Процедурная генерация задач по программированию

Петр Советов, РТУ МИРЭА

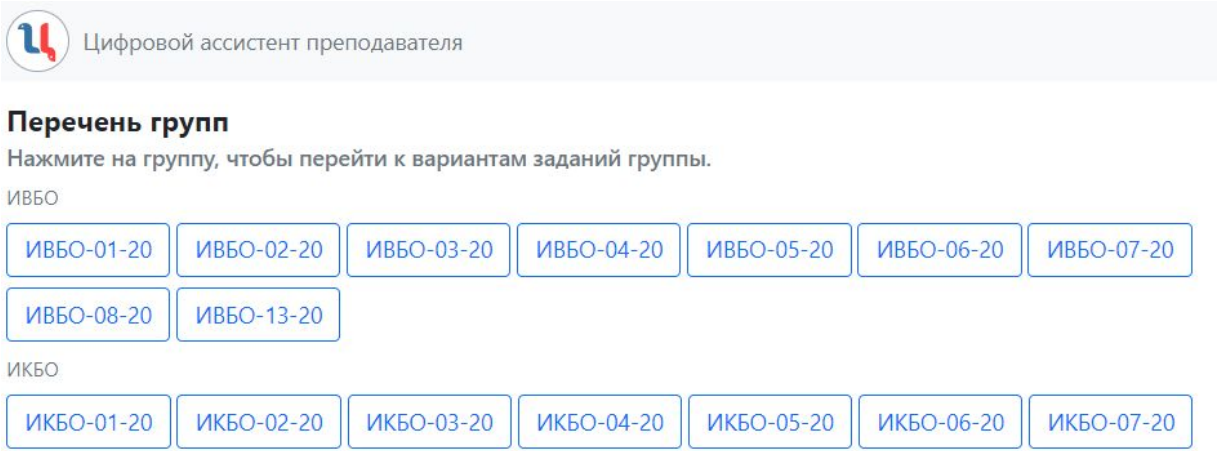
PiterPy, 2022

Предыстория

Новый курс программирования на Питоне в РТУ МИРЭА:

- **1500+** студентов;
- проблема **плагиата**;
- **ручная проверка** решений;
- **новые вариации** задач для тренировки.

Откуда взять столько задач? Процедурно сгенерировать!



Но генераторы задач – не панацея:

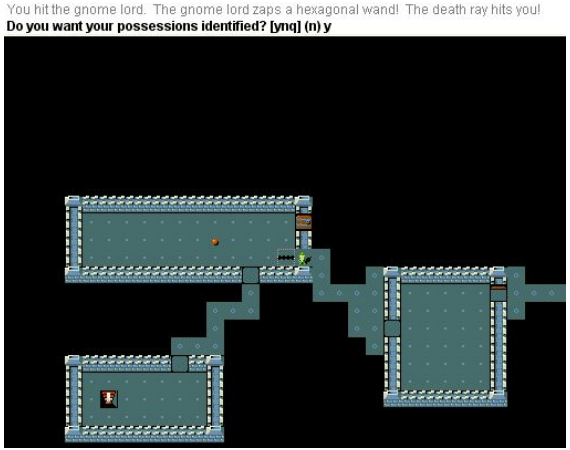
- они используются для **небольших тренировочных** задач;
- преподаватели создают вручную **сложные** и **интересные** задачи.

Вдохновляющие идеи

Процедурная генерация в играх

- **лабиринты;**

Nethack



Dwarf Fortress

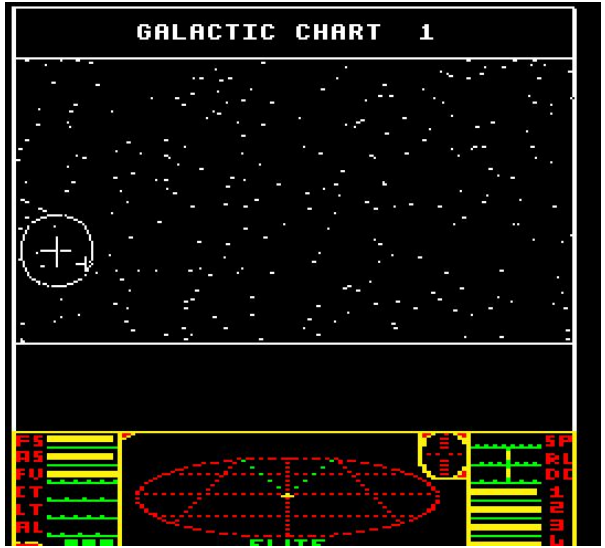
- лабиринты;
- **текстуры и 3d модели;**

