Научная статья
УДК 004.424.5, 004.021
DOI: 10.14529/ctcr220402

ОПТИМИЗАЦИЯ АЛГОРИТМОВ В ГОЛОВОЛОМКЕ HUMAN RESOURCE MACHINE ПО ЧИСЛУ КОМАНД И ОПЕРАЦИЙ

А.В. Копотева[1], *kopoteva_av@mail.ru*
Е.А. Кабиольский[2], *evgkab@susu.ru*

[1] *Пермский национальный исследовательский политехнический университет, Березниковский филиал, Березники, Россия*
[2] *Южно-Уральский государственный университет, Челябинск, Россия*

Аннотация. Видеоигры являются одним из распространенных видов современного досуга, а также относятся к синтетическому искусству коллективного авторства. Достаточно много написано о вреде компьютерных игр, однако есть и исследования, свидетельствующие о пользе некоторых их видов, в том числе различных головоломок. Часть из них являются весьма специфическими продуктами, в частности, различные симуляторы программирования. **Цель исследования.** Данная работа посвящена оптимизации наиболее сложных алгоритмов в видеоигре Human Resource Machine в соответствии с принятыми в ней критериями оптимальности. Предполагается, что решение каждой задачи должно содержать не более заданного числа команд и выполняться не более чем за определенное количество шагов. Для большей части задач может быть найдено решение, одновременно удовлетворяющее обоим критериям, однако некоторые головоломки требуют реализации двух отдельных алгоритмов. Всего в игре 36 задач различного типа и уровня сложности, из них выбрано 6 доставивших автору наибольшие трудности при прохождении. **Материалы и методы.** В работе описываются идеи и алгоритмы решения задач на естественном языке, удовлетворяющие заданным ограничениям, а также приводятся результирующие значения количества команд в коде и операций в процессе его исполнения. Точные решения не приводятся, поскольку в силу громоздкости игрового языка соответствующие алгоритмы с комментариями оказались слишком объемны для статьи. Кроме того, данная статья призвана лишь констатировать возможность нахождения собственных оптимальных решений рассматриваемых головоломок. **Результаты.** Для двух задач из шести рассмотренных в работе удалось получить алгоритмы, удовлетворяющие ограничениям и по числу команд, и по количеству операций. Для оставшихся четырех задач разработано по паре алгоритмов, каждый из которых удовлетворяет одному из ограничений. **Заключение.** По результатам проделанной работы можно утверждать, что существуют компьютерные игры, которые являются не просто развлечением, а могут представлять собой весьма непростой интеллектуальный вызов. Нахождение оптимальных решений в головоломках Human Resource Machine оказалось весьма непростым и интересным занятием, позволившим избавиться от шаблонов и иначе взглянуть на некоторые классические задачи программирования.

Ключевые слова: видеоигра, головоломка, программирование, сортировка, оптимизация алгоритма

Для цитирования: Копотева А.В., Кабиольский Е.В. Оптимизация алгоритмов в головоломке Human Resource Machine по числу команд и операций // Вестник ЮУрГУ. Серия «Компьютерные технологии, управление, радиоэлектроника». 2022. Т. 22, № 4. С. 16–26. DOI: 10.14529/ctcr220402

Original article
DOI: 10.14529/ctcr220402

COMMANDS AMOUNT AND EXECUTION STEPS OPTIMIZATION IN HUMAN RESOURCE MACHINE PUZZLES SOLUTION ALGORITHMS

A.V. Kopoteva*[1], *kopoteva_av@mail.ru
E.A. Kabiolskiy*[2], *evgkab@susu.ru
[1] *Perm National Research Polytechnic University, Berezniki Branch, Berezniki, Russia*
[2] *South Ural State University, Chelyabinsk, Russia*

Abstract. Videogames are one of the most common forms of modern leisure, and also can be viewed as a form of collective authorship synthetic art. We often hear about computer games health hazard, but there are also studies indicating the benefits of some of their types, including various puzzles. Some puzzle videogames are very specific products, in particular, various programming simulators. **Aim.** In this work we consider the most complex algorithms optimization in the video game Human Resource Machine according to in-game optimality criteria. It is assumed that each task solution should contain no more than a given number of commands and be executed in no more than a certain number of steps. For most problems, a solution that simultaneously satisfies both criteria can be found, but some puzzles require two separate algorithms development. There are 36 tasks of various types and difficulty in the game, we selected 6 of them that proved to be the most difficult for optimization. **Materials and methods.** The issue describes chosen problems solving ideas and algorithms in natural language that satisfy both in-game optimality criteria, and also provides the resulting values of code commands number and amount of execution operations. Exact solutions are not included in the issue, because due to the game language cumbersome nature the corresponding algorithms with comments are too ponderous for the article. In addition, this article's purpose is only to state the possibility and basic ideas of optimal solutions to the puzzles under consideration and not describing solutions themselves. **Results.** For two of the six tasks considered in the issue, we managed to obtain algorithms that satisfy the limitations both in the numbers of commands and operations. For the remaining four tasks, a couple of algorithms have been developed, one for each constraint. **Conclusion.** Based on the results of this issue we can confirm that there are computer games that are not just for entertainment, sometimes they can offer a real intellectual challenge. Finding optimal puzzles solutions in Human Resource Machine turned out to be a difficult and interesting task, which allowed us to break patterns and take a different look at some classical programming problems.

