



DOI: 10.32517/2221-1993-2024-23-1-54-59

А. А. Никандров,

Российский государственный педагогический университет им. А. И. Герцена, Санкт-Петербург, Россия

К. Р. Пиотровская

Российский государственный педагогический университет им. А. И. Герцена, Санкт-Петербург, Россия;

Санкт-Петербургский государственный университет, Санкт-Петербург, Россия

ЭЛЕКТИВНЫЙ КУРС ПО МАШИННОМУ ОБУЧЕНИЮ В СТАРШЕЙ ШКОЛЕ: ЗАДАЧИ СПОРТИВНОГО ПРОГРАММИРОВАНИЯ

Аннотация

В статье обсуждается перспектива преподавания элементов машинного обучения в старшей школе. Представлен обзор зарубежного опыта по изучению основ машинного обучения в школе. Предлагается система задач для элективного курса по машинному обучению для учащихся старших классов российских общеобразовательных школ. Осуществлен разбор пяти типовых задач спортивного программирования, которые являются авторской разработкой, при этом каждая задача имеет свой рейтинг по аналогии с предлагаемыми на известных ресурсах олимпиадного движения по информатике: CodeForces, LeetCode, AtCoder. Электив предназначен для школьников, которые уже имеют базовые знания в области программирования на языке Python 3, и состоит из трех модулей, которые ориентированы на различный уровень владения этим языком программирования. Учащийся имеет возможность повышать или снижать уровень прохождения курса. Элективный курс может быть реализован в системе Stepik. Обучение может осуществляться как в дистанционной форме, так и в форме смешанного обучения. Проверка программного кода заданий происходит автоматически. Рассмотрение теории и разбор задач осуществляются на интерактивной вычислительной платформе Jupyter Notebook.

Ключевые слова: машинное обучение, элективный курс, автоматическая проверка программного кода, старшая школа, олимпиадное программирование, спортивное программирование, дистанционное обучение.

Контактная информация

Никандров Алексей Андреевич, аспирант кафедры цифрового образования, Институт информационных технологий и технологического образования, Российский государственный педагогический университет им. А. И. Герцена, Санкт-Петербург, Россия; *адрес:* 191186, Россия, г. Санкт-Петербург, набережная реки Мойки, д. 48; *e-mail:* nik190397@mail.ru

Пиотровская Ксения Раймондовна, доктор пед. наук, канд. тех. наук, доцент; профессор кафедры методики обучения математике и информатике, факультет математики, Российский государственный педагогический университет им. А. И. Герцена, Санкт-Петербург, Россия; *адрес:* 191186, Россия, г. Санкт-Петербург, набережная реки Мойки, д. 48;

профессор кафедры математической лингвистики, филологический факультет, Санкт-Петербургский государственный университет, Санкт-Петербург, Россия; *адрес:* 199034, Россия, г. Санкт-Петербург, Университетская наб., д. 7/9; *e-mail:* krp62@mail.ru

A. A. Nikandrov

Herzen State Pedagogical University of Russia, St. Petersburg, Russia

A. A. Nikandrov, X. R. Piotrowska

Herzen State Pedagogical University of Russia, St. Petersburg, Russia;

St. Petersburg State University, St. Petersburg, Russia

ELECTIVE COURSE ON MACHINE LEARNING IN HIGH SCHOOL: PROBLEMS OF SPORTS PROGRAMMING

Abstract

The article discusses the prospect of teaching elements of machine learning in high school. An overview of foreign experience in studying the fundamentals of machine learning at school is presented. A system of tasks is proposed for an elective course on machine learning for high school students in Russian secondary schools. An analysis of five typical tasks of competitive programming, which are the author's development, was carried out, and each task has its own rating, similar to those offered on the well-known resources of the informatics Olympiad movement: CodeForces, LeetCode, AtCoder. The elective is intended for schoolchildren who already have basic knowledge of programming in Python 3, and consists of three modules that are aimed at different levels of proficiency in this programming language. The student has the opportunity to increase or decrease the level of completion of the course. The elective course can be implemented in the Stepik system. Training can be carried out either in distance learning or in the form of blended learning. The program code of tasks is checked automatically. Consideration of theory and analysis of tasks are carried out on the interactive computing platform Jupyter Notebook.