.kkrieger

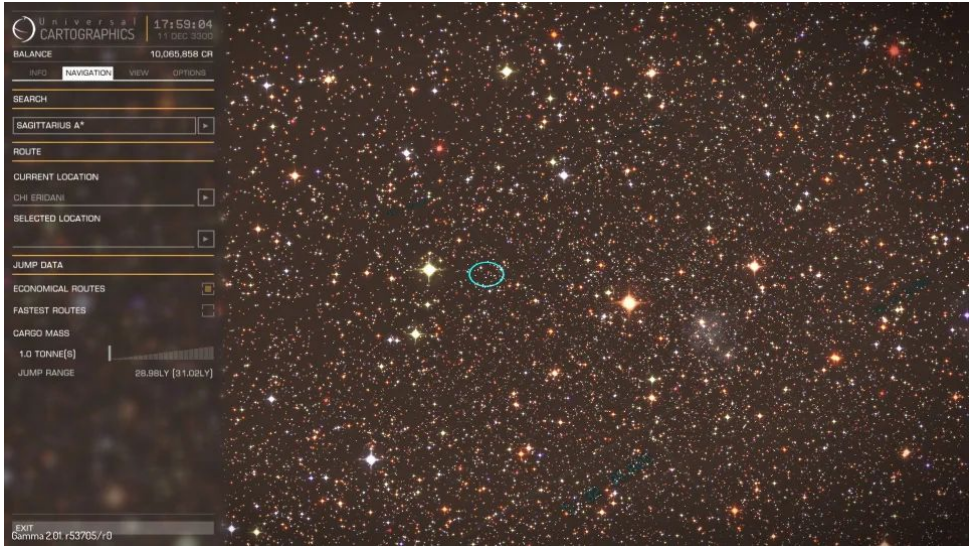
Процедурная генерация в играх

- лабиринты;
- текстуры и 3d-модели;
- **галактики.**

Elite

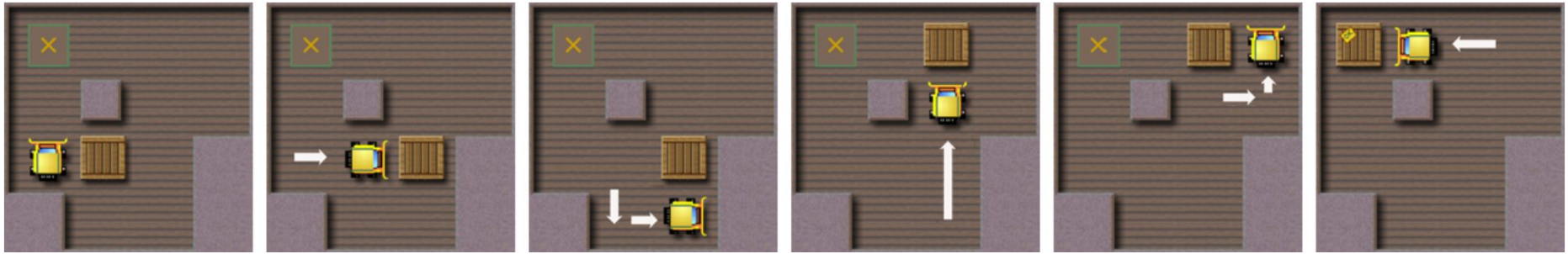


Elite Dangerous



Процедурная генерация головоломок

Sokoban



Как генерируется уровень:

1. Генерация случайной конфигурации из набора шаблонов.
2. Проверка корректности.
3. Проверка на увлекательность.

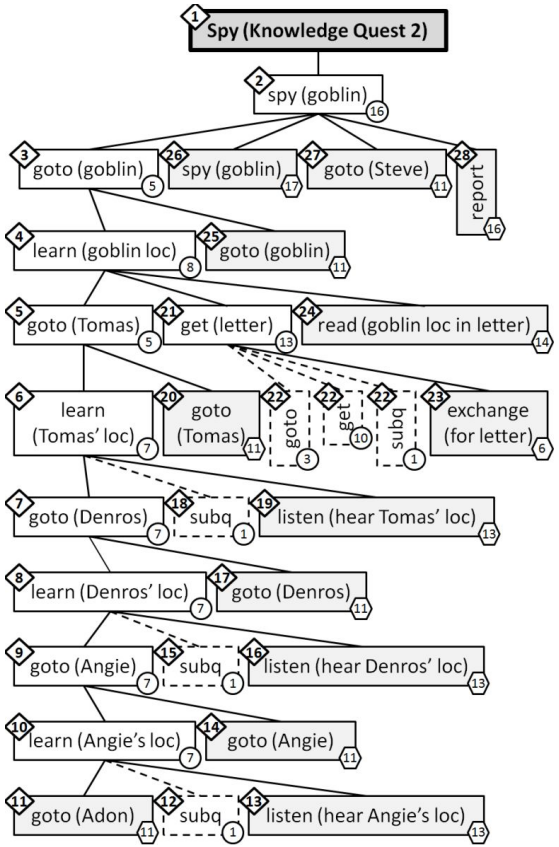
Полезная идея: генерация не от начального, а от целевого состояния.

De Kegel B., Haahr M. Procedural puzzle generation: A survey //IEEE Transactions on Games. – 2019. – Т. 12. – №. 1. – С. 21-40.

Процедурная генерация головоломок

Генератор квестов NPC для MMORPG с использованием **порождающих грамматик**:

```
<QUEST> ::= <Knowledge> | <Comfort> |
<Reputation> | <Serenity> | <Protection> |
<Conquest> | <Wealth> | <Ability> | <Equipment>
<Comfort> ::= <Obtain Luxuries> | <Kill pests>
<Obtain Luxuries> ::= <get> <goto> give
<get> ::= ε [Уже есть]
<get> ::= <steal> [Украсть у кого-нибудь]
 ...
```



Doran J., Parberry I. A prototype quest generator based on a structural analysis of quests from four MMORPGs //Proceedings of the 2nd international workshop on procedural content generation in games. – 2011. – С. 1-8.

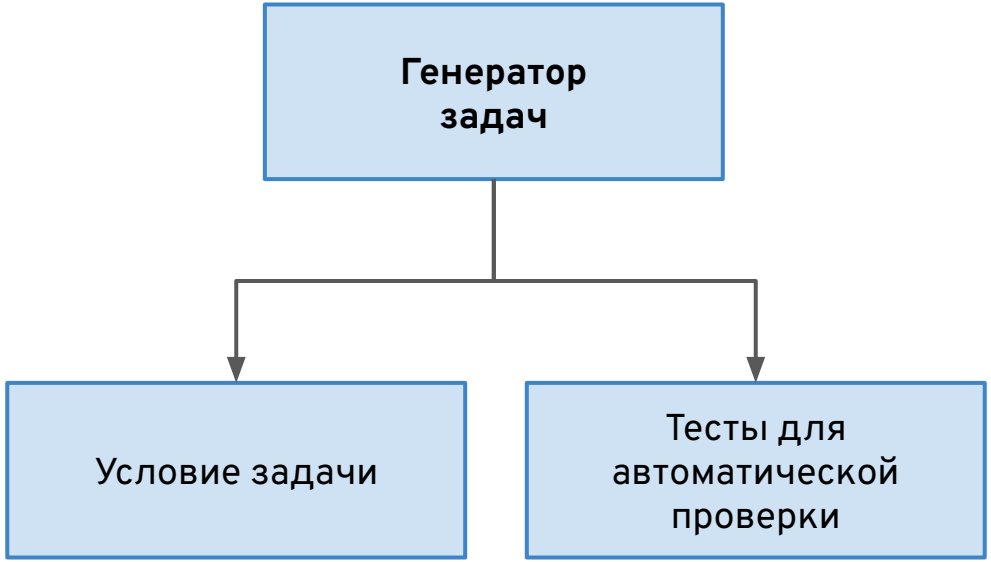
Автоматическое тестирование: фаззинг

```
SELECT O6fo,-977091.1,-36.46 FROM inventory
SELECT g3 FROM inventory WHERE -3.0≠V/g/b+Q*M*G
UPDATE inventory SET z=a,x=F_,Q=K WHERE p(M)<_*S
UPDATE inventory SET R=L5pk WHERE e*l*y-u>K+U(:)
SELECT _/d*Q+H/d(k)<t+M-A+P FROM inventory
SELECT F5 FROM inventory
UPDATE inventory SET jWh.=a6 WHERE wcY(M)>IB7(i)
UPDATE inventory SET U=y WHERE L(W<c,(U≠W))<V(((q)==m<F),O,l)
DELETE FROM inventory WHERE M/b-O*h*E<H-W>e(Y)-P
SELECT ((kP(86)+b*S+J/Z/U+i(U))) FROM inventory
...
```