Keywords: videogame, puzzle, programming, sorting, algorithm optimization

For citation: Kopoteva A.V., Kabiolskiy E.A. Commands amount and execution steps optimization in Human Resource Machine puzzles solution algorithms. *Bulletin of the South Ural State University. Ser. Computer Technologies, Automatic Control, Radio Electronics*. 2022;22(4):16–26. (In Russ.) DOI: 10.14529/ctcr220402

Введение

Одной из распространенных и стремительно развивающихся сфер индустрии развлечений в настоящее время являются компьютерные и консольные видеоигры. Количество платформ и жанров на рынке весьма велико, а качество и содержание видеоигр существенно варьируют. Несмотря на более чем полувековую историю существования видеоигр как феномена и продукта, исследования, посвященные их месту в современной культуре и влиянию на человека, его физическое здоровье, способности и психику, активно ведутся и в настоящее время. Наличие сюжетной, графической и музыкальной составляющих, а также авторства (хоть и зачастую коллективного) позволяет считать видеоигры видом массовой культуры, обладающим элементами современного синтетического искусства [1, 2]. Помимо отрицательных последствий увлечения видеоиграми (ухудшение зрения, проблемы с опорно-двигательным аппаратом, зависимость) исследователи выделяют и положительные их виды [3]. К ним относят, в частности, повышение способности к концентрации, увеличение скорости реакции, развитие памяти, логики, способности решать нестандартные задачи, возможность обучения вообще и изучения иностранных языков в частности. Например, в [4] устанавливается положительное влияние игры в тетрис на мозговую и

сердечную деятельность, а в [5] обсуждается положительное влияние компьютерных игр на людей, вынужденных находиться в изоляции в период пандемии COVID-19. В [6] подтверждается возможность улучшения навыков алгоритмизации у учащихся с помощью видеоигры с элементами программирования. Авторы [7] устанавливают комплексное улучшение когнитивных функций лиц, 20 часов игравших в головоломки, основанные на физике. С нашей точки зрения видеоигра как любая иная форма досуга может быть просто деятельностью с целью отдыха или способом саморазвития и самореализации. Если мы говорим об игре как о способе отдыха, то это могут быть игры совершенно разных жанров с интересным сюжетом, оригинальной идеей, необычной реализацией или стилизацией, как минимум приятной графикой и музыкальным сопровождением. Если же мы говорим об игре как способе саморазвития, то это, во-первых, обучающие игры, которые в настоящее время активно вовлекаются в образовательный процесс [8, 9], а во-вторых, головоломки. Исследования, посвященные головоломкам и их решению, ведутся достаточно активно, причем часть изучаемых игр существует как самостоятельная форма досуга (например, судоку [10], кроссворды [11], лабиринты [12]), другие же возникли и получили распространение именно в формате видеоигры (например, сокобан [13, 14] и сапер [15]). Еще одна группа видеоигр-головоломок посвящена решению различных задач программирования. Игровые продукты такого рода являются нишевыми, разрабатываются небольшими студиями или даже единственным автором, имеют индивидуальный визуальный стиль, реализуют собственный язык программирования и предлагают игроку набор задач увеличивающейся по мере прохождения сложности, которые могут быть решены различными способами. При этом нам не удалось найти исследования, посвященные разработке алгоритмов решения таких задач, обнаружились лишь обсуждения в игровом сообществе Steam и прохождения в Интернет. Данная работа призвана восполнить этот пробел, поскольку некоторые задачи в таких играх весьма нетривиальны и заслуживают внимания.

Описание головоломки Human Resource Machine

Прекрасным примером игры-головоломки для программистов является Human Resource Machine от звездной студии Tomorrow Corporation, авторов World of Goo и Little Inferno. Это студия, которая создает маленькие шедевры, обладающие любопытным сюжетом, оригинальной идеей, нешаблонными игровыми механиками, самобытной графикой, отличным музыкальным и звуковым сопровождением. Ну и в случае Human Resource Machine и ее продолжения 7 Billion Human – это еще и разнообразные задачи-головоломки на программирование. Причем самое интересное заключается в том, что задачи можно не просто решить, а решить в соответствии с двумя различными критериями оптимальности. И если нахождение произвольного решения вполне достижимо, то оптимизация в части задач, собственно, и является головоломкой и представляет собой весьма нетривиальный процесс. Всего в игре 36 задач возрастающей сложности, предполагающих разработку алгоритма преобразования набора входных данных в выходные в соответствии с определенными требованиями. 25 задач образуют сюжетную линию игры, а 11 являются необязательными для прохождения (бонусными) в силу своей повышенной сложности. Естественно, чем больше номер задачи, тем выше ее сложность в теоретическом и прикладном плане. С точки зрения программирования все задачи достаточно простые и нахождение их решения человеку, обладающему некоторыми навыками программирования, не представляет существенной сложности. Наиболее нетривиальным моментом игры является оптимизация соответствующих алгоритмов («испытания оптимизации»): каждая задача может быть решена с ограничениями на количество команд в коде и на число операций в процессе выполнения кода. Для части задач удается найти решения, удовлетворяющие обоим ограничениям, для других приходится разрабатывать алгоритмы, удовлетворяющие каждому ограничению в отдельности. В данной статье рассмотрены лишь те задачи, оптимизация которых потребовала существенных умственных усилий. Все решения, о которых далее пойдет речь, найдены автором самостоятельно, хотя информацию об алгоритмах решения некоторых задач и особенностях их реализации внутриигровыми средствами пришлось искать в Интернете. Совокупное время, проведенное автором в игре, – около 45 часов. Естественно, наибольшим интеллектуальным вызовом стали некоторые бонусные и последний сюжетный уровень, о них и пойдет речь далее. Заметим, что внутриигровой язык программирования достаточно громоздок, а наличие готового решения потенциально лишает чита-