Keywords: machine learning, elective course, automatic code checking, high school, competitive programming, sports programming, distance learning.

1. Введение

В последнее время интерес общественности к искусственному интеллекту невероятно возрос. Машинное обучение, важнейшее подмножество искусственного интеллекта, постепенно превращается в новый инструмент, революционизирующий практику поиска знаний [11]. Преимущества, связанные с ознакомлением школьников со сложностями и внутренней работой машинного обучения, вызывают повышенный интерес среди заинтересованных сторон в сфере образования [1, 14]. Образование в области машинного обучения считается актуальным для будущих поколений, которым предположительно предстоит жить в эпоху этого обучения. Поскольку данный инструмент постепенно превращается в обычное явление в повседневной жизни людей, раннее знакомство с фундаментальными процессами машинного обучения может облегчить понимание детьми окружающего мира [8]. Согласно мнению западных исследователей (см., например, [11]), понимание того, как можно обучить модель, имеет решающее значение для разработки детьми полезных ментальных моделей для изучения искусственного интеллекта и интеллектуальных устройств, с которыми они сейчас часто взаимодействуют. Знакомство с основами машинного обучения должно способствовать повышению мотивации к исследовательской деятельности в области искусственного интеллекта следующих поколений разработчиков программного обеспечения [15].

В высших учебных заведениях широко распространено обучение основам концепций и методов [9, 15], а в последнее время за рубежом предпринимались попытки ввести преподавание новой концепции и в школах. В ряде работ рассмотрена явная демонстрация принципов машинного обучения (см., например, [2, 5, 12]). В частности, разработана система Any-Cubes, прототип игрушки, с помощью которой ученики могут интуитивно и в игровой форме проводить исследования по машинному обучению [13]. Предложена система Zhorai — средство общения для детей, позволяющее изучать концепции машинного обучения [11].

Продолжает расти число исследований, посвященных разработке учебных программ, материалам, инструментам или платформам для машинного обучения в школах разных стран (см., например, [1, 4, 16]). Но проблема в том, что начатые инициативы и проекты пока разрозненные, что затрудняет формирование обзора существующих работ по машинному обучению в сфере образования.

2. Материалы и методы элективного курса

Предлагаемый в данной статье набор задач спортивного программирования (далее — задачи) предназначен для элективного курса по машинному обучению для старшеклассников. Задачи не являются явными с точки зрения машинного обучения, но они закладывают фундамент для дальнейшего развития школьника в указанной области, поскольку подразумевают алгоритмический способ мышления. Их нельзя решить без четкого плана, что соответствует требованиям методик

машинного обучения, где обычно необходимо отчетливо предвидеть будущий результат.

Элективный курс предлагается для учащихся, которые уже имеют базовые знания в области программирования на языке Python 3. Курс состоит из трех модулей, ориентированных на различный уровень владения языком Python. В курсе присутствуют черты адаптивности — обучающийся имеет возможность повышать или понижать уровень прохождения курса.

Электив реализуется как в дистанционной форме, так и в форме смешанного обучения. Рассмотрение теории и разбор задач осуществляются на интерактивной вычислительной платформе **Jupyter Notebook**, которая позволяет поддерживать рабочий процесс обучения: интерактивные исследования, публикации подробных записей вычислений.

В последнее десятилетие успехом пользуется российская разработка **Stepik** — условно открытая система, где любой желающий может создать и разместить собственный онлайн-курс, укладываясь в рамки пользовательских правил. Систему отличает возможность автоматического тестирования решений задач по программированию. Поэтому именно система Stepik была выбрана нами для организации онлайн-курса с автоматической проверкой решений и сбором статистики [3].