The Fuzzing Book. URL: https://www.fuzzingbook.org/

О разработанных генераторах задач

Результаты работы генератора задач



Требования к генератору задач

- **практически значимая тематика;**

Требования к генератору задач

- практически значимая тематика;
- **вариативность**;

Требования к генератору задач

- практически значимая тематика;
- вариативность;
- **увлекательность**;

Требования к генератору задач

- практически значимая тематика;
- вариативность;
- увлекательность;
- **контроль**.

Требования к генератору задач

- практически значимая тематика;
- вариативность;
- увлекательность;
- контроль;
- **быстродействие**.

Преобразование мат. нотации в код

Реализовать функцию:

$$f(x,z,y)=\sqrt{\frac{\left(z-45y^3\right)^7-29\left(\lfloor x^2-y^3-58y\rfloor\right)^6}{61x-85\arcsin^5\left(1-z^3-y^2\right)}}+\sqrt{\frac{x^3+4\left(93y^2-z\right)}{71x^5+\left(y-41x^2-z^3\right)^4}}$$

Реализовать функцию по рекуррентной формуле:

$$f_n=\begin{cases}-0.25, & n=0;\\ -0.55, & n=1;\\ 77\log_{10}\left(97+51f_{n-1}+f_{n-2}^3\right), & n\geq 2.\end{cases}$$

Реализовать итерационную функцию:

$$f(b,n,m) = \sum_{k=1}^{n}\sum_{c=1}^{b}\left(1+\frac{\left(c^2+1+63k^3\right)^2}{45}\right)+\prod_{i=1}^{b}\sum_{j=1}^{m}\sum_{c=1}^{n}\left(\left(c+2+\frac{i^2}{14}\right)^5-\frac{j^3}{21}\right)$$

Реализовать функцию, оперирующую векторами длины n:

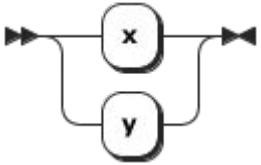
$$f(\vec{z},\vec{y},\vec{x}) = 40\sum_{i=1}^{n}3\left(\sqrt{92x_{\lceil i/4\rceil}^3+68y_{\lceil i/4\rceil}^2+\frac{z_{n+1-\lceil i/2\rceil}}{34}}\right)^4$$

Грамматика арифметического выражения

Форма Бэкуса-Наура (БНФ):

```
var ::= 'x' | 'y'
```

var:

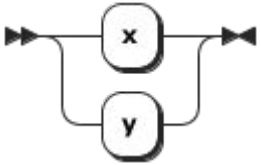


Грамматика арифметического выражения

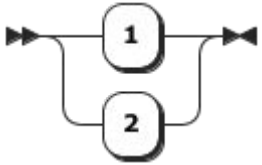
Форма Бэкуса-Наура (БНФ):

```
var ::= 'x' | 'y'
num ::= 1 | 2
```

var:



num:

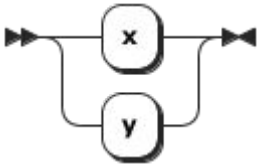


Грамматика арифметического выражения

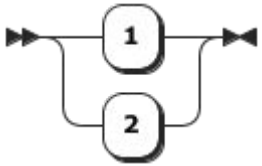
Форма Бэкуса-Наура (БНФ):

```
var ::= 'x' | 'y'
num ::= 1 | 2
 op ::= '+' | '*'
```

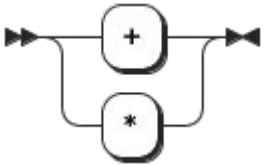
var:



num:



op:

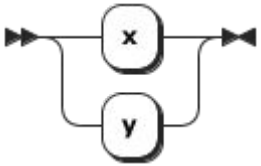


Грамматика арифметического выражения

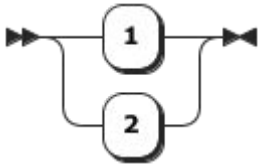
Форма Бэкуса-Наура (БНФ):

```
var ::= 'x' | 'y'
num ::= 1 | 2
 op ::= '+' | '*'
expr ::= var | num | '(' expr ')' | expr op expr
```

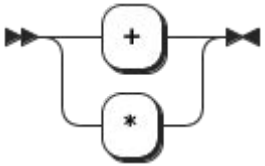
var:



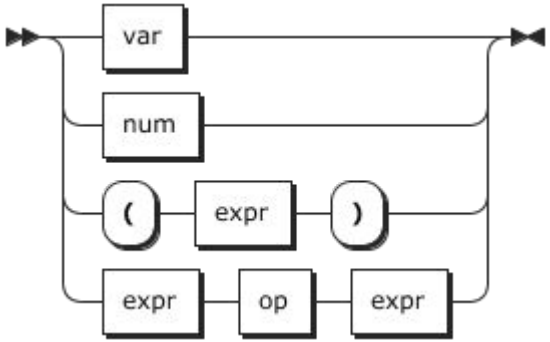
num:



op:



expr:



Грамматики бывают распознающие и **порождающие**!