теля удовольствия нахождения собственного, поэтому мы будет излагать лишь идею и алгоритм решения и приводить результаты работы соответствующего алгоритма.

Рассмотрим решения 6 наиболее сложных с точки зрения оптимизации уровней Human Resource Machine. Это 5 бонусных задач («Числа Фибоначчи», «Сортировка трех», «Упорядочиватель», «Подрыв чисел» и «Фабрика простоты») и последняя сюжетная задача «Этаж сортировки».

1. Числа Фибоначчи: для каждого значения на входе вывести последовательность чисел Фибоначчи, его не превышающих. Испытание объема: использовать не более 19 команд, испытание скорости: выполнить не более чем за 156 операций.

Авторское решение: два алгоритма, 18/172 и 28/96.

Идея решения, удовлетворяющего испытанию объема, заключается в том, что последовательность чисел начинается не с двух единиц, а с единицы и нуля (табл. 1). Каждое следующее значение получается как сумма двух предыдущих. Будем выводить значение в первой строке, если оно меньше полученного на входе числа. Алгоритм удается реализовать, используя 18 команд, однако при этом число операций превышает требуемые 156 за счет необходимости расчета текущего элемента $i+1$-го шага с перезаписью текущего элемента i-го шага в ячейку предыдущего элемента $i+1$-го шага.

Таблица 1

Идея алгоритма вывода чисел Фибоначчи с 18 командами

Table 1

18 commands Fibonacci number output algorithm idea

Вход	19							
Номер шага	1	2	3	4	5	6	7	8
Текущий элемент	1	1	2	3	5	8	13	21
Вывод текущего элемента	+	+	+	+	+	+	+	–
Предыдущий элемент	0	1	1	2	3	5	8	13

Идея решения, удовлетворяющего испытанию скорости, заключается в том, чтобы сразу вывести обе первые единицы, а последующие значения выводить, чередуя элементы (табл. 2). При этом увеличивается количество команд кода, так как приходится считывать и выводить значения из разных ячеек, однако уменьшается число операций за счет того, что не нужно постоянно записывать текущий элемент последовательности в ячейку, из которой происходит вывод.

Таблица 2

Идея алгоритма вывода чисел Фибоначчи с 96 операциями

Table 2

96 steps Fibonacci number output algorithm idea

№ шага	Элемент 1	Элемент 2	Вход
0	1 (выход)	1 (выход)	19
1	2 (выход)	1	
2	2	3 (выход)	
3	5 (выход)	3	
4	5	8 (выход)	
5	13 (выход)	8	
6	13	21 > 19	

2. Сортировка трех: для каждых трех входных значений вывести их в порядке возрастания. Испытание объема: использовать не более 34 команд, испытание скорости: выполнить не более чем за 78 операций.

Авторское решение: два алгоритма, 34/126 и 63/74.

По нашему мнению, это задача с самыми сложными заданиями оптимизации в игре. Причем не потому, что их действительно сложно реализовать, а потому что сами оптимальные методы решения несколько искусственны, поэтому до них сложно додуматься без подсказки. Кроме того,

анализ контента центра сообщества Steam показал, что алгоритма, удовлетворяющего обоим заданиям оптимизации, пока не найдено.

Итак, рассматриваемая задача является частным случаем задачи сортировки. Три различных числа дают 6 возможных порядков расположения, для их упорядочения в наихудшем случае необходимо три сравнения (табл. 3).

Таблица 3

Количества сравнения для сортировки трех элементов

Table 3

Comparison quantities for sorting three elements

№	Порядок расположения	Сравнения для упорядочения
1	1 2 3	Два сравнения: – второй меньше третьего; – первый меньше второго (а значит, и третьего, последовательность упорядочена)
2	1 3 2	Два сравнения: – второй больше третьего – меняем их местами (получаем 1 2 3); – первый меньше второго (а значит, и третьего, последовательность упорядочена)
3	2 1 3	Три сравнения; – второй меньше третьего; – первый больше второго; – первый меньше третьего, значит, меняем местами 1 и 2 местами (получаем 1 2 3, последовательность упорядочена)
4	2 3 1	Три сравнения: – второй больше третьего – меняем их местами (получаем 2 1 3); – первый больше второго; – первый меньше третьего, значит, меняем местами 1 и 2 местами (получаем 1 2 3, последовательность упорядочена)
5	3 1 2	Три сравнения: – второй меньше третьего; – первый больше второго; – первый больше третьего, значит, меняем 1 и 2 местами (получаем 1 3 2), меняем 2 и 3 местами, получаем 1 2 3, последовательность упорядочена)
6	3 2 1	Три сравнения: – второй больше третьего – меняем их местами (получаем 3 1 2); – первый больше второго; – первый больше третьего, значит, меняем 1 и 2 местами (получаем 1 3 2), меняем 2 и 3 местами, получаем 1 2 3, последовательность упорядочена)