Практически все (за исключением нескольких) задачи электива являются авторской разработкой, при этом каждая имеет свой рейтинг по аналогии с предлагаемыми на известных отечественных и международных ресурсах олимпиадного движения по информатике: CodeForces [7], LeetCode [10], AtCoder [6].

Предлагаемые учащимся задания первого модуля находятся в диапазоне 1000–1200, второго модуля — 1300–1400, третьего — 1500–1600 по шкале оценивания ресурса CodeForces.

3. Примеры разбора задач

Разберем решение нескольких задач, которые предлагаются в элективном курсе по машинному обучению. В описании каждой задачи дана ссылка на материалы электронного курса, организованного на платформе Stepik.

Задача 1.

Задача на сортировку:

<https://stepik.org/lesson/688258/step/1?unit=687627>

Условие.

Пусть существует число k . Если массив имеет минимум k чисел, то определим его показатель как сумму k наибольших чисел этого массива.

В качестве примера рассмотрим $b = [2, 3, 5, 5, 4]$ и $k = 4$. Тогда четырьмя наибольшими элементами b являются 3, 5, 5 и 4. В этом случае показатель b равен $3 + 5 + 5 + 4 = 17$.

Даны массив $b_1, b_2 \dots b_n$, вышеописанное число k и натуральное число s . Необходимо разбить массив b ровно на s подмассивов (подмассив — непрерывная подпоследовательность b) следующим образом:

1) каждый элемент b содержится ровно в одном подмассиве;

- 2) каждый подмассив имеет минимум k чисел;
- 3) сумма показателей всех s подмассивов максимально возможная.

Входные данные.

В первой строке даны числа n, k, s ($1 \leq n \leq 5 \cdot 10^4$, $1 \leq k, 2 \leq s, k \cdot s \leq n$), где n — количество чисел в массиве b , k — описанное выше число, s — число, отвечающее на вопрос, на сколько подмассивов нужно провести разбиение.

Во второй строке даны числа $b_1, b_2, \dots, b_n$ ($-10^9 \leq b_i \leq 10^9$).

Выходные данные.

В единственной строке выведите максимально возможную сумму показателей подмассивов.

Общая идея решения задачи.

По условию задачи необходимо разбить исходный массив b на s подмассивов, у каждого из которых длина не менее k . Также гарантируется, что количество элементов $k \cdot s$ не меньше длины массива b . Далее необходимо максимизировать суммы s подмассивов. Для этого следует отсортировать массив b и взять наибольшие $k \cdot s$ чисел, чтобы их можно было разделить произвольным образом между s подмассивами, поскольку значение итоговой суммы не изменится.

Протокол ввода/вывода и код на Python 3 представлены на рисунке 1.

Sample Input 1:	Решение на Python 3:
11 3 3	<code>n, k, s = [</code>
3 5 6 3 8 7 2 8 8 4 1	<code>int(num)</code>
	<code>for num in input().split()</code>
Sample Output 1:	<code>]</code>
52	<code>a = [</code>
	<code>int(num)</code>
	<code>for num in input().split()</code>
Sample Input 2:	<code>]</code>
1 1 1	<code>b = a.copy()</code>
100	<code>b.sort(reverse=True)</code>
Sample Output 2:	
100	

Рис. 1. Протокол ввода/вывода и код задачи 1

Задача 2.

Задача на жадный алгоритм:

<https://stepik.org/lesson/688258/step/2?unit=687627>

Условие.

Имеется m левых и m правых варезек. Каждая варезка обладает своим цветом, заданным строчной буквой английского алфавита или знаком вопроса, если цвет не определен.

Каково максимальное количество пар варезек, таких, что одна варезка — левая, вторая — правая и при этом их цвета либо совпадают (например, ('z', 'z') или ('?', '?')), либо один цвет задан строчной латинской буквой, а другой — знаком вопроса (например, ('z', '?') или ('?', 'z'))?

