задачи охватывают все аспекты разработки, включая базы данных, алгоритмы, теорию графов, структуры данных и динамическое программирование. Кроме задач доступны обучающие планы, чтобы освоить тему с нуля. Для специалистов доступны инструменты, которые помогут тестировать, отлаживать и даже писать собственные проекты онлайн [5, 6].

Ресурс LeetCode имеет большие возможности для обучения, и разработчик самостоятельно выбирает стратегию обучения. Остановимся на положительных аспектах LeetCode по мнению разработчиков:

– сервис помогает найти путь решения проблемы, а не просто подобрать подходящий алгоритм. Это помогает в работе, когда нужно отследить путь появления ошибки, выявить причину;

– повышается скорость закрытия простых задач, развивается навык решения на автомате – в деятельности разработчиков тоже есть рутина, некоторые задачи нужно выполнять быстро, но при этом качественно;

– количество ошибок становится гораздо меньше, ведь перед тем, как отправить готовый ответ, нужно по несколько раз проверить правильность решения. Это помогает учитывать и просчитывать сразу несколько вариантов развития событий;

– легче учить другие языки сразу на практике, параллельно погружаясь в теорию в виде материалов на сайте. Новички особенно стараются сделать код чистым, написать к нему понятное объяснение.

– развитие самодисциплины [7].

Задачи использования ресурса LeetCode в обучении студентов IT-специальностей.

Как уже отмечалось ранее, ресурс, в основном, используется для подготовки программистов к техническим собеседованиям. Вторым аспектом его использования является самостоятельное изучение алгоритмов и структур данных, и как следствие, повышение уровня решения алгоритмических задач конкретного программиста.

Можно ли использовать ресурс в процессе обучения студентов IT-специальностей? Безусловно.

Отметим задачи, которые решаются в процессе обучения: более детальное изучение алгоритмов и структур данных; получение практики решения прикладных алгоритмических задач; получение навыка тестирования программ; получение практики написания *«чистого кода»*; умение оценивать временную и емкостную сложность алгоритмических задач; повышение уровня английского языка.

Как видим, решаемые задачи для студентов и программистов во многом перекликаются.

Раскроем и обоснуем указанные пункты.

Практически для всех студентов IT-специальностей предусмотрены курсы по изучению алгоритмов и структур данных. Например, для специальности 6-05-0612-01 Программная инженерия в БрГТУ курс так и называется «Алгоритмы и структуры данных». Охватить и детально рассмотреть все аспекты структур данных и алгоритмов в годичном курсе просто невозможно, курс построен скорее «вширь», что вполне обосновано. Ресурс как раз и позволяет начать изучение «вглубь», раскрывая в конкретной задаче нюансы и возможности применения структур данных и алгоритмов.

Тестирование – это важная часть написания программы (программного продукта). Обычно студент, написав программу, проверяет ее на 3–4 тестовых примерах, выбор которых на данный момент соответствует уровню его знаний и понимания решаемой задачи, порой не учитывая все граничные условия, исключения или варианты входных данных. Ресурс же «прогоняет» написанную программу на десятках вариациях входных данных и может оказаться, что не прошел, к примеру, единственный тест из 40. А это и заставляет студента рассматривать проблему «вглубь», искать какие нюансы он не учел.

«Чистый код», иногда говорят *«белый код»*, оттачивается не столько практикой, сколько видением и пониманием чужого чистого кода и, как следствие, использование приемов и методов «чистого кода». В разделе Discuss программисты выкладывают свои решения, которые не только оптимально решены, но и *«красиво»* написаны.

Программист должен уметь обязательно оценивать временную и емкостную сложность решаемых задач, поскольку от этого напрямую зависит быстродействие работы программного продукта и затраты на оборудование. В некоторых задачах ресурса иногда сразу указывается требуемая временная или емкостная сложность, например, *O(n).* По умолчанию предполагается, что необходимо решить задачу наиболее оптимально, для чего и служат сравнительные диаграммы времени выполнения программы или затрат по памяти в сравнении с вариантами других программистов. И если предложенное решение находится в диапазоне меньше 60%, возможно решение не оптимально и есть возможность улучшить результат.