Естественная идея решения следующая. Считываем все три входных значения. На первом шаге сравниваем второе и третье значения, если второе больше, то меняем их местами. На втором шаге сравниваем первое и третье значения, если первое больше третьего, то меняем их местами. И на третьем шаге сравниваем первое и второе значения, если первое больше второго, то меняем их местами. В результате получаем упорядоченную последовательность, которую необходимо отправить на выход. Соответствующее авторское решение содержит 36 команд и выполняется за 120 операций, т. е. не удовлетворяет ни одному из испытаний оптимизации.

Модификация алгоритма для испытания объема разработана автором, исходя из следующих соображений. Поскольку для сортировки трех значений сравнений должно быть два или три, но при этом число команд превышает допустимый предел, необходимо выполнить два раза по два сравнения. Алгоритм приобретает следующий вид.

1. Считываем и сохраняем все три сортируемых значения.
2. Сравниваем второе и третье значение, если третье меньше второго, меняем их местами.

3. Сравниваем первое и второе значения:

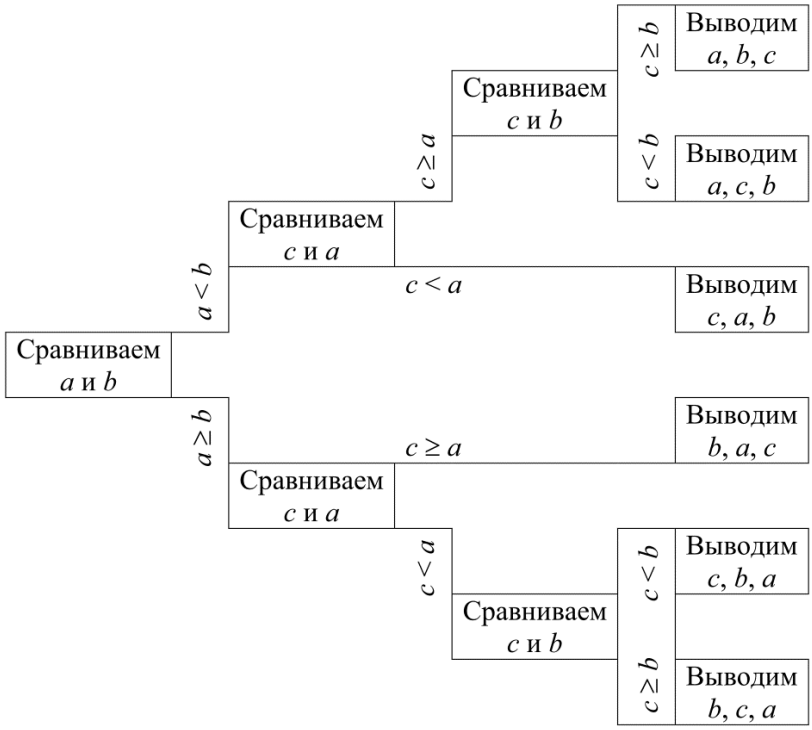
– если второе меньше первого, меняем их местами и переходим к пункту 2;

– если первое меньше второго, то последовательность упорядочена, переходим к пункту 4.

4. Выводим упорядоченную последовательность.

Данный алгоритм удается реализовать, используя 34 команды, однако необходимость выполнить четыре сравнения вместо трех приводит к увеличению числа операций со 120 до 126.

Алгоритм, обеспечивающий сортировку за заданное число операций, представляет собой анализ вариантов в табл. 3 и организацию вывода соответствующим образом. Лучшей его иллюстрацией является дерево решений (см. рисунок). Естественно, результирующий код получается громоздким (63 команды), а команда вывода фигурирует в нем 18 раз. Тем не менее решение на его основе позволяет отсортировать набор входных последовательностей за 74 операции при требуемых по заданию 78.



Анализ значений и организация вывода в задаче «Сортировка трех»
“Sorting three” problem values analysis and output organization

3. Упорядочиватель: для двух данных слов вывести первое в алфавитном порядке. Испытание объема: использовать не более 39 команд, испытание скорости: выполнить не более чем за 109 операций.

Авторское решение: один алгоритм, 36/108. В данной задаче имеет смысл охарактеризовать особенности входных данных, так как именно они определяют специфику решения. Алгоритм должен обработать одну пару английских слов, т. е. нет необходимости организовывать цикл для ввода данных. Слова могут быть как разной, так и одинаковой длины. Для обозначения окончания слова используется ноль.

Идея решения заключается в считывании обоих слов, организации счетчиков адресов букв для каждого из них и последовательного вычитания из текущей буквы первого слова соответствующей буквы второго слова. Пока разность букв нулевая, т. е. они совпадают, счетчики увеличиваются на 1. Как только получилась положительная (отрицательная) разность, выводим второе (первое) слово. Если при считывании буквы получили ноль, выводим короткое слово.