Обратите внимание: требуется найти максимальное количество варезек без их возвращения. Например, если уже нашли подходящую пару, где левая — с индексом i , а правая — с индексом j , то больше для левых варезек

не нужно рассматривать варианты с индексом i , а для правых — с индексом j .

Входные данные.

В первой строке дано число m — количество пар варезек ($1 \leq m \leq 5 \cdot 10^4$).

Во второй/третьей строке даны цвета левых/правых варезек, заданные строчными латинскими буквами или знаком вопроса.

Выходные данные.

В единственной строке выведите максимально возможное количество пар варезек, удовлетворяющих условию задачи.

Общая идея решения задачи.

Предлагаемое решение — классическое с точки зрения работы с жадными алгоритмами. Посчитаем количество символов в каждой из строк. Далее те символы, которые совпадают в каждой из строк (кроме знаков '?'), подпадают под первое условие. Вторым действием считаем пары, где из одной любой строки берем знак вопроса, а из другой — любой символ, кроме '?'. Последним шагом остается подсчитать количество пар, состоящих только из знака вопроса.

Протокол ввода/вывода и код на Python 3 представлены на рисунке 2.

	Решение на Python 3:
	<code>from collections import defaultdict</code>
	<code>def cycle(s, w, flag):</code>
	<code>while s and w:</code>
	<code> pairs.append(</code>
	<code> [s.pop(),</code>
	<code> w.pop()]</code>
	<code> if flag</code>
	<code> else</code>
	<code> [w.pop(),</code>
	<code> s.pop()]</code>
	<code>)</code>
	<code>def func(s, w, flag):</code>
	<code> for c in s:</code>
	<code> if c != '?' and c in w:</code>
	<code> cycle(s[c], w[c], flag)</code>
	<code> if not w[c]:</code>
	<code> del w[c]</code>
	<code>def func2(w, s, flag):</code>
	<code> for c in s:</code>
	<code> if c != '?':</code>
	<code> cycle(w['?'], s[c], flag)</code>
Sample Input 1:	
9	
?stepikk?	
lifeis?kk	
Sample Output 1:	
8	
Sample Input 2:	
7	
abcdcba	
e??????	
Sample Output 2:	
6	
	<code>pairs = []</code>
	<code>func(1, r, 1)</code>
	<code>func(r, 1, 0)</code>
	<code>func2(1, r, 1)</code>
	<code>func2(r, 1, 0)</code>
	<code>while 1['?'] and r['?']:</code>
	<code> pairs.append([1['?'].pop(),</code>
	<code> r['?'].pop()])</code>
	<code>print(len(pairs))</code>

Рис. 2. Протокол ввода/вывода и код задачи 2

Задача 3.

Задача на жадный алгоритм:

<https://stepik.org/lesson/693003/step/1?unit=692613>

Условие.

Дана строка *string*, состоящая из 0, 1 и 2. Вы должны заменить минимальное количество символов в этой строке так, чтобы она стала сбалансированной, т. е. количество цифр 0, 1 и 2 должно быть одинаковым. Замены можно делать лишь на 0, 1 или 2.

При этом из всех строк, удовлетворяющих требованиям, описанным выше, необходимо получить лексикографически наименьшую.

Входные данные.

В первой строке дано число n — длина строки *string* ($3 \leq n \leq 5 \cdot 10^4 + 1$). Гарантируется, что n делится на 3.

Во второй строке дана строка *string*, состоящая из 0, 1, 2.

Выходные данные.

Выведите ответ в единственной строке.

Общая идея решения задачи.

