Лабораторная работа №7 по информатике: 8 факультет, 1 курс, 1 семестр 2011/12 уч. года
Программирование в алгоритмической модели Маркова

Краткое описание интерпретатора алгоритмов Маркова

Интерпретатор алгоритмов Маркова **nam** функционирует в лабораторной среде UNIX. Вызов **nam** осуществляется из среды интерпретатора команд следующим образом:

nam ***[-m]*** *<имя-файла-с-НАМ>.nam*

Параметр ***-m*** включает интерпретацию метасимволов алгоритма. Метасимволы задаются заглавными буквами (A, B, C) и воспринимаются как любые буквы алфавита. Тем самым метасимволы задают группы однотипных правил.

В данной системе алгоритмы хранятся в текстовых файлах с расширением ***.nam***, создаваемых с помощью любого доступного редактора текстов. Каждая продукция должна располагаться на отдельной строке. Символы $\rightarrow$ и $|\rightarrow$ представляются составными знаками (словами) **->** и **->.** соответственно. Начальные и конечные пробелы слов в левой и правой части правил игнорируются. Протокол работы **nam** помещается в текстовый файл **nam.out**, создаваемый (или перезаписываемый) в текущей директории.

На алгоритмы Маркова, выполняемые **nam**, налагаются следующие ограничения:

- В состав алгоритма может входить до 62 продукций. Алгоритмы, не удовлетворяющие этому требованию, не интерпретируются.
- Не рекомендуется использовать правила, значащая длина слов в левой и правой части которых превышает 9 знаков.
- Длина входного сообщения и всех сообщений, полученных в результате выполнения подстановок, не может превышать 80. При нарушении этого требования интерпретация текущего сообщения прерывается. После этого возможна обработка другого входного сообщения.

Во время работы **nam** экран терминала имеет следующую структуру:

- в двух верхних строках вводится входное сообщение и отображается текущее обрабатываемое сообщение;
- в третьей строке, выделенной инверсным отображением, выводятся диагностики и предупреждения системы, а также подсказки по горячим клавишам;
- в строках 4–24 в несколько столбцов отображается текст алгоритма. Текущая применяемая продукция помечается слева знаком **>**.

После ввода входного сообщения начинается его пошаговая обработка. При этом пользователь может ввести одну из следующих команд, нажав соответствующую комбинацию клавиш:

	ENTER/ RETURN	Продолжение пошаговой интерпретации
R	ENTER/ RETURN	Переход к интерпретации текущего сообщения без остановок
S	ENTER/ RETURN	Прерывание обработки текущего сообщения
	CTRL-C	Прерывание работы интерпретатора

Для *домашних работ* в среде MS Windows могут быть использованы другие программы, входящие в состав ***e***-хрестоматии к курсу: NAM (автор Ганущак А. А.) , мультимодельную систему Рыбакова К. А., а также системы программирования на РЕФАЛЕ (уточнить у доц. Левинской М. А.).

Пример алгоритма Маркова, и программы на РЕФАЛе, добавляющих единицу к *изображению* двоичного числа:

```
>1    ->    1>
>0    ->    0>
>     ->    <
1<    ->    <0
0<    ->.   1
<     ->.   1
```

```
add {
      e.1 '1' = <add e.1> '0';
      e.2 '0' = e.2 '1';
      = '1';
    }
```

```
        ->    >
```

Этот же алгоритм с использованием метасимволов (проф. Титов В.К.):

```
>A    ->    A>
>     ->    <
1<    ->    <0
0<    ->.   1
<     ->.   1
      ->    >
```

Ещё один пример программирования в марковской модели (доц. Левинская М.А.) показывает, что алгоритм дифференцирования записывается на РЕФАЛе простой транслитерацией соответствующих математических правил :

```
diff {
       e.1 '+' e.2 = <diff e.1> '+' <diff e.2>;
       e.1 '*' e.2 = <diff e.1> '*' e.2 '+' <diff e.2> '*' e.1;
       'x' = '1';
       e.1 , <Type e.1>: 'D' e.2 = '0';
     }
```

Другие примеры НАМ приведены в конспекте лекций по курсу. Там же имеются ссылки на литературу по НАМ.

Интерпретатор **nam** разработан доц. Журавлёвой Т.Э. в 1995 г. на MicroVAX II в среде Ultrix 32.

В настоящее время на лабораторном сервере тестирования DEC Alpha в среде NetBSD (AXP4) доступна *только* экспериментальная версия интерпретатора НАМ со встроенным редактором алгоритмов. Метасимволы в этой версии не поддерживаются!

"Всякое слово есть число, всякое число есть слово."
Основной закон Каббалы

Варианты заданий (* отмечены задачи повышенной сложности)

