ТЕМЫ И ПЛАНЫ ПРАКТИЧЕСКИХ ЗАНЯТИЙ

Модуль 1. Машина Тьюринга.

1. Машины Тьюринга (4 часа):

– решение задач на обработку нечисловых данных;

– решение задач на обработку числовых данных (в унарной и десятичной системах счисления);

– создание циклических машин Тьюринга.

2. Эмулятор МТ (2 часа).

Модуль 2. Примитивно-рекурсивные функции. Нормальные алгоритмы Маркова.

3. Рекурсивные функции (4 часа):

– доказательство примитивной рекурсивности функций от одной и нескольких переменных;

– восстановление функций по схеме примитивной рекурсии.

4. Нормальные алгоритмы Маркова (2 часа):

– решение задач на работу с числовыми и нечисловыми объектами;

– доказательство нормальной вычислимости функций.

5. Эмулятор НАМ (2 часа).

Модуль 3. Машины с неограниченными регистрами.

6. Машина с неограниченными регистрами (2 часа):

– создание алгоритмов, работающих с нечисловыми объектами;

– создание алгоритмов, работающих с числовыми объектами.

7. Эмулятор МНР (2 часа).

Модуль 4. Вычислимость и разрешимость.

8. Нумерация программ (2 часа).

Модуль 5. Эффективные операции на вычислимых функциях. Сложность вычисления.

9. Машина Поста (2 часа).

10. Эмулятор МП (2 часа).

МОДУЛЬ 1. МАШИНЫ ТЬЮРИНГА

Практическое занятие 1

Машины Тьюринга (МТ)

Машина Тьюринга разработана Аланом Тьюрингом в 1936 г.

Машина Тьюринга состоит из:

1) управляющего устройства, которое может находиться в одном из состояний $Q = \{q_1, q_2, \ldots, q_n\}$;

2) ленты, бесконечной в обе стороны, разбитой на ячейки, в каждой из которых может быть записан один из символов конечного алфавита $A = \{a_1, a_2, \ldots, a_m\}$;

3) устройства обращения к ленте, считывающей и пишущей головки, которая в каждый момент времени обозревает ячейку ленты и в зависимости от символа в этой ячейке записывает новый символ, стирает этот или оставляет прежним и переходит в новое состояние или оставляет прежнее состояние.

Память МТ – конечное множество состояний (внутренняя память) и лента (внешняя память).

Данные МТ – слова в алфавите ленты.

Элементарные шаги МТ – считывание и запись символов, сдвиг головки вправо или влево, переход управляющего устройства в следующее состояние.

Детерминированность МТ – последовательность ее шагов определяется: для любого состояния q_i и символа a_j однозначно заданы:

1) q_i' – состояние;

2) символ a_j', который надо записать вместо a_j;

3) d_k – направление сдвига головки (может принимать значения R, L, E).

Машину Тьюринга можно записывать в виде:

– системы правил (команд);

– таблицы (строки – состояния; столбцы – символы; на пересечении – новое состояние, символ и направление сдвига головки);

– диаграммы (или графа).

Пример. Сложение. Во введенном раннее представлении чисел сложить числа a и b – это значит слово $1^a * 1^b$ переработать в слово 1^{a+b}, т.е. удалить разделитель * и сдвинуть одно из слагаемых, скажем, первое, к другому. Это осуществляет машина T_+ с четырьмя состояниями и следующей системой команд (первая команда введена для случая, когда а = 0 и исходное слово имеет вид $* 1^b$):

$$q_1{*}\to q_z\lambda R;$$
$$q_1 1\to q_2\lambda R;$$
$$q_2 1\to q_2 1R;$$
$$q_2{*}\to q_3 1L;$$
$$q_3 1\to q_3 1L;$$
$$q_3\lambda\to q_z\lambda R.$$