В результате работы алгоритма нули должны располагаться как можно левее, после чего аналогичное условие касается единиц, и двойки занимают все оставшиеся места, при этом требуется минимизировать количество замен. Замены осуществляются лишь в том случае, когда количество нулей, единиц или двоек строго больше $each = n // 3$. Проход по строке совершается три раза в предложенном коде, а именно: слева направо для

Решение на Python 3:

```
from collections import Counter

n = int(input())
each = n // 3
s = [ch for ch in input().strip()]
c = Counter(s)

for ind in range(n):
    if c[s[ind]] > each > c['0'] \
       and s[ind] != '0':
        c[s[ind]] -= 1
        s[ind] = '0'
        c['0'] += 1

for ind in range(-1, -len(s) - 1, -1):
    if c[s[ind]] > each > c['2'] \
       and s[ind] != '2':
        c[s[ind]] -= 1
        s[ind] = '2'
        c['2'] += 1
    elif c['0'] > each > c['1'] \
         and s[ind] == '0':
        c['0'] -= 1
        s[ind] = '1'
        c['1'] += 1

for ind in range(n):
    if c['2'] > each > c['1'] \
       and s[ind] == '2':
        c['2'] -= 1
        s[ind] = '1'
        c['1'] += 1

print("".join(s))
```

Sample Input 1:

3

210

Sample Output 1:

210

Sample Input 2:

6

111222

Sample Output 2:

011022

Sample Input 3:

6

222222

Sample Output 3:

001122

расстановки нулей, затем справа налево — для двоек и, наконец, слева направо — для единиц. Стоит отметить, что необязательно соблюдать предложенный порядок, а также можно сократить количество циклов, например, совместив первый и последний циклы, которые совершают корректировки количества нулей и единиц.

Протокол ввода/вывода и код на Python 3 представлены на рисунке 3.

Задача 4.

Задача на обработку запросов:

<https://stepik.org/lesson/693003/step/2?unit=692613>

Условие.

Существует два типа запросов:

1. Получение прямоугольной банкноты размером $a \times b$, где a — длина, b — высота. Банкноту можно переворачивать, в этом случае a становится высотой, а b — длиной.
2. Получение прямоугольной барсетки размером $h \times w$, где h — высота, а w — длина.

Когда приходит запрос второго типа, если все предыдущие банкноты могут поместиться в барсетку, нужно вывести слово «Yes», иначе вывести слово «No».

Банкнота помещается в барсетку, если $(a \leq h \text{ и } b \leq w)$ или $(b \leq h \text{ и } a \leq w)$. Банкноты в барсетке могут накладываться друг на друга, а сама барсетка вмещает бесконечное количество банкнот.

Входные данные.

В первой строке дано число n — количество запросов ($2 \leq n \leq 5 \cdot 10^4$).

Далее поступают n строк, каждая из которых содержит запрос первого или второго типа:

1. $+ ab$ ($1 \leq a, b \leq 10^4$) — банкнота размером $a \times b$.
2. $? hw$ ($1 \leq h, w \leq 10^4$) — барсетка размером $h \times w$, для которой надо проверить, что все предыдущие банкноты могут в нее уместиться.

Выходные данные.

Для каждого запроса второго типа выведите «Yes», если все банкноты могут уместиться в барсетке, иначе «No».

Решение на Python 3:

```
cur_x, cur_y = 0, 0
for _ in range(int(input())):
    operation, *query = [
        char
        for char in input().split()
    ]
    query = [
        int(num)
        for num in query
    ]
    query.sort()
    if operation == '+':
        cur_x = max(cur_x, query[0])
        cur_y = max(cur_y, query[1])
    else:
        print(
            "YES"
            if cur_x <= query[0]
            and cur_y <= query[1]
            else "NO"
        )
```

Sample Input:

7

+ 2 2

+ 1 2

? 3 3

+ 4 3

? 2 4

+ 4 4

? 4 5

Sample Output:

YES

NO

YES

Рис. 3. Протокол ввода/вывода и код задачи 3

Рис. 4. Протокол ввода/вывода и код задачи 4

Общая идея решения задачи.

Будем придерживаться одного соглашения: расположение короткой стороны банкноты вертикально, а длинной стороны — горизонтально. Аналогично поступаем с переворачиванием барсетки. Таким образом, удастся достичь единого расположения не только для банкнот, но и для барсеток. Далее остается лишь пересчитывать максимум при получении параметров новой банкноты и сравнить его с параметрами входящей барсетки. Благодаря поддержанию показателя максимума среди банкнот получается сохранять линейную сложность алгоритма.