Алгоритм решения задачи следующий.

1. Считываем первое слово, записываем его последовательно в ячейки памяти, начиная с нулевой. Ноль определяет конец первого слова, увеличенный на 1 адрес соответствующей ячейки определяет начало второго слова.

2. Записываем второе слово в ячейки памяти, начиная с запомненного на предыдущем шаге адреса.

3. Запоминаем адреса начальных букв обоих слов.

4. Считываем текущий знак первого слова:

– если это ноль, то первое слово короче второго, отправляем его на выход, адрес первой буквы нулевой;

– если это не ноль, переходим к шагу 5.

5. Считываем текущий знак второго слова.

– если это ноль, то второе слово короче первого, отправляем его на выход, адрес первой буквы известен;

– если это не ноль, переходим к шагу 6.

6. Сравниваем буквы первого и второго слова (вычитаем из буквы первого слова второе, получаем разницу их позиций в латинском алфавите):

– если результат нулевой, то буквы одинаковые, увеличиваем на 1 адреса букв обоих слов и переходим к шагу 4;

– если результат положительный, то буква первого слова имеет больший номер, чем буква второго слова, т. е. по алфавиту первым будет второе слово, отправляем его на выход, адрес первой буквы известен;

– если результат отрицательный, то буква первого слова имеет меньший номер, чем буква второго слова, т. е. по алфавиту первым будет первое слово, отправляем его на выход, адрес первой буквы нулевой.

4. Подрыв чисел: каждое число на входе разбейте на отдельные цифры и выведите их. Испытание объема: использовать не более 30 команд, испытание скорости: выполнить не более чем за 165 операций.

Авторское решение: два алгоритма, 30/218 и 51/159.

На вход подаются трехзначные и двузначные числа, а также цифры. В двух ячейках памяти хранятся числа 10 и 100.

Формально алгоритм решения задачи весьма незамысловат.

1. Формируем нулевые счетчики количества сотен и десятков.

2. Последовательно уменьшаем входное значение на 100 и увеличиваем на единицу соответствующий счетчик до достижения двузначного числа. Если первое же вычитание дает отрицательный результат, то число двузначное, переходим к пункту 4.

3. Выводим счетчик сотен, если он отличен от нуля.

4. Последовательно уменьшаем двузначное значение на 10 и увеличиваем на единицу соответствующий счетчик до достижения однозначного числа.

5. Выводим счетчик десятков, если он не нулевой или если число было трехзначным.

6. Выводим оставшуюся цифру.

Данная задача является предпоследней задачей повышенной сложности в игре, и реализовать алгоритмы, удовлетворяющие ограничениям на число команд и операций, оказалось весьма непросто. Дополнительную сложность представляет необходимость определить, следует ли выводить нулевой счетчик десятков на шаге 5. В случае алгоритма, удовлетворяющего испытанию объема, данная проверка реализуется следующим образом: нулевой счетчик десятков выводится, если разность нулевого счетчика десятков и счетчика сотен отрицательная (т. е. счетчик сотен положительный).

Алгоритм, удовлетворяющий испытанию скорости, несколько сложнее. Он предполагает определение порядка числа: трехзначные, двузначные и однозначные. Однозначные числа не записываются в память, для них не формируются счетчики десятков и сотен. Для двузначных чисел формируется только счетчик десятков. Однако уменьшения числа операций за счет экономии на счетчиках оказалось недостаточно, поэтому приходится выполнять вычитание из текущего значения двух сотен/десятков за один цикл и после каждого вычитания проверять положительность результата. При этом существенно возрастает не только число команд алгоритма (до 51), но и скорость его работы; результирующий алгоритм выполняется за 159 операций при требуемых 165.

5. Фабрика простоты: для каждого числа на входе выведите его простые множители в порядке возрастания. Испытание объема: использовать не более 28 команд, испытание скорости: выполнить не более чем за 399 операций.

Авторское решение: два алгоритма, 24/452 и 46/360.

Алгоритм, соответствующий испытанию объема, имеет следующий вид.

1. Считываем входное значение, записываем его в память.
2. Проверяем, не равно ли текущее значение в памяти единице:
– если текущее значение равно единице, считываем следующее входное значение;
– если текущее значение больше единицы, переходим к шагу 3 алгоритма.
3. Формируем наименьший возможный множитель числа – двойку, записываем его в память.
4. Последовательно вычитаем из числа в памяти текущий множитель до достижения нуля или отрицательного результата:
– если результат отрицательный, данное число на данный множитель нацело не делится, увеличиваем множитель на 1 и переходим в начало шага 4;
– если результат нулевой, то число делится на множитель нацело, переходим к шагу 5.
5. Выводим текущий множитель.
6. Находим результат от деления данного числа на данный множитель:
– формируем счетчик;
– последовательно вычитаем текущий множитель из текущего числа и увеличиваем счетчик до достижения нулевого результата;
– полученное значение счетчика определяет результат деления;
– заменяем текущее число в памяти на значение счетчика;
– переходим к шагу 3 алгоритма.

Анализ входных значений позволил установить, что они не превосходят двадцати. Тогда идея алгоритма, соответствующего испытанию скорости, состоит в установлении, является ли данное значение из списка простых чисел от двух до девятнадцати делителем поданного на вход числа.