Задания

1. Дан алфавит A = {1, 0} и состояния Q = $\{q_1\}$. Построить машину Тьюринга, удаляющую 0.
2. Построить машину Тьюринга T_+ для сложения двух натуральных чисел, записанных в унарной системе. Построить диаграмму машины Тьюринга T_+ , A = {1}.
3. Дан алфавит A={1}. Построить машину Тьюринга для получения следующего натурального числа.
4. Имеется машина Тьюринга с алфавитом A={1} и внутренними состояниями Q = $\{q_1\}$ со следующей системой команд:

$$q_1 0 \to q_z 1$$
$$q_1 1 \to q_1 1R.$$

Определить, в какое слово переработает машина каждое из следующих слов, если она находится в начальном состоянии q_1:

a) $10q_1110011$;

b) $110q_111101$;

c) $1q_100111$.

Ответ изобразить схематически в виде последовательности конфигураций, возникающих на ленте на каждом такте работы машины.

5. Дано слово в алфавите A = {a, b, c, d}. Построить машину Тьюринга, удаляющую все буквы a.

6. Построить машину Тьюринга T_{++} для суммирования N натуральных подряд идущих чисел, записанных в унарной системе через «*». Пример: 111*11*1111* … *11.

7. Построить машину Тьюринга T_, которая вычисляет разность двух натуральных чисел, записанных в унарной системе через разделитель. Например: 11111-111.

Индивидуальное задание 1

Машины Тьюринга

1. Имеется машина Тьюринга с внешним алфавитом A = {0, 1} и внутренними состояниями Q = $\{q_1\}$ со следующей системой команд:

$$q_1 1 \to q_1 0R$$
$$q_1 0 \to q_1 1R$$
$$q_1 \lambda \to q_z 1.$$

Определить, в какое слово переработает машина слово α, если она находится в начальном состоянии q_1. Ответ изобразить схематически в виде последовательности конфигураций, возникающих на ленте на каждом такте работы машины.

1) $11q_101\lambda01$;	4) $11q_101\lambda01$;
2) $100q_11\lambda01$;	5) $100q_11\lambda01$;
3) $11\lambda01q_101$;	6) $11\lambda01q_101$;

7) $q_1 10\lambda 1100$;	16) $1q_1 111111$;
8) $1101q_1 001$;	17) $110q_1 1\lambda 01$;
9) $11q_1 01101$;	18) $0q_1 000\lambda 01$;
10) $1q_1 1010\lambda 1$;	19) $100q_1 1\lambda 01$;
11) $10011q_1 01$;	20) $0\lambda 1q_1 0101$;
12) $1\lambda 1q_1 0101$;	21) $11q_1 010\lambda 1$;
13) $110q_1 \lambda 101$;	22) $0111q_1 01\lambda$;
14) $\lambda 11q_1 0101$;	23) $1\lambda q_1 0110\lambda$;
15) $110q_1 1\lambda 01$;	24) $110q_1 1001$

2. Дана машина Тьюринга с внешним алфавитом A = {1, 0} и внутренними состояниями Q = {q_1, q_2, q_3, q_4}, записанная в виде таблицы:

A \ Q	q_1	q_2	q_3	q_4
λ	$q_2\lambda R$	$q_3\lambda R$	$q_4\lambda R$	$q_z\lambda R$
1	$q_1\lambda R$	$q_3 1R$	$q_4\lambda R$	$q_4 1L$
0	$q_1 1R$	$q_3 0R$	$q_4 1R$	$q_1 0L$

Изображая на каждом такте работы машины получающуюся конфигурацию, определите, в какое слово перерабатывает машина слово α, исходя из положения, когда машина находится в состоянии q_1.