DSL описания случайных выражений

(Domain Specific Language)

```
SOURCE = '''
var = x | y
num = 1 | 2
op = + | *
expr = var | num | ( expr ) | expr op expr'''
```

DSL: синтаксический разбор

```python
def parse_bnf(source):
    grammar = {}
    rules = [line.split('=') for line in source.strip().split('\n')]
    for name, body in rules:
        grammar[name.strip()] = [alt.split() for alt in body.split('|')]
    return grammar
```

```
{'expr': [['var'], ['num'], ['(', 'expr', ')'], ['expr', 'op', 'expr']],
 'num': [['1'], ['2']],
 'op': [['+'], ['*']],
 'var': [['x'], ['y']]}
```

DSL: порождение случайных выражений


```
def generate_expr(grammar, rule):
    if rule in grammar:
        seq = choice(grammar[rule]) # Случайный выбор альтернативы
        return ''.join([generate_expr(grammar, name) for name in seq])
    return str(rule)
```

```
>>> generate_expr(parse_bnf(SOURCE), 'expr')
```

DSL: примеры работы


```
(y)
x*2*1*(x)
(((y+2)+(2)))*(y)
1+x+x
2
2+1*y*y+(2)
y
2+((1))*2
x
(2)
1
1
2
y*(y)+(y)*x+y
(1)
y
x+((2)+2+1+x*2*(y*((1))+x*2))
...
```

– **выбор осуществляется равновероятно;**

```
expr = var [20%] | num [20%] | ( expr ) [10%] | expr op expr [50%]
```

Проблемы DSL

- выбор осуществляется равновероятно;
- **контекст не учитывается;**

$$f(n,m,x) = \sum_{i=1}^{m}\sum_{c=1}^{n}\left(\left(90x^2\right)^5 + c^2 + 88\left(\sqrt{46x+i^2}\right)^4\right)$$

Проблемы DSL

- выбор осуществляется равновероятно;
- контекст не учитывается;
- **нет пользовательских функций в правилах;**

```
num = randint(0, 1000)
```

Проблемы DSL

- выбор осуществляется равновероятно;
- контекст не учитывается;
- нет пользовательских функций в правилах;
- **выдается строка, а не промежуточное представление.**

Пример AST

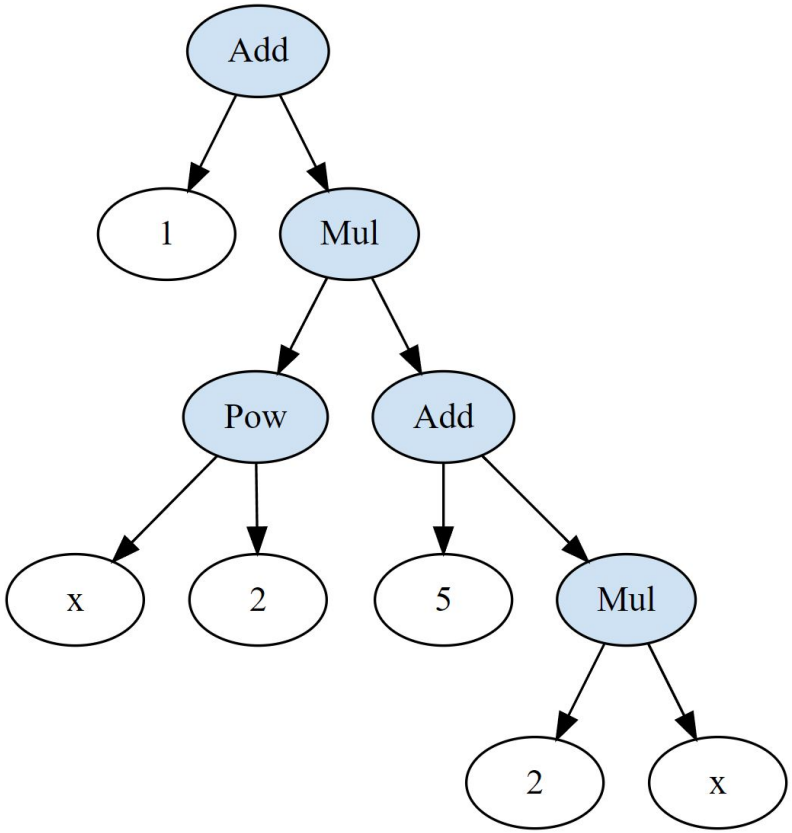
(Abstract Syntax Tree)

```
x**2*(2*x + 5) + 1
```

Пример AST

(Abstract Syntax Tree)

```
x**2*(2*x + 5) + 1
```

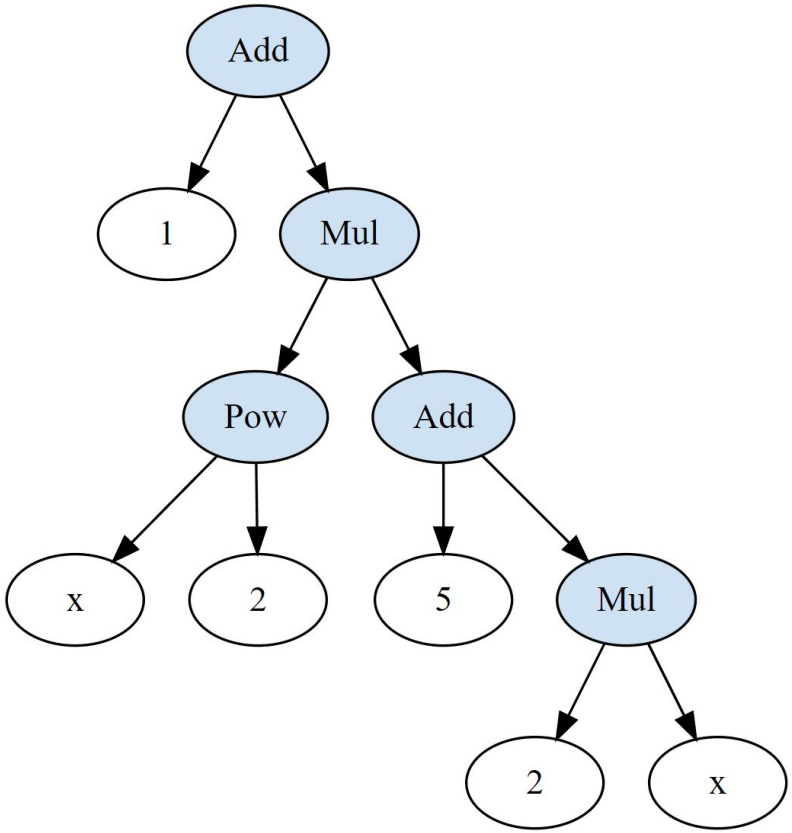


Пример AST

(Abstract Syntax Tree)

```
x**2*(2*x + 5) + 1
```

```
Add(
  Mul(
    Pow(Symbol("x"), Integer(2)),
    Add(Mul(Integer(2), Symbol("x")),
        Integer(5))
  ),
  Integer(1)
)
```