Алгоритм, соответствующий испытанию скорости, имеет следующий вид.

1. Формируем список простых чисел от двух до 19, записываем их в ячейки с адресами, начиная с нуля и возрастающими на единицу.
2. Считываем входное значение, записываем его в память.
3. Устанавливаем адрес текущего простого множителя нулевым (двойка).
4. Проверяем, не равно ли текущее значение в памяти единице:
– если текущее значение равно единице, считываем следующее входное значение;
– если текущее значение больше единицы, переходим к шагу 5 алгоритма.
5. Устанавливаем счетчик, показывающий целую часть от деления данного числа на данный простой множитель.
6. Последовательно вычитаем из числа в памяти текущий множитель до достижения нуля или отрицательного результата, увеличивая счетчик целой части:
– если результат отрицательный, данное число на данный множитель нацело не делится, увеличиваем адрес множителя на 1 и переходим к шагу 5;
– если результат нулевой, то число делится на множитель нацело, переходим к шагу 7.
7. Выводим текущий множитель.
8. Записываем значение счетчика на место числа, для которого ищутся простые множители.
9. Переходим к шагу 4 алгоритма.

6. Этаж сортировки: отсортируйте каждую входную последовательность в порядке возрастания и выведите результат. Испытание объема: использовать не более 34 команд, испытание скорости: выполнить не более чем за 714 операций.

Авторское решение: один алгоритм, 33/706.

Идея решения состоит в разбиении сортируемого массива на головную и хвостовую части, причем головная часть является уже отсортированной; в хвостовой части ищется минимальный элемент, меняется местами с первым элементом хвостовой части, после чего головная часть удлиняется, а хвостовая укорачивается на этот элемент. Специфика алгоритма, удовлетворяющего

обоим испытаниям, заключается в сортировке массива в порядке убывания и выводе его элементов, начиная с «хвоста». Он имеет следующий вид.

1. Считываем массив, подлежащий сортировке, в качестве индексов заменяемого, минимального и текущего элементов определяем индекс последнего элемента массива, запоминаем его.

2. Уменьшаем индекс текущего элемента на 1, сравниваем результат с нулем:

– если результат отрицательный, достигнут последний элемент массива, переходим к шагу 4;

– иначе переходим к шагу 3.

3. Сравниваем значение текущего и минимального элементов:

– если текущее значение меньше минимального, переопределяем индекс минимального элемента на текущий и переходим к шагу 2;

– иначе переходим к шагу 2.

4. Меняем местами заменяемый и минимальный элементы, выводим найденный минимальный элемент.

5. Уменьшаем счетчик заменяемого, минимального и текущего элементов на 1, переходим к шагу 2 алгоритма.

Заключение

В статье рассмотрены авторские оптимизированные алгоритмы шести наиболее сложных задач игры-головоломки Human Resource Machine. Оптимизация в соответствии с правилами игры выполнялась по количеству команд кода и операций, производимых в процессе его выполнения. Часть оптимальных решений удовлетворяет обоим требованиям, т. е. обеспечивает заданное число операций при заданном количестве команд, остальные потребовали разработки двух различных алгоритмов. Отметим, что в игровом сообществе Steam можно найти и другие варианты решения тех же задач, возможно, с еще меньшим количеством команд/операций, однако автор придерживается мнения, что собственное решение головоломки приносит гораздо больше удовлетворения, чем чужое. Кроме того, установлено, что даже для самых сложных задач в игре реально отыскать оптимальные решения, не прибегая к копированию чужого кода. Таким образом, даже простые задачи по программированию могут иметь сложные решения, требующие нестандартных подходов и инновационных идей, что не только приносит массу интеллектуального удовольствия и удовлетворения от полученных работающих решений, но и позволяет использовать их, например, в качестве олимпиадных заданий.

Список литературы

1. Месеняшина Л.А., Селютин А.А. Компьютерная игра: аттракцион? Искусство? // Знак: проблемное поле медиаобразования. 2011. № 1 (7). URL: https://cyberleninka.ru/article/n/kompyuternaya-igra-attraktsion-iskusstvo (дата обращения: 23.08.2022).

2. Шереметьева М.А. Статус компьютерных игр в современной культуре // Ярославский педагогический вестник. 2019. № 6. URL: https://cyberleninka.ru/article/n/status-kompyuternyh-igr-v-sovremennoy-kulture (дата обращения: 23.08.2022).

3. Sălceanu C. The Influence of Computer Games on Children's Development. Exploratory Study on the Attitudes of Parents // Procedia – Social and Behavioral Sciences. 2014. Vol. 149. P. 837–841. DOI: 10.1016/j.sbspro.2014.08.323. URL: https://www.sciencedirect.com/science/article/pii/ S1877042814050368 (дата обращения: 23.08.2022).

4. P82 Assessment of Correlation Between EEG Frequency Subbands and HRV in Playing Puzzle Video Game / S. Gündoğdu, Ö.H. Çolak, E. Apaydın Doğan, E. Gülbetekin, Ö. Polat // Clinical Neurophysiology. 2020. Vol. 131, iss. 4. P. e217-e218. DOI: 10.1016/j.clinph.2019.12.080. URL: https://www.sciencedirect.com/science/article/pii/S1388245719314464 (дата обращения: 23.08.2022).