Протокол ввода/вывода и код на Python 3 представлены на рисунке 4.

Задача 5.

Задача на сортировку:

<https://stepik.org/lesson/693003/step/3?unit=692613>

Условие.

Дано m отрезков, которые заданы своими концами $left_i$ и $right_i$, где $1 \leq i \leq m$. Необходимо все отрезки разделить на два непустых множества так, чтобы любой отрезок из одного множества не пересекался с любым отрезком другого множества.

Входные данные.

В первой строке дано число m — количество отрезков ($2 \leq m \leq 5 \cdot 10^4$).

Далее поступают m строк, каждая из которых содержит два числа $left_i$ и $right_i$ ($1 \leq left_i \leq right_i \leq 10^6$), т. е. границы i -го отрезка.

Выходные данные.

Вывести «Yes», если возможно выполнить условие задачи, иначе «No».

Решение на Python 3:	
Sample Input 1:	<pre> n = int(input()) threes = [] for order_num in range(n): a, b = [int(num) for num in input().split()] threes.append([a, b, order_num]) threes.sort(reverse=True) g = {} fg = threes.pop() g[fg[2]] = 1 while threes: sg = threes.pop() if sg[0] <= fg[1]: g[sg[2]] = 1 fg[1] = max(fg[1], sg[1]) else: g[sg[2]] = 2 while threes: sg = threes.pop() g[sg[2]] = 2 print("YES" if any(2 for k in g if g[k] == 2) else "NO") </pre>
3	
2 5	
4 5	
5 6	
Sample Output 1:	NO
Sample Input 2:	
4	
3 7	
2 9	
10 15	
10 10	
Sample Output 2:	YES

Рис. 5. Протокол ввода/вывода и код задачи 5

Общая идея решения задачи.

Для данной задачи существуют как минимум два решения, использующих сортировку по левой и по правой границе. Мы станем придерживаться сортировки по левой границе отрезков. Перебираем отрезок за отрезком, проходя слева направо. Обращаем внимание на правую границу каждого отрезка. Если она пересекается с одним из последующих отрезков, то переходим к нему и проделываем аналогичное действие. В том случае, когда нет пересечения, возможны два исхода, а именно: отсутствует следующий отрезок, тогда нет возможности разделить на два непересекающихся множества; существует следующий отрезок, но его начало не пересекается с концом текущего.

В случае второго исхода текущий и все предыдущие отрезки относятся к первому множеству, а оставшиеся — ко второму, это означает, что существует возможность деления на два непересекающихся множества.

Протокол ввода/вывода и код на Python 3 представлены на рисунке 5.

4. Обсуждение и выводы

Применение технологий машинного обучения позволяет изменить парадигму решения задач с аналитического подхода на подход, основанный на данных, с помощью компьютерных программ, которые обучают модели на основе имеющихся сведений и прогнозируют вероятные результаты.

Все представленные выше задачи были опробованы на группе учащихся в 17 человек. В таблице приведена статистика решения обсуждаемых задач, выработанная платформой Stepik. Из полученных данных видно, что ни один учащийся не справился со всеми заданиями. Это можно объяснить тем, что существовала возможность сменить траекторию обучения, а следовательно, студенты шли по пути меньшего сопротивления и занимались более простыми заданиями.

Таблица

Статистика успешности при решении задач

№ задачи	Количество студентов, представивших правильное решение	Количество правильных решений, % от общего числа попыток
1	8	67
2	5	93
3	5	80
4	1	100
5	1	100

В ближайшей перспективе запланирована корректировка, связанная с усложнением простых задач с целью повышения активности на вышеописанных задачах. Также будут добавлены новые типы задач и пополнены уже существующие.