Встроенный функциональный DSL (eDSL)


```
def app(f): return f() if callable(f) else f
def alt(*args, prob=None): return choices(args, prob)[0]

var = Symbol('x')
num = lambda: Integer(randint(1, 10))
op = lambda: alt(Add, Mul)
prim = lambda: app(alt(var, num, expr))
binary = lambda: op()(expr(), expr())
expr = lambda: app(alt(binary, prim, prob=[0.3, 0.7]))
```

SymPy

Примеры работы eDSL

$$8x + 80$$

$$10x^2 + x + 8$$

$$108x\left(x + 6\right)\left(5x + 3\right)$$

$$399x^3$$

$$8\left(x + 5\right)\left(x^2 + 3x + 5\right)$$

$$x^2 + 2x + 6$$

$$6x^2 + 3x + 12$$

$$24x + 192$$

$$6x\left(x^2 + x + 11\right)$$

$$x\left(x + 3\right)\left(x + 8\right)$$

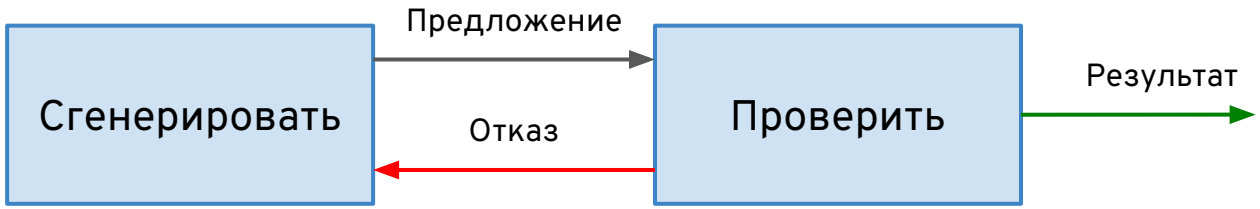
$$\ldots$$

LaTeX

Выбор подходящих выражений


```
def filtered_expr(expr):
    while True:
        e = simplify(expr())
        if e.count(x) and 1 < e.count_ops() < 10 and \
           not e.has(oo, -oo, zoo, nan):
            return e
```

Универсальный метод generate-and-test



- эволюционные алгоритмы;
- задачи удовлетворения ограничениям (constraint satisfaction problems);
- генеративно-состязательные сети (generative adversarial networks);
- …

Проблема: превышение глубины рекурсии

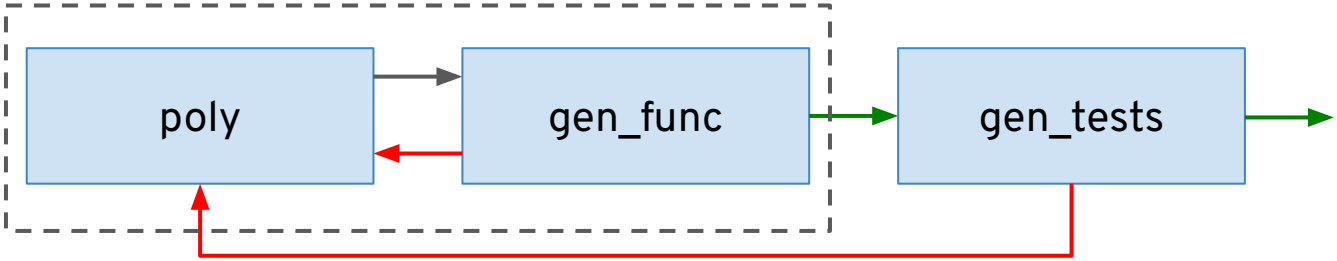

```
# expr = lambda: app(alt(binary, prim, prob=[0.3, 0.7]))
expr = lambda: app(alt(binary, prim))
```

```
Traceback (most recent call last):
...
    n = len(population)
RecursionError: maximum recursion depth exceeded while calling a Python object
```

Упрощенный генератор задач: архитектура

(вариации на перевод полинома в код)

- генератор задач разбит на подгенераторы — **иерархический generate-and-test**;
- для большего контроля используются **шаблоны** (poly). Это также позволит **избежать RecursionError**.



Упрощенный генератор задач: одночлен

```
def mono(p):
    return alt(
        Mul(Integer(randint(-10, 10)), Pow(x, p)),
        Mul(1 / Integer(randint(-10, 10)), Pow(x, p)),
        prob=[0.7, 0.3]
    )
```

$$8x^6$$
$$\frac{x^2}{7}$$
$$-2x^5$$
$$-x^6$$
$$5x^6$$
$$-9x^3$$
$$-x$$
$$-8x^3$$
$$8x^3$$
$$-\frac{x^5}{2}$$
$$\ldots$$

Упрощенный генератор задач: многочлен


```
def poly(max_power, max_terms):
    powers = sample(range(1, max_power),
                    randint(1, max_terms))
    return Add(*[mono(p) for p in powers])
```

$$9x^4 - 6x$$

$$7x^3 - 9x^2 - 9x$$

$$-7x^4 - 8x$$

$$-4x^4 + \frac{x^3}{9} - 4x^2 - 10x$$

$$-10x^4 + 10x^3 - 7x^2$$

$$-\frac{x}{9}$$

$$\frac{x^4}{7} + \frac{x^3}{4} - 3x^2 - 10x$$

$$-5x^2 + \frac{x}{8}$$

$$2x^3$$

$$-\frac{x^4}{6} - 4x^3 + 5x^2 + 5x$$

$$\ldots$$

Упрощенный генератор задач: тесты

```
def gen_tests(func, num_tests):
    tests = []
    while len(tests) < num_tests:
        x = round(uniform(-1, 1), 2)
        y = func.evalf(subs={'x': x})
        if (x, y) not in tests:
            tests.append((x, y))
    return tests
```

Возможные дополнительные ограничения для тестов:

- несовпадающие значения **y**;
- диапазон для **y**;
- время выполнения теста;
- ...