1) $1101\lambda 0q_1 1\lambda 011$;	8) $110q_1 \lambda 01\lambda 1001$;
2) $10\lambda 0q_1 10\lambda\lambda 101$;	9) $11011q_1 \lambda 01\lambda 01$;
3) $11\lambda q_1 \lambda 01\lambda 0101$;	10) $10\lambda 0q_1 110\lambda\lambda 1$;
4) $1001\lambda q_1 011\lambda 0\lambda$;	11) $1001\lambda 01q_1 \lambda 101$;
5) $110\lambda 01\lambda q_1 \lambda 101$;	12) $1\lambda 10q_1 101\lambda 01\lambda$;
6) $110\lambda q_1 01\lambda 1\lambda 01$;	13) $11q_1 0\lambda 01\lambda\lambda 101$;
7) $10\lambda 11\lambda 0q_1 1\lambda 00$;	14) $\lambda 110q_1 \lambda 01\lambda 101$;

15) 110λ01λ$q_1$1λ01;	20) 0λλ0$q_1$1λ10101;
16) 11λ01λ$q_1$11111;	21) 1λ1011λ1λ$q_1$101;
17) 11$q_1$01λ01λλ01;	22) 00λ$q_1$11λ1λ011;
18) 00λ01λ$q_1$00λ01;	23) λ110q_1λ01101λ;
19) 1001λλ01λ$q_1$01;	24) 1001λ10q_1λλ11

3. Постройте машину Тьюринга, которая бы в слове *cabdabda* выполнила указанные действия. Исполнить полученный алгоритм.

1) заменяла *ab* на *e*	13) добавляла *s* до *b*
2) добавляла *e* после *b*	14) сдвигала на 1 символ влево
3) добавляла *f* до *a*	15) сдвигала на 1 символ вправо
4) сдвигала на 2 символа влево	16) заменяла *da* на k
5) сдвигала на 2 символа вправо	17) добавляла *e* после *a*
6) заменяла *b* на *f*	18) добавляла *c* до *d*
7) добавляла *c* после *a*	19) переносила 2 символа слева в конец слова
8) меняла местами 1 символ и последний	20) заменяла *db* на *e*
9) переносила 2 символа справа в начало слова	21) добавляла *bc* после *b*
10) меняла местами 2-й символ и предпоследний	22) заменяла *a* на *p*
11) заменяла *d* на *a*	23) добавляла *e* после *b*
12) добавляла *f* после *c*	24) упорядочивала последовательность

Практическое занятие 2

Машины Тьюринга

1. Построить машину Тьюринга, которая правильно вычисляет функцию $f(x) = x + 1$ (число записано в двоичной системе счисления).

2. Построить машину Тьюринга, которая правильно вычисляет функцию $f(x) = 0$, А = {1, 0}.

3. Построить машины Тьюринга, которые в алфавите А = {1, 0} выполняют:

1) правый сдвиг: $q_1 01^x 0 \underset{Б^+}{\Rightarrow} 01^x q_z 0$;

2) левый сдвиг (самостоятельно): $01^x q_1 0 \underset{Б^-}{\Rightarrow} q_z 01^x 0$;

3) транспозицию: $01^x q_1 01^y 0 \underset{В}{\Rightarrow} 01^y q_z 01^x 0$;

4) удвоение: $q_1 01^x 0 \underset{Г}{\Rightarrow} q_z 01^x 01^x 0$.

Индивидуальное задание 2

Машины Тьюринга. Конструирование машин Тьюринга

1. Дан алфавит А = {a, b, c, d, e, f} и внутренние состояния Q = {q_1, q_2, …, q_z}. Построить машину Тьюринга для транспозиции элементов. Например, последовательность *abcd* заменить на *dcba*.

2. Построить машину Тьюринга, вычисляющую функцию:

$$f(x_1, x_2, \ldots, x_n) = x_m.$$

Пример: 1110110…011. Оставить *m*-й набор единиц, остальные стереть.

3. Дана конечная совокупность единиц, вписанных в ячейки без пропусков. Построить машину Тьюринга, которая записывала бы в десятичной системе счисления число этих единиц, т.е. пересчитывала набор этих единиц.