0. Кодирование числа в римской записи по Цезарю в алфавите {I,V,X,L,C,D,M} (разбирается на занятии).
1. * Входное слово представляет собой два троичных числа без знака, разделенные знаком "+". Составить алгоритм вычисления суммы этих чисел.
2. * Входное слово представляет собой два троичных числа без знака, разделенные знаком "-". Составить алгоритм вычисления разности этих чисел.
3. * Входное слово представляет собой троичное число без знака. Составить алгоритм копирования числа. Результат должен состоять из исходного слова и его копии, разделенных знаком "=".
4. * Входное слово представляет собой троичное число без знака. Составить алгоритм реверса (инвертирования) числа (записи его цифр в обратном порядке).
5. Составить алгоритм перевода числа из троичной системы счисления в девятиричную.
6. Составить алгоритм перевода числа из девятиричной системы счисления в троичную.
7. * Входное слово представляет собой два троичных числа без знака, разделенные знаком "<". Составить алгоритм вычисления троичного логического сдвига первого числа влево на число разрядов второго числа.
8. * Входное слово представляет собой два троичных числа без знака, разделенные знаком ">". Составить алгоритм вычисления троичного логического сдвига первого числа вправо на число разрядов второго числа.
9. * Входное слово представляет собой два двоичных числа без знака, разделенные знаком "<". Составить алгоритм вычисления двоичного логического сдвига второго числа влево на число разрядов, равное первому числу.
10. * Входное слово представляет собой два двоичных числа без знака, разделенные знаком ">". Составить алгоритм вычисления двоичного логического сдвига второго числа вправо на число разрядов, равное первому числу.
11. ** Входное слово представляет собой два двоичных числа без знака, разделенные знаком "*". Составить алгоритм вычисления двоичного арифметического сдвига второго числа влево на число разрядов, равное

первому числу.

12. ** Входное слово представляет собой два двоичных числа без знака, разделенные знаком "/". Составить алгоритм вычисления двоичного арифметического сдвига второго числа вправо на число разрядов, равное первому числу.
13. ** Входное слово представляет собой два двоичных числа без знака, разделенные знаком "~". Составить алгоритм вычисления двоичного циклического сдвига второго числа влево на число разрядов первого числа.
14. ** Входное слово представляет собой два двоичных числа без знака, разделенные знаком "~". Составить алгоритм вычисления двоичного циклического сдвига второго числа вправо на число разрядов первого числа.
15. ** Входное слово представляет собой два двоичных числа без знака, разделенные знаком "$". Составить алгоритм выделения разрядов первого числа по маске, в качестве которой используется второе число.
16. ** Входное слово представляет собой два двоичных числа без знака, разделенные знаком "$". Составить алгоритм выделения разрядов второго числа по маске, в качестве которой используется первое число.
17. * Входное слово представляет собой два двоичных числа без знака, разделенные знаком "&". Составить алгоритм вычисления поразрядной конъюнкции исходных чисел.
18. * Входное слово представляет собой два двоичных числа без знака, разделенные знаком "|". Составить алгоритм вычисления поразрядной дизъюнкции исходных чисел.
19. * Входное слово представляет собой произвольную последовательность десятичных чисел без знака, разделенных знаками "#". Составить алгоритм вычисления числа слов в последовательности.
20. * Входное слово представляет собой два троичных числа без знака, разделенные знаком "^". Обменять числа местами.
21. ** Входное слово представляет собой два двоичных числа без знака, разделенные знаком "%". Вычислить наибольший общий делитель исходных чисел.
22. Входное слово представляет собой десятичную запись целого неотрицательного числа в прямой кодировке. Получить дополнительную кодировку для отрицательного числа с тем же абсолютным значением.
23. Входное слово представляет собой десятичную запись целого неотрицательного числа в прямой кодировке. Получить обратную кодировку для отрицательного числа с тем же абсолютным значением.
24. ** Составить алгоритм умножения двух неотрицательных целых чисел в алфавите {|}.
25. Составить алгоритм увеличения на единицу целого неотрицательного числа в шестнадцатиричной позиционной системе счисления.
26. Составить алгоритм уменьшения на единицу целого неотрицательного числа в шестнадцатиричной позиционной системе счисления.
27. Составить алгоритм, восстанавливающий целое число в шестнадцатиричной позиционной системе счисления по его дополнительному коду.
28. Составить алгоритм, восстанавливающий целое число в шестнадцатиричной позиционной системе счисления по его обратному коду.
29. ** Составить алгоритм натурализации десятичного числа в позиционной записи (перевода в единичную систему счисления {|}).
30. Входное слово представляет собой два двоичных числа без знака, разделенных символом "&&". Составить алгоритм вычисления логического произведения (&& в Си) исходных чисел.
31. Составить алгоритм кодирования слова в латинском алфавите по Цезарю.
32. ** Входное слово представляет собою последовательность латинских букв, за которой следует двоичное число. Составить алгоритм кодирования первого слова по коду Цезаря с ключом, равным второму слову.
33. Составить алгоритм перевода числа из четверичной системы счисления в шестнадцатиричную.
34. Составить алгоритм перевода числа из шестнадцатиричной системы счисления в четверичную.
35. Составить алгоритм двоичного подсчёта числа гласных в слове латинского алфавита.
36. ** Составить алгоритм подсчета в натуральной системе числа чётных членов в последовательности десятичных чисел, разделенных “**;**”.
37. ** Составить алгоритм возведения числа в натуральной записи в квадрат.
38. ** Составить алгоритм возведения числа в кардинальной записи в квадрат
39. Составить алгоритм перевода десятичных цифровых сообщений в азбуку Морзе.
40. * Составить алгоритм вычисления двоичного числа – двоичного логарифма двоичного числа.
41. * Составить алгоритм вычисления троичного числа – троичного логарифма троичного числа.
42. *** Составить алгоритм проверки делимости десятичного числа на 11.

Варианты заданий составлены доц. Журавлевой Т.Э., проф. Зайцевым В.Е. и доц. Сошниковым Д.В.