Несмотря на очевидные плюсы использования ресурса в процессе изучения алгоритмических курсов, следует сделать два замечания:

– если дать студенту какую-либо задачу с ресурса, то студент 2-го курса скорее всего самостоятельно ее не решит, в крайнем случае решит ее «в лоб»;

– преподаватель должен либо сам уметь решать задачи с ресурса, а это порой олимпиадный уровень программирования, либо составить базу разобранных и решенных задач с ресурса, используя раздел Discuss либо описание задач с других Интернет-ресурсов.

Предположим, что вторая проблема решена и осталась только задача обучения студентов.

И тут возможны несколько вариантов или комбинации этих вариантов.

1. Преподаватель описывает алгоритм. Предпочтительным является еще и диалоговое обсуждение алгоритма. Студент должен понять алгоритм, написать оптимальную программу и успешно протестировать ее.

2. Приводится оптимальный код программы, принятый ресурсом, и студенту предлагается описать по пунктам алгоритм решения задачи. Студент обучается читать чужой код, что по началу может быть и нелегко и что является необходимым навыком каждого программиста. И, конечно, при данном варианте студент изучает алгоритм и учится писать чистый код.

3. Студенту дается условие (порядковый номер) задачи, предлагается найти оптимальный алгоритм решения задачи, реализовать его и успешно протестировать. Поскольку, как уже отмечалось, самостоятельное решение алгоритмических задач для студентов может быть сложным, студент может зайти в раздел Discuss и посмотреть предлагаемые решения с их описанием. Описание ведется на английском языке и его приводит конкретный программист, оно может быть и не всегда понятным, и не всегда развернутым, зато однозначно учит студента разбираться в возникающих проблемах. Данный вариант предполагает полную самостоятельную работу студента и приобретение указанных ранее навыков очевидно.

Также может приводиться описание алгоритма и кода программы, где строчка кода либо пропущена, либо указана неверно.

Выбор конкретных вариантов использования ресурса для обучения студентов диктуется целями, которые преподаватель ставит в лабораторной работе, его креативностью и творческим подходом.

На кафедре интеллектуальных информационных технологий БрГТУ ряд преподавателей используют ресурс, причем используют его по-разному.

Ниже рассмотрены возможные варианты использования ресурса на практических примерах.

Практика использования ресурса LeetCode в обучении студентов IT-специальностей.

Выше рассматривались варианты и идеи использования ресурса в обучении студентов IT-специальностей.

Сейчас рассмотрим конкретную реализацию этих идей на примере курса «Алгоритмы и структуры данных», читаемого для студентов специальности 6-05-0612-01 Программная инженерия. Как видим из названия курса, он перекликается с названием соответствующих разделов ресурса и, соответственно, темы рассматриваемых задач и вопросов во многом тоже перекликаются. Именно задачи раздела Algorithms и используются в процессе обучения.

На ресурсе также есть задачи, выделенные в отдельные группы: по деревьям, строкам, хеш-таблицам и другим структурам данных и алгоритмам.

Пример 1. При изучении хеш-таблиц на лабораторной работе дается задача построения полиномиальной хеш-функции для строк со всеми стандартными операциями и задача *1. Two Sum* ресурса LeetCode. В предлагаемом решении задачи применяется нетривиальный подход использования хеш-таблицы, что требует описания и обсуждения алгоритма. Временная сложность предлагаемого решения $O(n)$ против сложности $O(n^2)$ при стандартном подходе.

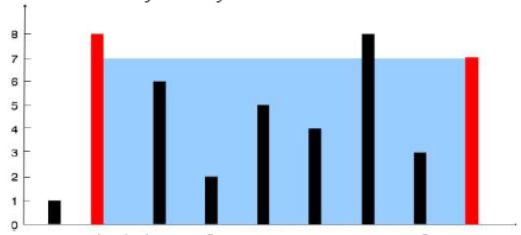
Пример 2. При изучении темы «Деревья» целая лабораторная работа выделена на решение задачи *208. Implement Tree (Prefix Tree).* Задача предусматривает построение префиксного дерева со всеми стандартными операциями. Тема «Деревья» подробно рассматривается в лекционном курсе, и поэтому указанная задача не предполагает дополнительного описания.

Пример 3. Приведем пример описания задачи 1 *1. Container With Most Water* в лабораторной работе, посвященной жадным алгоритмам (рисунок 1).

11. Container With Most Water

Given n non-negative integers $a_1, a_2, \ldots, a_n$, where each represents a point at coordinate (i, a_i). n vertical lines are drawn such that the two endpoints of the line i is at (i, a_i) and $(i, 0)$. Find two lines, which, together with the x-axis forms a container, such that the container contains the most water.