5. Kim M. Does playing a video game really result in improvements in psychological well-being in the era of COVID-19? // Journal of Retailing and Consumer Services. 2021. Vol. 61. DOI: 10.1016/j.jretconser.2021.102577. URL: https://www.sciencedirect.com/science/article/pii/S0969698921001430 (дата обращения: 23.08.2022).

6. Zhao W., Shute V.J. Can playing a video game foster computational thinking skills? // Computers & Education. 2019. Vol. 141. DOI: 10.1016/j.compedu.2019.103633. URL: https://www.sciencedirect.com/science/article/pii/S0360131519301861 (дата обращения: 23.08.2022).

7. Oei A.C., Patterson M.D. Playing a puzzle video game with changing requirements improves executive functions // Computers in Human Behavior. 2014. Vol. 37. P. 216–228. DOI: 10.1016/j.chb.2014.04.046. URL: https://www.sciencedirect.com/science/article/abs/pii/ S0747563214002672?via%3Dihub (дата обращения: 23.08.2022).

8. Шмелев Б.А. Лингводидактический потенциал обучающих компьютерных игр // Вестник ТГУ. 2021. № 192. DOI: 10.20310/1810-0201-2021-26-192-58-69. URL: https://cyberleninka.ru/article/n/lingvodidakticheskiy-potentsial-obuchayuschih-kompyuternyh-igr (дата обращения: 20.08.2022).

9. How instructional context can impact learning with educational technology: Lessons from a study with a digital learning game / B.M. McLaren, J.E. Richey, H. Nguyen, X. Hou // Computers & Education. 2022. Vol. 178. DOI: 10.1016/j.compedu.2021.104366. URL: https://www.sciencedirect.com/science/article/pii/S0360131521002438 (дата обращения: 23.08.2022).

10. Maji A.K., Jana S., Pal R.K. An Algorithm for Generating only Desired Permutations for Solving Sudoku Puzzle // Procedia Technology. 2013. Vol. 10. P. 392–399. DOI: 10.1016/j.protcy.2013.12.375. URL: https://www.sciencedirect.com/science/article/pii/S2212017313005379 (дата обращения: 23.08.2022).

11. Anu Th., Sangeetha S. Towards a Semantic Approach for Candidate Answer Generation in Solving Crossword Puzzles // Procedia Computer Science. 2020. Vol. 171. P. 2310–2315. DOI: 10.1016/j.protcy.2013.12.375. URL: https://www.sciencedirect.com/science/article/pii/S1877050920312424 (дата обращения: 23.08.2022).

12. Jiang M., Tejada P.J., Wang H. Quell // Theoretical Computer Science. 2015. Vol. 593. P. 70–78. DOI: 10.1016/j.tcs.2015.05.044. URL: https://www.sciencedirect.com/science/article/pii/S0304397515004983 (дата обращения: 23.08.2022).

13. Васильев А.П., Абрамов А.Х. Искусственный интеллект на основе нейронных сетей // Academy. 2018. № 5 (32). URL: https://cyberleninka.ru/article/n/iskusstvennyy-intellekt-na-osnove-neyronnyh-setey (дата обращения: 20.08.2022).

14. Crippa M., Lanzi P.L., Marocchi F. An analysis of Single-Player Monte Carlo Tree Search performance in Sokoban // Expert Systems with Applications. 2022. Vol. 192. DOI: 10.1016/j.eswa.2021.116224. URL: https://www.sciencedirect.com/science/article/abs/pii/S0957417421015372?via%3Dihub (дата обращения: 23.08.2022).

15. A solver of single-agent stochastic puzzle: A case study with Minesweeper / Ch. Liu, Sh. Huang, G. Naying et al. // Knowledge-Based Systems. 2022. Vol. 246. DOI: 10.1016/j.knosys.2022.108630. URL: https://www.sciencedirect.com/science/article/pii/S0950705122002842 (дата обращения: 23.08.2022).

References

1. Mesenyashina L.A., Selyutin A.A. Computer Game: Attraction? Art? *Znak: problemnoe pole mediaobrazovanija*. 2011;(1). Available at: https://cyberleninka.ru/article/n/kompyuternaya-igra-attraktsion-iskusstvo (accessed 23.08.2022). (In Russ.)

2. Sheremet'yeva M.A. [Status of Computer Games in Contemporary Culture]. *Yaroslavl Pedagogical Bulletin*. 2019;(6). Available at: https://cyberleninka.ru/article/n/status-kompyuternyh-igr-v-sovremennoy-kulture (accessed 23.08.2022). (In Russ.)

3. Sălceanu C. The Influence of Computer Games on Children's Development. Exploratory Study on the Attitudes of Parents. *Procedia – Social and Behavioral Sciences*. 2014;149:837–841. DOI: 10.1016/j.sbspro.2014.08.323. Available at: https://www.sciencedirect.com/science/article/pii/S1877042814050368 (accessed 23.08.2022).