4. Построить машину Тьюринга, которая выполняет умножение двух чисел в унарной системе счисления.

5. Даны два набора единиц. Они разделены *. Построить машину Тьюринга, которая выбирала бы больший из этих наборов, а меньший стирала.

6. Дана строка из букв «a» и «b». Разработать машину Тьюринга, которая переместит все буквы «a» в левую, а буквы «b» – в правую части строки. Каретка находится над крайним левым символом строки.

7. На ленте машины Тьюринга находится целое положительное число, записанное в десятичной системе счисления. Найти произведение этого числа на число 11. Каретка обозревает крайнюю правую цифру числа.

8. На ленте машины Тьюринга находится десятичное число. Определить, делится ли это число на 5 без остатка. Если делится, то записать справа от числа слово «y», если нет – «n».

9. На ленте машины Тьюринга записано число в пятеричной системе счисления. Каретка находится над крайней правой цифрой. Записать цифры этого числа в обратном порядке.

10. Даны два натуральных числа n и m, представленные в унарной системе счисления. Между этими числами стоит знак «*». Построить машину Тьюринга, определяющую, равны эти числа или нет.

11. Даны два натуральных числа n и m, представленные в двоичной системе счисления. Между этими числами стоит знак «*». Найти разность этих чисел.

12. Дано число в двоичной системе счисления. Разработать машину Тьюринга, которая будет умножать это число на два (в двоичной системе счисления со сдвигом влево).

13. Сконструировать машину Тьюринга, которая выступит в качестве двоично-восьмеричного дешифратора.

14. Даны два натуральных числа n и m, заданных в унарной системе счисления. Числа n и m представлены наборами символов «1», разделенных «/». В конце набора стоит знак « = ». Разработать машину Тьюринга, которая будет производить деление нацело двух натуральных чисел n и m и находить остаток от деления.

15. Построить машину Тьюринга, которая выполняет деление на три в унарной системе счисления.

16. Дан алфавит A = {a, b, c, …, z} и внутренние состояния Q = {q_1, q_2, …, q_z}. Построить машину Тьюринга, которая подсчитывает количество букв *a* в заданной последовательности.

17. Даны два целых положительных числа в двоичной системе счисления. Разработать машину Тьюринга, которая будет находить сумму этих чисел.

18. На ленте машины Тьюринга записано число в двоичной системе счисления. Каретка находится над крайней правой цифрой. Записать цифры этого числа в обратном порядке.

19. Дано натуральное число *n*, представленное в унарной системе счисления. Выяснить, является это число четным или нечетным.

20. Сконструировать машину Тьюринга, которая выступит в качестве четверично-двоичного дешифратора.

21. На ленте машины Тьюринга в произвольном порядке записаны 4 буквы *k, l, m, n*. Каретка обозревает крайнюю левую букву. Необходимо построить машину Тьюринга, которая расположит эти буквы по алфавиту.

22. На ленте машины Тьюринга в трех секциях в произвольном порядке записаны 4 цифры 3, 5, 7, 9. Каретка обозревает крайнюю левую цифру. Необходимо построить машину Тьюринга, которая расположит эти цифры в порядке убывания.

23. Дан алфавит A = {+, =, 0, 1} и внутренние состояния Q = {q_1, q_2, …, q_z}. Построить машину Тьюринга, проверяющую, является ли исходная последовательность арифметическим выражением.

24. Дан алфавит A = {(,)} и внутренние состояния Q = {q_1, q_2, …, q_z}. Построить машину Тьюринга, проверяющую, верно ли в исходной последовательности расставлены скобки.