Упрощенный генератор задач: примеры условий

Задача 1

Реализовать функцию:

$$f(x) = -4x^4 + 7x^3 - \frac{x}{8}.$$

```
f(0.56) ≈ 7.66e-1
f(0.12) ≈ -3.73e-3
f(-0.42) ≈ -5.91e-1
f(-0.18) ≈ -2.25e-2
```

Задача 2

Реализовать функцию:

$$f(x) = \frac{x\left(-18x - 1\right)}{9}.$$

```
f(-0.46) ≈ -3.72e-1
f(-0.95) ≈ -1.70e+0
f(0.08) ≈ -2.17e-2
f(-0.25) ≈ -9.72e-2
```

Pandoc

```
>>> math.isclose(7.66e-1, 7.663e-1, rel_tol=1e-3)
True
>>> math.isclose(7.66e-1, 7.673e-1, rel_tol=1e-3)
False
```

В реальных генераторах задач встречаются мат. функции…

“The math module consists mostly of thin wrappers around the **platform C math library** functions” (https://docs.python.org/3/library/math.html)

```
Python 3.10.2 (tags/v3.10.2:a58ebcc, Jan 17 2022, 14:12:15) [MSC
v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more
information.
>>> import math; f'{math.tan(0.2766632934733237):.20e}'
'2.83945164866099886147e-01'
```

```
Python 3.10.2 (main, Jan 15 2022, 18:02:07) [GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more
information.
>>> import math; f'{math.tan(0.2766632934733237):.20e}'
'2.83945164866099941658e-01'
```

...и math.isclose здесь поможет

```
def f(n):
    if n == 0: return 0.2
    else: return math.tan(1 + 21 * f(n - 1) ** 3)
```

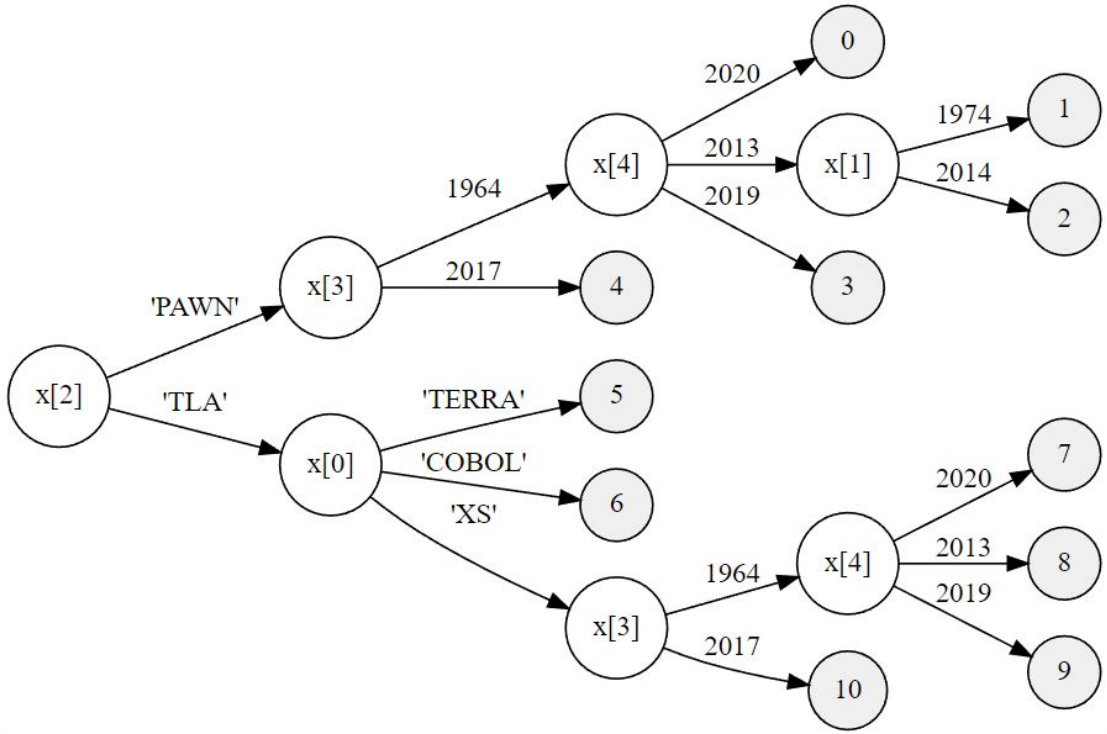
```
>>> f(21) # Windows
-0.10428733799246341

>>> f(21) # Linux
-0.48646202888412904
```

Решение:
Вычислять каждый тест **дважды**. С полной точностью и с **урезанной** точностью мат. функций. Пропускать только функции без различий между результатами обоих прогонов.

Генерация задач не ограничивается математикой!

Генератор задач на вычисление дерева решений



Примеры результатов вычислений:

```
main(['COBOL', 1974, 'TLA', 1964, 2020]) = 6
main(['TERRA', 1974, 'PAWN', 2017, 2019]) = 4
```

Graphviz

Анализ кода студентов

Многие создают одну функцию с большим количеством вложенных if'ов.
Некоторые создают интерпретатор деревьев решений.
Но есть и оригинальные подходы:

```
def main(x):
    arr = [[x[0], 1965, 2015, x[3], 'PAN'],
           [x[0], 1965, 2015, x[3], 'LEX'],
           [x[0], 1965, 1967, x[3], x[4]],
           [x[0], 2012, 2015, x[3], x[4]],
           [1992, 2012, 1967, x[3], 'PAN'],
           [1990, 2012, 1967, x[3], 'PAN'],
           [1992, 2012, 1967, x[3], 'LEX'],
           [1990, 2012, 1967, x[3], 'LEX'],
           [x[0], 2005, 2015, 2017, x[4]],
           [x[0], 2005, 1967, 2017, x[4]],
           [x[0], 2005, x[2], 1993, x[4]]]
    return arr.index(x)
```

Проверка качества присланного кода

- библиотека **pycodestyle** для проверки на **PEP8**;
- библиотека **Radon** для проверки на **цикломатическую сложность** (Cyclomatic Complexity, CC) — количество различных путей выполнения в функции.

  Если CC > 10, то код нуждается в рефакторинге.

