Примерен тест

По

***Компютърна лингвистика***

При гл. ас. д-р ***Георги Петров Пашев***

За студенти от спец. ***Биоинформатика – Биологически факултет***

1. Напишете регулярен израз, който валидира дробни числа като 12.34, -12.3, +1.2.

**[\-\+]?(\d+) (\.\d+)?**

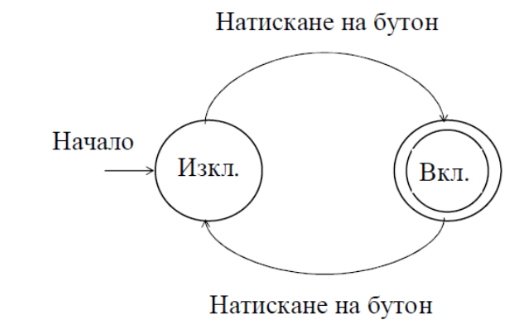
1. Напишете регулярен израз, който валидира имена (име, презиме, фамилия) в следните примерни формати: Иван Петров Петров; Георги П. Пашев; Драган К. Драганов; Драган Кон. Драганов.

**[А-Я][а-я]+ +[А-Я] (([а-я]+(\.)?)|( \.)) +[А-Я][а-я]+**

1. Напишете регулярен израз, който валидира цени в долари, като например: $ 12.34; USD12; USD 1234,45. Разделителя на цяла и дробна част да може да бъде както точка, така и запетая.

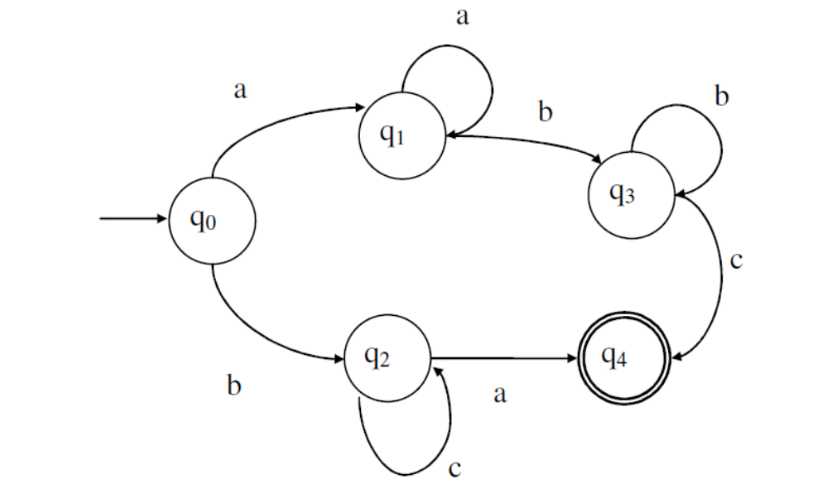
**([A-Z]{3}|\$) \*[\+\-]?(\d+)([\.,]\d{2})?**

1. Даден е следния краен автомат:



Определете: 1. Множеството от състояния; 2. Начално състояние; 3. Множеството от крайни състояния; 4. Азбуката от символи, които се разпознават от автомата.

1. **{изкл., вкл.};**
2. **Изкл.;**
3. **{вкл.};**
4. **{Натискане на бутон}**
5. Напишете таблицата на преходите на автомата, даден тук:



|  |  |  |  |
| --- | --- | --- | --- |
|  | а | b | c |
| Q0 | Q1 | Q2 |  |
| Q1 | Q1 | Q3 |  |
| Q2 | Q4 |  | Q2 |
| Q3 |  | Q3 | Q4 |
| Q4 |  |  |  |

1. Напишете поне 3 правилни за крайния автомат от миналия въпрос поредици от символи.

bca abc aabc

1. Дадена е контекстно свободната граматика:

S -> A A .

A -> a .

A -> b .

А -> a A .

Напишете поне 3 примерни поредици от символи, които са правилни спрямо тази граматика.

**aa bb ab aaaab**

1. Напишете контекстно независима граматика, която може да „парсне“ успешно адреси от вида. Терминалните символи може да ги опишете директно с регулярни изрази в граматиката във формат r”регулярен израз”:

|  |  |
| --- | --- |
| Георги Пашев  Пловдив 4000,  Ул. „Капитан Райчо“ №73А  Тел: 09009887899 | Ivan Draganov  Bulgaria  Sofia 1000  Bul. “Rakovski” 1  Tel: 09089000090 |

