Определение. Эзотерические языки программирования (далее ЭЯП) – это вид ЯП, не предназначенных для практического применения. Некоторые из этих языков имеют намеренно ограниченные возможности, другие же полны по Тьюрингу. Нередко ЭЯП создаются в качестве шутки, но в контексте статьи нас в первую очередь будут интересовать языки, создающие новые концепции программирования, и именно на эти концепции мы будем опираться при обзоре языков.

1) Максимальное отличие.

В эту категорию попадают ЭЯП, созданные с целью получения ЯП, максимально непохожего на существующие.

INTERCAL – один из первых эзотерических языков программирования, созданный в 1972 году. Этот язык был изначально задуман как пародия на существовавшие тогда языки, и создатели целенаправленно занимались его усложнением. Об этом говорит, например, существование оператора *PLEASE,* не имевшего какой-либо смысловой нагрузки.

Пример кода:

```
DO ,1 <- #13
PLEASE DO ,1 SUB #1 <- #238
DO ,1 SUB #2 <- #108
DO ,1 SUB #3 <- #112
DO ,1 SUB #4 <- #0
DO ,1 SUB #5 <- #64
DO ,1 SUB #6 <- #194
PLEASE DO ,1 SUB #7 <- #48
DO ,1 SUB #8 <- #26
DO ,1 SUB #9 <- #244
PLEASE DO ,1 SUB #10 <- #168
DO ,1 SUB #11 <- #24
DO ,1 SUB #12 <- #16
DO ,1 SUB #13 <- #162
PLEASE READ OUT ,1
PLEASE GIVE UP
```

Эта программа выводит на экран надпись «Hello, world!». Здесь *SUB* – это индекс массива, # – префикс константы, <- – оператор присваивания. При этом оператор *PLEASE* должен быть использован 4 или 5 раз, в противном случае возникнет ошибка избыточной или недостаточной вежливости.

Тьюринг-полнота: да.

FALSE – созданный в 1993 году стековый ЭЯП. Его создатель преследовал две цели: во-первых, возможность написания компилятора размером менее килобайта, во-вторых, сильная обфускация.

Арифметические операторы берут два значения с вершины стека, совершают над ними соответствующую операцию и возвращают результат в стек. Помимо этого, язык имеет операции сравнения, логические И, ИЛИ и отрицание, а также довольно специфические операции для работы со стеком – дублирование или вершины стека, циклическая перестановка двух или верхних элементов стека, а также копирование n-ого элемента стека на вершину. Переменные обозначаются строчной латинской буквой, но обычно используются для хранения лямбда-выражений. Сами лямбда-выражения записываются в квадратных скобках

Пример кода:

```
[1+]i:
a;1=[2i;!]?
```

Сначала объявляется функция i, выполняющая инкрементацию числовой переменной. После чего к числу 2 применяется функция i, если значение переменной a равно единице.

Тьюринг-полнота: да.

Thue – разработанный в 2000 году мета-язык, позволяющий реализовать любые языки.

Программы на этом языке состоят из двух частей. Первая – это таблица правил, построенных следующим образом:

<исходная строка>::=<строка замены>,

заканчивающаяся пустым правилом

::=

Вторая часть – это начальное состояние, записанное после таблицы в виде строки символов. Во время работы программа ищет в строке символы из левой половины таблицы и заменяет их на символы из правой в соответствии с указанными правилами. Работа завершается, когда к строке нельзя применить ни одного правила. Для ввода и вывода существуют отдельные правила:

<исходная строка>::=:::

При активации указанная строка будет заменена на строку из входного потока.

Аналогично, вывод активируется следующим образом:

<исходная string>::=~*<выходная строка>*

Пример кода:

```
1_::=1++
0_::=1
01++::=10
11++::=1++0
_0::=_
_1++::=10
::=
_1111111111_
```

Эта программа реализует инкрементацию двоичного числа, ограниченного знаками подчёркивания.

Тьюринг-полнота: по принципу работы программа на *Thue* аналогична машине Тьюринга, поэтому язык является Тьюринг-полным.

Spoon – диалект одного из наиболее известных ЭЯП. От оригинала записывается лишь тем, что команды в нём записываются последовательностями нулей и единиц. При этом синтаксис команд таков, что даже при записи программы без разделителей текст программы можно однозначно определить.

Управляемая языком машина состоит из упорядоченного набора ячеек и указателя текущей ячейки. В классической реализации машина содержит 30000 ячеек размером 1 байт каждая.