4. Gündoğdu S., Çolak Ö.H., Apaydın Doğan E., Gülbetekin E., Polat Ö. P82 Assessment of Correlation Between EEG Frequency Subbands and HRV in Playing Puzzle Video Game. *Clinical Neurophysiology*. 2020; 131(4):217–218. DOI: 10.1016/j.clinph.2019.12.080. Available at: https://www.sciencedirect.com/science/article/pii/S1388245719314464 (accessed 23.08.2022).

5. Kim M. Does Playing a Video Game Really Result in Improvements in Psychological Well-Being in the Era of COVID-19? *Journal of Retailing and Consumer Services*. 2021;61. DOI: 10.1016/j.jretconser.2021.102577. Available at: https://www.sciencedirect.com/science/article/pii/S0969698921001430 (accessed 23.08.2022).

6. Zhao W., Shute V.J. Can playing a video game foster computational thinking skills? *Computers & Education*. 2019;141. DOI: 10.1016/j.compedu.2019.103633. Available at: https://www.sciencedirect.com/science/article/abs/pii/S0360131519301861?via%3Dihub (accessed 23.08.2022).

7. Oei A.C., Patterson M.D. Playing a puzzle video game with changing requirements improves executive functions. *Computers in Human Behavior*. 2014;37:216–228. DOI: 10.1016/j.chb.2014.04.046. Available at: https://www.sciencedirect.com/science/article/abs/pii/S0747563214002672?via%3Dihub (accessed 23.08.2022).

8. Shmelev B.A. Linguodidactic Potential of Educational Computer Games. *Tambov University Review: Series Humanities*. 2021;192. (In Russ.) DOI: 10.20310/1810-0201-2021-26-192-58-69. Available at: https://cyberleninka.ru/article/n/lingvodidakticheskiy-potentsial-obuchayuschih-kompyuternyh-igr (accessed 20.08.2022).

9. McLaren B.M., Richey J.E., Nguyen H., Hou X. How instructional context can impact learning with educational technology: Lessons from a study with a digital learning game. *Computers & Education*. 2022;178. DOI: 10.1016/j.compedu.2021.104366. Available at: https://www.sciencedirect.com/science/article/pii/S0360131521002438 (accessed 23.08.2022).

10. Maji A.K., Jana S., Pal R.K. An Algorithm for Generating only Desired Permutations for Solving Sudoku Puzzle. *Procedia Technology*. 2013; 10:392–399. DOI: 10.1016/j.protcy.2013.12.375. Available at: https://www.sciencedirect.com/science/article/pii/S2212017313005379 (accessed 23.08.2022).

11. Anu Th., Sangeetha S. Towards a Semantic Approach for Candidate Answer Generation in Solving Crossword Puzzles. *Procedia Computer Science*. 2020;171:2310–2315. DOI: 10.1016/j.protcy.2013.12.375. Available at: https://www.sciencedirect.com/science/article/pii/S1877050920312424 (accessed 23.08.2022).

12. Jiang M., Tejada P.J., Wang H. Quell. *Theoretical Computer Science*. 2015;593:70–78. DOI: 10.1016/j.tcs.2015.05.044. Available at: https://www.sciencedirect.com/science/article/pii/S0304397515004983 (accessed 23.08.2022).

13. Vasil'yev A.P., Abramov A.H. [Artificial Intelligence Based on Neural Networks]. *Academy*. 2018;5(32). (In Russ.) Available at: https://cyberleninka.ru/article/n/iskusstvennyy-intellekt-na-osnove-neyronnyh-setey (accessed 20.08.2022).

14. Crippa M., Lanzi P.L., Marocchi F. An analysis of Single-Player Monte Carlo Tree Search performance in Sokoban. *Expert Systems with Applications*. 2022;192. DOI: 10.1016/j.eswa.2021.116224. Available at: https://www.sciencedirect.com/science/article/abs/pii/S0957417421015372?via%3Dihub (accessed 23.08.2022).

15. Liu Ch., Huang Sh., Naying G., Khalid M.N.A., Iida H. A solver of single-agent stochastic puzzle: A case study with Minesweeper. *Knowledge-Based Systems*. 2022;246. DOI: 10.1016/j.knosys.2022.108630. Available at: https://www.sciencedirect.com/science/article/pii/S0950705122002842 (accessed 23.08.2022).

Информация об авторах

Копотева Анна Владимировна, канд. техн. наук, доц. кафедры общенаучных дисциплин, Пермский национальный исследовательский политехнический университет, Березниковский филиал, Березники, Россия; kopoteva_av@mail.ru.

Кабиольский Евгений Алексеевич, аспирант кафедры информационно-аналитического обеспечения управления в социальных и экономических системах, начальник управления информатизации, Южно-Уральский государственный университет, Челябинск, Россия; evgkab@susu.ru.

Information about the authors

Anna V. Kopoteva, Cand. Sci. (Eng.), Ass. Prof. of the Department of General Scientific Disciplines, Perm National Research Polytechnic University, Berezniki Branch, Berezniki, Russia; kopoteva_av@mail.ru.

Evgeny A. Kabiolsky, Postgraduate Student of the Department of Information and Analytical Support of Management in Social and Economic Systems, Head of Informatization Department, South Ural State University, Chelyabinsk, Russia; evgkab@susu.ru.

Статья поступила в редакцию 20.09.2022
The article was submitted 20.09.2022