В будущем мы хотим перейти от меры цикломатической сложности к когнитивной сложности: https://blog.sonarsource.com/cognitive-complexity-because-testability-understandability/

К сожалению, наша версия Radon не всегда помогала


```
def main(x):
    match x[4]:
        case 2016:
            match x[2]:
                case 1967:
                    match x[3]:
                        case "MAX":
                            match x[1]:
                                case "NIT":
                                    f = 0
                                case "XML":
                                    f = 1
                                case "MUPAD":
                                    f = 2
                        case "XS":
                            f = 3

                case 1995:
                    match x[0]:
                        case 1995:
                            f = 4
                        case 1987:
                            f = 5
...
```

Генератор задач на преобразование табличных данных (1)

Пример 1

Исходная таблица:

1	2
Алексей Ч. Кешобиди;0.702	99.12.25
Рамиль Р. Сакиди;0.282	04.08.14
Рамиль Р. Сакиди;0.282	04.08.14
Федор Ц. Шизамянц;0.995	99.11.21
Рамиль Р. Сакиди;0.282	04.08.14

Результат преобразования:

1	2	3
0.7020	Кешобиди А.Ч.	25.12.99
0.2820	Сакиди Р.Р.	14.08.04
0.9950	Шизамянц Ф.Ц.	21.11.99

Генератор задач на преобразование табличных данных (2)

Пример 1

Исходная таблица:

1	2	3	4
(736) 168-53-89	Да	Табушберг, А.К.	0.05
(146) 023-41-73	Да	Нушешянц, Г.И.	0.24
(145) 344-11-84	Да	Вефук, А.Г.	0.71
(748) 991-72-72	Нет	Мулигин, Ф.М.	0.08

Результат преобразования:

1	2	3	4
0234173	1685389	3441184	9917272
1	1	1	0
Г. Нушешянц	А. Табушберг	А. Вефук	Ф. Мулигин
24%	5%	71%	8%

Генератор задач на разбор двоичных данных (а-ля PNG)

Реализовать разбор двоичного формата данных. Данные начинаются с сигнатуры `0x39 0x54 0x50 0x4c`, за которой следует структура A. Порядок байт: от старшего к младшему. Адреса указаны в виде смещений от начала данных. В решении предлагается использовать модуль struct.

Структура A:

Поле	Описание
1	Массив адресов (uint16) структур B, размер 3
2	Размер (uint32) и адрес (uint16) массива char
3	int16
4	Структура C

Структура B:

Поле	Описание
1	int8
2	double
3	int8

Структура C:

Поле	Описание
1	int32
2	Адрес (uint32) структуры D
3	Массив int8, размер 7

Пример 1

Двоичные данные:

```
(b'9TPL\x00!\x00+\x005\x00\x00\x00\x03\x00?\xccW\xc8\x8f\xac\xa1\x00\x(
 b"\x00B\x15'5\xf7}\x98\xd8\x95?\xedK\xad\xc6\x8f}RW\xfb?\xdd\x19eh\\\:
 b'1\xe5\xbf\xbb\xca\x8b\x10\xac\x1b\x10\xf8trhZ\xdf\x86]#\xfd\xeeH\xd
 b'\xbe\xc6\xf2\xf5Rq\x7fN\xac0\x80\xe8k\x11\xe4\x151\xcb\xf3;\x1e\xd9
 b'\x02\x18\x12n\x01\x0c\x92+\xfb\xc0')
```

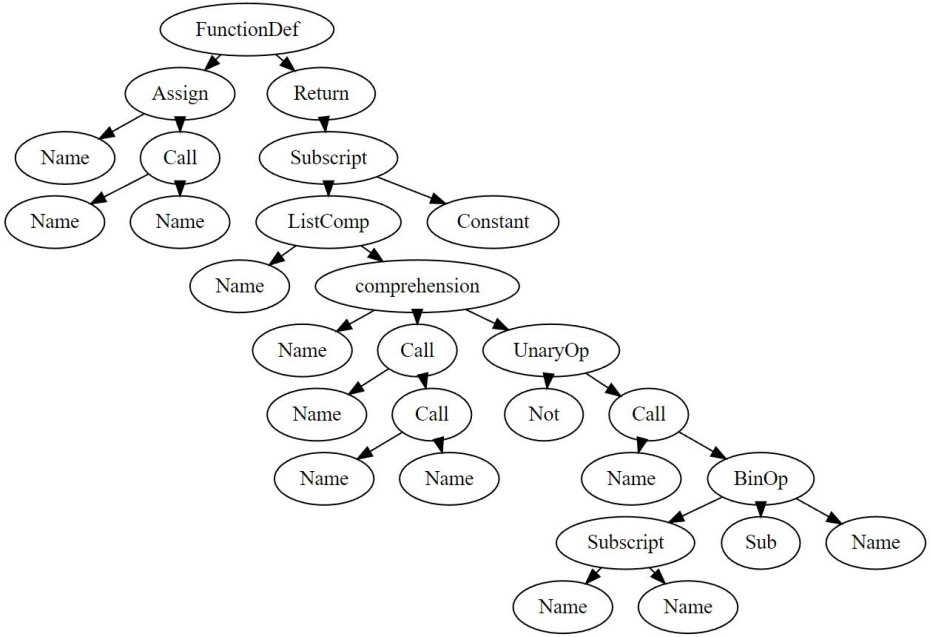
Результат разбора:

```
{'A1': [{'B1': -107, 'B2': 0.9154881361136324, 'B3': 87},
        {'B1': -5, 'B2': 0.4546750563768178, 'B3': 49},
        {'B1': -27, 'B2': -0.10855931431806831, 'B3': -8}],
 'A2': 'trh',
 'A3': -13225,
 'A4': {'C1': -930108255,
        'C2': {'D1': 23263,
               'D2': 134,
               'D3': [93, 35, 253, 238, 72],
               'D4': [16085378780796113521,
                      9173458815321402129,
                      16435097167187353305],
               'D5': 5493830897850253580,
               'D6': -1842611264},
        'C3': [21, 39, 53, -9, 125, -104, -40]}}
```

А как оценить качество решения задачи?