Spoon поддерживает следующие команды:

1	Значение текущей ячейки увеличивает на 1
000	Значение текущей ячейки уменьшают на 1
010	Следующая ячейка
011	Предыдущая ячейка
00100	Начало цикла
0011	Конец цикла
0010110	Запрос значения текущей ячейки
001010	Вывод значения текущей ячейки

Пример кода:

```
1 1 1 1 1 1 1 1 1 00100 010 1 1 1 1 1 1 1 010 1 1 1 1 1 1 1 1 1 010 1 1 1
010 1 011 011 011
011 000 0011 010 1 1 001010 010 1 001010 1 1 1 1 1 1 1 001010 001010 1 1 1
001010 010 1 1 001010
011 011 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 001010 010 001010 1 1 1 001010 000 000
000 000 000 000
001010 000 000 000 000 000 000 000 000 001010 010 1 001010 010 001010
```

В этом примере нулевая ячейка (счётчик) увеличивается на 10, после чего в ячейки с первой по четвёртую в цикле заносятся определённые числовые значения. После выхода из цикла ячейки окончательно дополняются до значений, соответствующих необходимым символам в ASCII-таблице, и символы выводятся на экран.

Тьюринг-полнота: да.

Malbolge – известный своей сложностью ЭЯП. Его создатель преследовал цель создания максимально сложного для понимания языка программирования.

Язык предназначен для машины, работающей в троичной системе счисления и имеющей три регистра: **a** – аккумулятор, **c** – регистр кода и **d** – регистр данных, а также 59049 ячеек памяти, имеющими значения, равные номерам ячеек. Машина определяет выполняемую команду по следующему алгоритму: к значению ячейки с адресом **c** прибавляется значение **c**, после чего берётся остаток от деления получившейся суммы на число 94.

Команды *Malbolge*:

Значение ([c]+c) % 94	Описание	Псевдокод
4	Переход к ячейке с номером [**d**]	C = [D]
5	Вывод на экран ASCII-символа с кодом **a** % 256	Print (A%256)
23	Ввод ASCII-символа в **a**	A = input
39	Значение [**d**] сдвигается вправо и сохраняется в [**d**] и **a**	A = [D] = rotate_right([D])

40	Запись значения [**d**] в **d**	D = [D]
62	Произвести троичную операцию “crazy”	A = [D] = crazy_op(A, [D])
68	Отсутствие операции	NOP
81	Конец программы	STOP

Таблица операции “crazy”:

crz		**A**		
		0	**1**	**2**
[**D**]	**0**	1	0	0
	1	1	0	2
	2	2	2	1

После выполнения каждая инструкция дополнительно шифруется по таблице перевода, а значения **c** и **d** инкрементируются.

Пример кода:

```
(=<`:9876Z4321UT.-Q+*)M'&%$H"!~}|Bzy?=|{z]KwZY44Eq0/{mlk**hKs_dG5[m_BA{?-Y;;V
b'rR5431M}/.zHGwEDCBA@98\6543W10/.R,+O<
```

Эта программа выводит на экран надпись «Hello, world!», однако язык настолько сложен для понимания, что создать данную программу удалось лишь с помощью специального ПО для генерации кода на *Malbolge*.

Тьюринг-полнота: в реализациях с неограниченной памятью.

2) Многомерные языки.

Befunge – стековый двухмерный язык программирования, созданный в 1993 году. Программа на этом языке записывается в тор, по которому перемещается указатель.

Пример кода:

```
vv  <       <
     2
     ^  v<
  v1<?>3v4
```

```
     ^   ^
  >  >?>  ?>5^
     v   v
   v9<?>7v6
     v  v<
     8
   .  >  >   ^
  ^<
```

Эта программа реализует генератор случайных чисел. Символы <, >, ^ и v задают направление перемещения указателя, а «?» означает перемещение в случайном направлении. При проходе указателя через числовую константу соответствующее число добавляется на вершину стека, а «.» соответствует команде вывода вершины стека целым числом.

Тьюринг-полнота: на бесконечной таблице.

><> (читается «fish») – во многом аналогичный *Befunge* язык.

Основное отличие – возможность создавать несколько стеков, а также инструкции отражения, изменяющие движение указателя в зависимости от того, в какую сторону он бы направлен изначально.

Argh! – ещё один двухмерный стековый ЭЯП.

Виртуальная машина этого языка работает с таблицей размерами 80x40 ячеек. Выполнение программы начинается с ячейки [0, 0], направление считывания кода задаётся командами **h, j, k** и **l,** означающими движение влево, вниз, вверх и вправо соответственно.