Notice that you may not slant the container.



Input: height = [1,8,6,2,5,4,8,3,7]
Output: 49
Explanation: The above vertical lines are represented by array [1,8,6,2,5,4,8,3,7]. In this case, the max area of water (blue section) the container can contain is 49.
Example 2:
Input: height = [1,1]
Output: 1

Рисунок 1 – Описание задачи «11. Container With Most Water»

Идея алгоритма.

Идея алгоритма состоит в вычислении площади контейнера, начиная с наибольшей длины, и затем на каждой итерации на единицу уменьшаем длину контейнера и вновь вычисляем площадь, сравнивая ее с ранее вычисленным максимальным значением. В случае, если новое значение площади превышает ранее вычисленное максимальное значение, то максимальному значению присваивается новое значение.

Проводим итерации до достижения минимальной длины контейнера и возвращаем максимальное значение площади.

При этом надо учитывать, что уменьшать длину на единицу на каждой итерации можно с двух сторон. При уменьшении длины со стороны более высокого столбика, новая площадь никогда не будет больше предыдущей. Поэтому уменьшение длины на каждой итерации должно происходить только со стороны более низкого столбика.

Алгоритм решения задачи является жадным, поскольку на каждой итерации мы заведомо выбираем оптимальное решение, его также можно отнести к методу двух указателей.

Подсчет площадей и их сравнение на каждой итерации цикла для данного примера выглядит так: 1) 8 > 0; 2) 49 > 8; 3) 49>18; 4) 49 > 40; 5) 49 > 16; 6) 49 > 15; 7) 49 > 4; 8) 49 > 6. Эффективность предлагаемого решения отражена на рисунке 2.

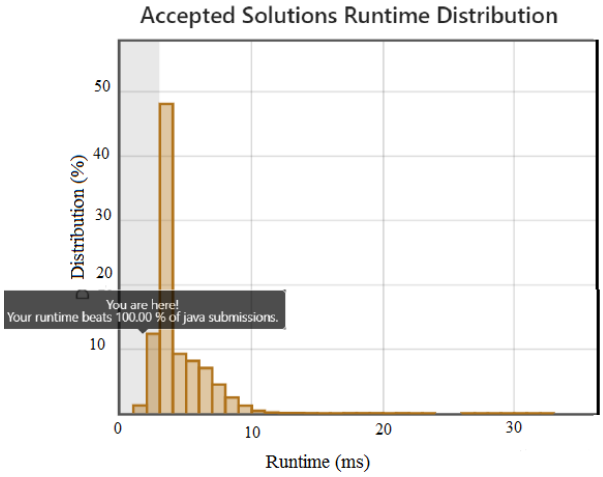


Рисунок 2 – Распределение принятых решений по времени выполнения

Описание алгоритма.

1. Устанавливаем индексы на *крайние элементы* массива, левый на *i*, правый – на *j*, соответственно, *left* и *right* – значения элементов для левого и правого индекса в массиве.

2. Начальное значение искомой величины *value* приравниваем к *нулю.*

3. Определяем, какое из значений *left* или *right меньше (высота столбика)* и для него вычисляем площадь воды.

4. Сравниваем полученную в пункте *4* площадь с вычисленным ранее значением. При необходимости меняем значение *value.*

5. Если выполняется неравенство *left* < *right*, увеличиваем *i* на единицу, в противном случае – уменьшаем *j*

на единицу. Двигаем индексы фактически навстречу друг другу.

6. Повторяем п.п. 3–5 пока выполняется условие $i < j$, по окончанию цикла получаем искомое значение величины *value*.

Вопросы по обработке алгоритма.

1. Чему равна временная сложность алгоритма?

2. Произвести подсчет площадей и их сравнение на каждой итерации цикла для массива значений: [5 2, 7, 8, 4, 6, 3].

3. В каких алгоритмах еще используется идея переноса границ?

В лабораторной работе также приведена реализация алгоритма на *Java*, т. е. описание задачи максимально возможное. И это позволяет варьировать описание задачи: оставить только идею алгоритма, или только описание алгоритма, или только листинг кода, или какую-то их комбинацию.

Интересным вариантом является пропуск пункта в описании алгоритма, в этом случае можно привести еще и код или идею алгоритма.