Автоматическая оценка качества решения

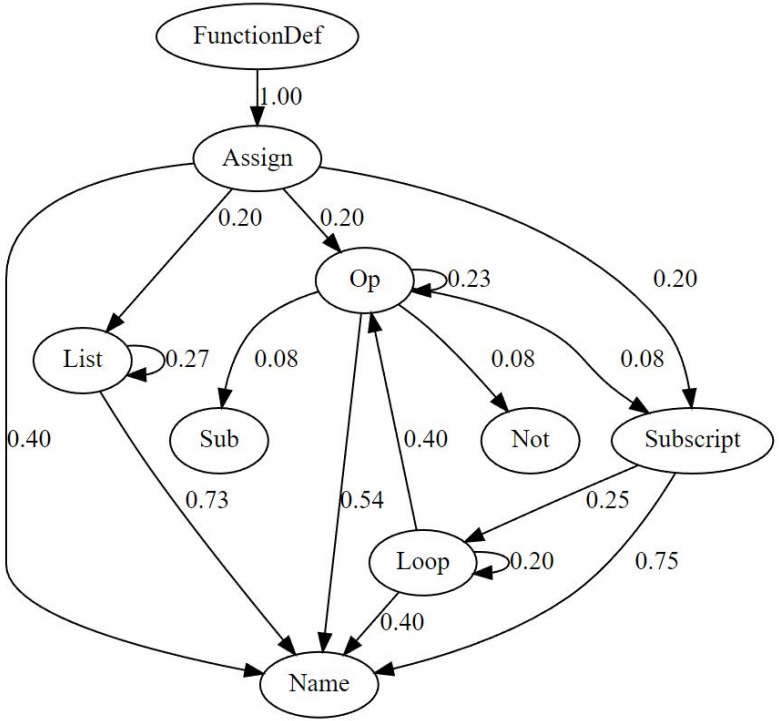
- **присланный код автоматически разбивается на кластеры по способам решения;**



Л. А. Демидова, П. Н. Советов, А. В. Горчаков, Кластеризация представлений текстов программ на основе цепей Маркова

Автоматическая оценка качества решения

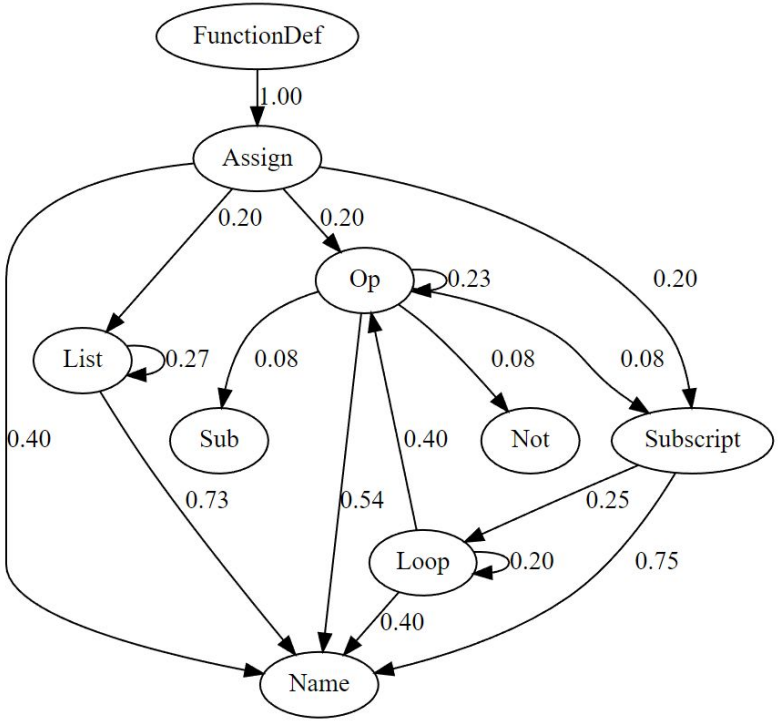
- присланный код автоматически разбивается на кластеры по способам решения;
- **преподаватели вручную помечают кластеры числом баллов;**



Л. А. Демидова, П. Н. Советов, А. В. Горчаков, Кластеризация представлений текстов программ на основе цепей Маркова

Автоматическая оценка качества решения

- присланный код автоматически разбивается на кластеры по способам решения;
- преподаватели вручную помечают кластеры числом баллов;
- **система автоматически определяет принадлежность нового решения к одному из кластеров.**



Л. А. Демидова, П. Н. Советов, А. В. Горчаков, Кластеризация представлений текстов программ на основе цепей Маркова

Выводы

Система ЦАП успешно используется уже **2 года**:

- **исчез плагиат**;
- оценки с точки зрения студентов стали более **объективными**;
- преподаватели тратят м**еньше сил на рутину**.
- регулярно добавляются генераторы задач **новых типов**.

Спасибо за внимание!

Генератор задач на разбор конфигурационного формата (1)

Пример 1

Входная строка:

```
<block> {{equ -971 => zatila; }};{{equ -3041=> zaed; }};{{ equ -6868
=> iserat; }}; {{ equ -4255=> laaza; }};</block>
```

Разобранный результат:

```
{'zatila': -971, 'zaed': -3041, 'iserat': -6868, 'laaza': -4255}
```

Пример 2

Входная строка:

```
<block> {{ equ -639=> sois; }}; {{ equ 4074=> tete;}};{{ equ 2605 =>
maan; }};</block>
```

Разобранный результат:

```
{'sois': -639, 'tete': 4074, 'maan': 2605}
```

Генератор задач на разбор конфигурационного формата (2)

Пример 1

Входная строка:

```
do .begin new { #2281 . #-293 . #9390 . #3934 } =: bied. .end .begin
new{ #9101 . #9069 . #-720} =: atinxe. .end done
```

Разобранный результат:

```
{'bied': [2281, -293, 9390, 3934], 'atinxe': [9101, 9069, -720]}
```

Пример 2

Входная строка:

```
do .begin new {#328 . #3658 . #-6214 } =: ceeda_630. .end .begin new
{#-2913 . #8224 } =: ribe. .end .begin new { #7281 . #2576 } =:
riso_491. .end .begin new {#3358 . #-2243 } =: raquin_703. .end done
```

Разобранный результат:

```
{'ceeda_630': [328, 3658, -6214],
 'ribe': [-2913, 8224],
 'riso_491': [7281, 2576],
 'raquin_703': [3358, -2243]}
```