Список источников

1. Босова Л. Л. О новых подходах к изучению школьной информатики в условиях цифровой трансформации общества // Информатика в школе. 2022. Т. 21. № 4. С. 5–14. EDN: DKRLZV. DOI: 10.32517/2221-1993-2022-21-4-5-14.
2. Левченко И. В., Абушкин Д. Б., Карташова Л. И. Модуль «Машинное обучение систем искусственного интеллекта» в общеобразовательном курсе информатики // Вестник Московского городского педагогического университета. Серия: Информатика и информатизация образования. 2020. № 4 (54). С. 27. EDN: LSTLMA. DOI: 10.25688/2072-9014.2020.54.4.03.
3. Никандров А. А., Пиотровская К. Р. Анализ образовательных данных дисциплины «Основы математической обработки информации» // Проблемы теории и практики обучения математике. Сборник научных работ, представленных на Международную научную конференцию «73 Герценовские чтения» (Санкт-Петербург, 21–25 апреля 2020 года). СПб.: РГПУ им. А. И. Герцена, 2020. С. 91–97. EDN: KТJMKЕ.
4. Рыжова Н. И., Трубина И. И., Королева Н. Ю., Филимонова Е. В. Искусственный интеллект как актуальный тренд содержания обучения информатике в условиях цифровизации // Преподаватель XXI век. 2022. № 2-1. С. 11–22. EDN: ZGIENM. DOI: 10.31862/2073-9613-2022-2-11-22.
5. Ткач Т. В. Машинное обучение и обработка больших данных в условиях современной школы // Информатика в школе. 2020. Т. 19. № 7. С. 25–29. EDN: JETIPR. DOI: 10.32517/2221-1993-2020-19-7-25-29.
6. AtCoder. <https://atcoder.jp/>
7. Codeforces. <https://codeforces.com>
8. Hitron T., Wald I., Erel H., Zuckerman O. Introducing children to machine learning concepts through hands-on experience // Proceedings of the 17th ACM Conference on Interaction Design and Children. 2018. P. 563–568. DOI: 10.1145/3202185.3210776.
9. Ho J. W., Scadding M., Kong S. C., Andone D., Biswas G., Hoppe H. U., Hsu T. C. Classroom activities for teaching artificial intelligence to primary school students // Proceedings of International Conference on Computational Thinking Education. 2019. The Education University of Hong Kong. P. 157–159.
10. LeetCode. <https://leetcode.com>
11. Lin P., Van Brummelen J., Lukin G., Williams R., Breazeal C. Zhorai: Designing a conversational agent for children to explore machine learning concepts // Proceedings of the AAAI Conference on Artificial Intelligence. 2020. Vol. 34. No. 9. P. 13381–13388. DOI: 10.1609/aaai.v34i09.7061.
12. Sabuncuoglu A. Designing one year curriculum to teach artificial intelligence for middle school // Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education. 2020. P. 96–102. DOI: 10.1145/3341525.3387364.
13. Scheidt A., Pulver T. Any-Cubes: A children's toy for learning AI: Enhanced play with deep learning and MQTT // Proceedings of Mensch Und Computer. 2019. P. 893–895. DOI: 10.1145/3340764.3345375.
14. Sulmont E., Patitsas E., Cooperstock J. R. What is hard about teaching machine learning to nonmajors? Insights from classifying instructors' learning goals // ACM Transactions on Computing Education (TOCE). 2019. Vol. 19. No. 4. P. 1–16. DOI: 10.1145/3336124.
15. Touretzky D., Gardner-McCune C., Martin F., See-horn D. Envisioning AI for k-12: What should every child know about AI? // Proceedings of the AAAI Conference on Artificial Intelligence. 2019. Vol. 33. No. 1. P. 9795–9799. DOI: 10.1609/aaai.v33i01.33019795.
16. Williams R., Park H. W., Breazeal C. A is for artificial intelligence: the impact of artificial intelligence activities on young children's perceptions of robots // Proceedings of the Conference on Human Factors in Computing. 2019. P. 1–11. DOI: 10.1145/3290605.3300677.