В конце каждой лабораторной работы приведены контрольные вопросы, которые должны выделять основные моменты алгоритма. Хорошей идеей для понимания алгоритма студентом является пошаговая или поитерационная его разбивка на конкретном наборе входных данных.

Выбор вариантов описания задачи зависит от многих факторов: цели и задач данной лабораторной работы, креативности преподавателя, уровня подготовки учебной группы.

Таким образом, каждый преподаватель, использующий в процессе обучения ресурс LeetCode, в том числе и на кафедре интеллектуальных информационных технологий БрГТУ, находит свой подход, свои авторские приемы по использованию ресурса.

Выводы.

В работе рассмотрен ресурс LeetCode как обучающая платформа по алгоритмам и структурам данных. Ресурс может использоваться как опытными разработчиками, так начинающими программистами и студентами. Работа с ресурсом требует предварительной теоретической подготовки. Ресурс имеет широкий выбор алгоритмических задач различной сложности, которые могут быть сгруппированы по темам. При возникновении сложностей с решением задач всегда можно обратиться к разделу Discuss, где разработчики выкладывают свои порой нетривиальные решения и объяснения.

Работа с ресурсом приносит неоценимую пользу, если заниматься на нем регулярно и систематически. Плюсы таких занятий очевидны: это и умение решать все более сложные алгоритмические задачи, и увеличение скорости написания стандартных заданий, и уменьшение количества ошибок, и практика «чистого» кода. Каждый практикующий LeetCode разработчик может внести свои пункты.

Как было показано, ресурс может использоваться как обучающий инструмент при изучении алгоритмов и структур данных студентами. В этом случае варианты его использования зависят от креативности преподавателя, его способности заинтересовать и увлечь студентов, не забывая при этом обучающую составляющую.

Литература

1. Зачем разработчику знать алгоритмы и структуры данных? [Электронный ресурс] / proglib. io. − Режим доступа: https:/ /proglib.io/p/zachem- razrabotchiku-znat-algoritmy-i-struktury-dannyh-2022-06-08. – Дата доступа: 23.10.2023.

2. Зачем программисту изучать алгоритмы [Электронный ресурс] / Tproger. ru. − Режим доступа: https:// tproger.ru/articles/why-learn-algorithms. – Дата доступа: 23.10.2023.

3. Изучаем алгоритмы: полезные книги, веб-сайты, онлайн-курсы и видиоматериалы [Электронный ресурс] / proglib.io. − Режим доступа: https:/ /proglib.io/p/ awesome-algorithms. – Дата доступа: 24.10.2023.

4. Где практиковаться начинающему разработчику [Электронный ресурс] / blog.skillfactory.ru. – Режим доступа: https://blog.skillfactory.ru/gde-praktikovatsya/ -nachinayushhemu-razrabotchiku. – Дата доступа: 24.10.2023.

5. Тренажер программирования LeetCode – что это и как его использовать [Электронный ресурс] / blog.tutortop. − Режим доступа: https://blog.tutortop.ru/ trenazher-programmirovania-leetcode. – Дата доступа: 23.10.2023.

6. Как правильно решать задачи на LeetCode: подробный гайд по тренажеру для программистов [Электронный ресурс] / skillbox.ru. − Режим доступа: https:// skillbox.ru/media/code/kak-pravilno-reshat-zadachi-na-leetcode-podrobnyy-gayd-po-trenazhyeru-dlya-programmistov. – Дата доступа: 23.10.2023.

7. Как задачи на LeetCode прокачали меня как разработчика, или по-честному про алгоритмы [Электронный ресурс] / habr.com. − Режим доступа: https://habr. com/ru/articles/747970. – Дата доступа: 23.10.2023.

DESIGN OF THE EDUCATIONAL PROCESS USING LEETCODE RESOURCE FOR TRAINING IT SPECIALISTS

T.A. Glushchenko, Y.V. Savitsky

Brest State Technical University, Brest, Belarus, yurysav1971@gmail.com

Abstract. The possibilities of using the LeetCode resource for the training of IT specialists are considered, practical aspects of using this resource are presented using the example of the course “Algorithms and Data Structures” taught for students of the specialty 6-05-0612-01 Software Engineering at the Department of Intelligent Information Technologies of Brest State Technical University.

Keywords. LeetCode resource, algorithm, data structure, training for IT specialists.