**S -> Name NL Town PK NL Address NL Phone**

**Name -> r”[А-Я][а-я]+ +[А-Я] (([а-я]+(\.)?)|( \.)) +[А-Я][а-я]+”**

**NL -> r”[,;]?\n”**

**Town -> r”[А-Я][а-я]+ [А-Я][а-я]+ +”**

**PK -> r”[0-9]{4}”**

**Address -> r“(Ul.|Bul.) +\”?[А-Я][а-я]+ +[А-Я] (([а-я]+(\.)?)|( \.)) +[А-Я][а-я]+ \”? \*№? \*\d+[A-ZА-Я]?”**

**Phone -> r”[0-9]{3,}”**

1. Дадена е следната база знания в PROLOG:

|  |
| --- |
| % is\_techer/1  is\_teacher(georgi\_pashev). is\_teacher(georgi\_totkov). is\_teacher(kasakliev).  % is\_student/1  is\_student(maria). is\_student(dragan). is\_student(ivan).  % teaches/2  teaches(georgi\_pashev, cl). teaches(georgi\_totkov, cl). teaches(kasakliev, ma ).  % studies/2  studies(dragan, cl). studies(ivan, cl). studies(maria, cl).  % teacher\_of/2  teacher\_of(X, Y) :- is\_teacher(X), is\_student(Y), teaches(X, Z), studies(Y, Z).  % teachers\_of/2  teachers\_of(Xs, Y) :- member(X, Xs), is\_teacher(X), is\_student(Y), teaches(X, Z), studies(Y, Z).  % not\_teaches/2  not\_teaches(X, Y) :- is\_teacher(X), \+ (teaches(X, Y)); is\_student(X).  % not\_is\_student/1  not\_is\_student(X) :- is\_teacher(X). |

* 1. Напишете заявка, която връща всички студенти на georgi\_pashev.

**teacher\_of(georgi\_pashev, X)**

* 1. Напишете заявка, която връща предметите, които изучава dragan.

**studies(dragan, X)**

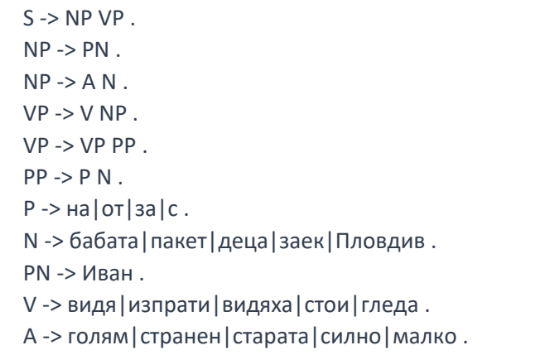
* 1. Напишете правило и заявка по него, които ще върнат в променливи (X, Y) всички наредени двойки X = преподавател, Y = студент, за дисциплина cl.

**% teacher\_of/3**

**teacher\_of(X, Y, Z) :- is\_teacher(X), is\_student(Y), teaches(X, Z), studies(Y, Z).**

**teacher\_of(X, Y, cl)**

1. Дадена е следната контекстно независима граматика:



* 1. Напишете примерно изречение, което да отговаря на нея.

**Иван видя Иван.**

* 1. Отговаря ли на граматиката изречението: „**голям пакет видя странен бабата**“.

**да**

* 1. Отговаря ли на граматиката изречението: **„голям Иван гледа за Пловдив**“. Ако не, какво правило трябва да добавите в граматиката, за да започне да отговаря.

**не**

**NP -> A PN .**

1. Даден е така описания в Python 3.8 детерминиран краен автомат:

begin\_state = "q0"  
end\_states = {"q4": "1"}  
  
FSM = {  
 "q0": {"a": "q1", "b": "q2"},  
 "q1": {"a": "q1", "b": "q3"},  
 "q2": {"c": "q2", "a": "q4"},  
 "q3": {"b": "q3", "c": "q4", "d": None}  
 }

и следната функция, която парсва стринг по него:

def parseFSM(text, FSM, beginState, endStates):  
 cur\_state = beginState  
 try:  
 for i in range(0, len(text)):  
 if FSM[cur\_state][text[i]] is not None:  
 cur\_state = FSM[cur\_state][text[i]]  
 else:  
 return False  
 if endStates[cur\_state] is not None:  
 return True  
 else:  
 return False  
 except KeyError as e:  
 print({"cur\_state": cur\_state, "exception": e, "text\_letter": text[i]})  
 return False

11.1. Какво ще върне функцията parseFSM при следното извикване:

print(parseFSM("bac", FSM, begin\_state, end\_states))

**false**

11.2. Какво ще върне функцията parseFSM при следното извикване:

print(parseFSM("aaabc", FSM, begin\_state, end\_states))

**true**

1. Даден е регулярният израз:

*r'[\-+]?\d+(\.\d+)?'*

* 1. Какво „парсва“/валидира той?

**Валидира отрицателни или положителни цели или дробни числа**

* 1. Променете го, така, че да позволява не само десетична точка, но и запетая.

*r'[\-+]?\d+([\.,]\d+)?'*

* 1. Задължително ли валиден текст трябва да започва с + или -?

**не**

* 1. Добавете накрая на регулярния израз незадължителна валута, състояща се от 3 големи букви на латиница, например: BGN, USD, … .

*r'[\-+]?\d+([\.,]\d+)? \*[A-Z]{3}'*

1. Какво прави лематизатора в морфологичния анализатор?

**Преобразува дума в основна форма.**

1. Подредете по сложност/ниво на абстрактност следните машини от по-проста към по-сложна, като 1 е най-проста, а 3 е най-сложна:

|  |  |
| --- | --- |
| **Машина на Тюринг** | 3 |
| **Регулярен израз** | 1 |
| **Контекстно независима граматика** | 2 |