Пример кода:

```
j     world
lppppppPPPPPPsrfj
 hello,      *  j
            qPh
```

В этом примере указатель начинает движение вниз, после чего продолжает двигаться вправо. Оператор **p** указывает на вывод на экран символа, находящегося в ячейке под ним, а заглавная **P** – вывод символа в верхней ячейке. Однако печать символа переноса строки осуществляется более интересным способом. Команда **s** копирует находящийся в нижней ячейке символ («*» с ASCII-кодом 42), **r** уменьшает значение стека на величину, равную коду символа из нижней ячейки (ASCII-код пробела – 32, при

вычитании получается 10 – код переноса строки). **f** сохраняет значение вершины стека в нижнюю ячейку, две команды **j** и команда **h** переводят указатель вниз и влево, где нужный символ печатается оператором **P.** Оператор **q** означает конец программы.

Тьюринг-полнота: в реализации *Aargh!,* имеющей неограниченное число строк.

4DL– первый четырёхмерный стековый ЯП. Программа записывается на четырёхмерной решётке, соответственно, может иметь 8 направлений выполнения.

Команды языка позволяют указать перемещение указателя в любом из восьми направлений, перемещение в стек соседнего в одном из направлений символа, математические операции со стеком и операции ввода-вывода.

Однако пользователь ресурса habrahabr.ru указывает на скудные возможности языка с таким набором команд, и предлагает свою реализацию, в которую добавлены команды \ (перестановка двух верхних ячеек стека) и ^ (снятие числа с вершины стека). С использованием этих команд он реализует вычисление суммы чисел из входной строки.

Пример кода:

Рис. 1. Скриншот кода программы, приведённого в статье

Тьюринг-полнота: зависит от реализации. Существует полная по Тьюрингу вариация – *4DL+*.

3) Языки с литературным синтаксисом.

Chef – ЭЯП, программы на котором имитируют кулинарные рецепты.

Пример кода:

```
Hello World Souffle.

Ingredients.
72 g haricot beans
101 eggs
108 g lard
111 cups oil
32 zucchinis
119 ml water
114 g red salmon
100 g dijon mustard
33 potatoes

Method.
Put potatoes into the mixing bowl.
Put dijon mustard into the mixing bowl.
```

```
Put lard into the mixing bowl.
Put red salmon into the mixing bowl.
Put oil into the mixing bowl.
Put water into the mixing bowl.
Put zucchinis into the mixing bowl.
Put oil into the mixing bowl.
Put lard into the mixing bowl.
Put lard into the mixing bowl.
Put eggs into the mixing bowl.
Put haricot beans into the mixing bowl.
Liquefy contents of the mixing bowl.
Pour contents of the mixing bowl into the baking dish.

Serves 1.
```

Здесь *haricot beans, eggs* и прочие названия ингредиентов – имена переменных, количество, масса или объём ингредиентов в метрической системе – численные значения переменных (в данном случае – ASCII-код). Функция *put… into the mixing bowl* помещает соответствующее значение в стек, команда *Liquefy* говорит о том, что значения будут выводиться именно как Юникод-символы, а команда *Serve* выводит содержимое *baking dish* на устройство вывода и завершает программу.
Тьюринг-полнота: да.
Shakespeare – эзотерический ЯП с программами, имитирующими пьесы Шекспира.
Пример кода:

```
Romeo, a young man with a remarkable patience.
Juliet, a likewise young woman of remarkable grace.
Ophelia, a remarkable woman much in dispute with Hamlet.
Prince Hamlet, the flatterer of Andersen Insulting A/S.

                    Act I: Hamlet's insults and flattery.
                    Scene I: The insulting of Romeo.
[Enter Hamlet and Romeo]
Hamlet:
You lying stupid fatherless big smelly half-witted coward! You are as
stupid as the difference between a handsome rich brave hero and thyself!
Speak your mind!
You are as brave as the sum of your fat little stuffed misused dusty
old rotten codpiece and a beautiful fair warm peaceful sunny summer's
day. You are as healthy as the difference between the sum of the
sweetest reddest rose and my father and yourself! Speak your mind!
You are as cowardly as the sum of yourself and the difference
between a big mighty proud kingdom and a horse. Speak your mind.
Speak your mind!
[Exit Romeo]
```

Здесь представлен фрагмент программы, выводящей на экран “Hello, world!”. Переменные объявляются путём представления персонажей, где имя персонажа становится именем переменной, а описание игнорируется компилятором. Обращаясь к персонажу, другой персонаж задаёт его значение *(You are as… as…).*

Описывая друг друга, персонажи используют существительные и прилагательные из зарезервированного списка, которые могут быть положительными, отрицательными или нейтральными. Комбинация существительных и прилагательных разной эмоциональной окраски задаёт значения переменных, между которыми могут производиться и арифметические операции (*You are as cowardly as the **sum** of yourself and the **difference** between…)*. Команда *Speak your mind!* выводит переменную на устройство вывода.