Практическое занятие 3

Работа с эмулятором МТ

Знакомство с эмулятором «Машина Тьюринга»

1. Запустите эмулятор «ТМ»:

2. Познакомьтесь с интерфейсной частью программы, рассмотрите возможности меню (рис. 1.1);

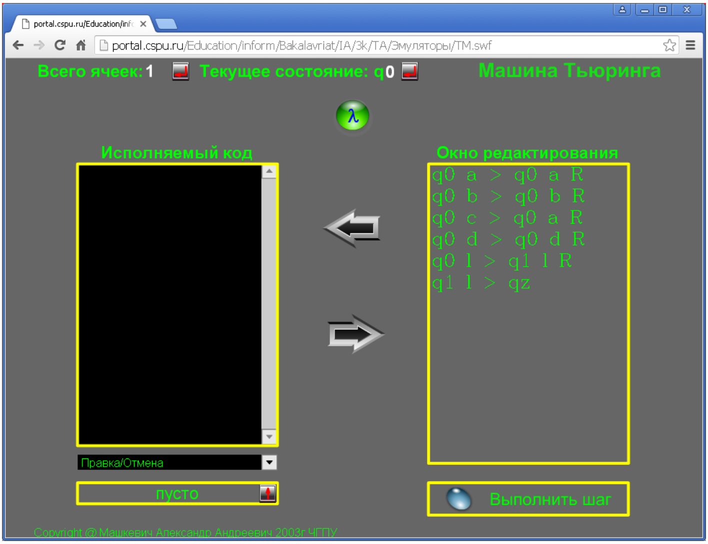

Рис. 1.1. Внешний вид эмулятора МТ

Задание 1. На ленте машины Тьюринга содержится последовательность символов “+”. Напишите программу для машины Тьюринга, которая каждый второй символ “+” заменит на “–”. Замена начинается с правого конца последовательности. Автомат в состоянии q_1 обозревает один из символов указанной последовательности.

В состоянии q_1 машина ищет правый конец числа, в состоянии q_2 – пропускает знак “+”, при достижении конца последовательности – останавливается. В состоянии q_3 машина знак “+”

заменяет на знак “–”, при достижении конца последовательности она останавливается.

Задание 2. Требуется построить машину Тьюринга, которая прибавляет единицу к числу на ленте. Входное слово состоит из цифр целого десятичного числа, записанных в последовательные ячейки на ленте. В начальный момент машина находится против самой правой цифры числа.

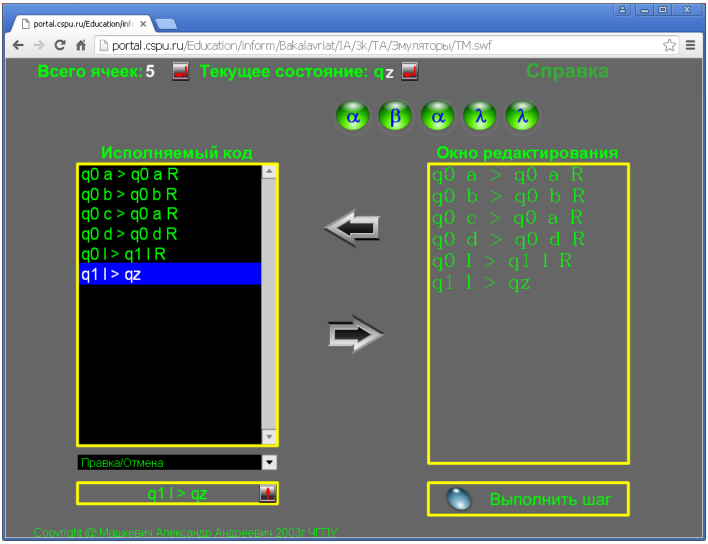

Рис. 1.2. Пример вычисления

Решение. Машина должна прибавить единицу к последней цифре числа. Если последняя цифра равна 9, то ее нужно заменить на 0 и прибавить единицу к предыдущей цифре. В этой машине Тьюринга q_1 – состояние изменения цифры, q_z – состояние останова. Если в состоянии q_1 автомат видит цифру 0..8, то он заменяет ее на 1..9 соответственно и переходит в состояние q_z, т.е. машина останавливается. Если же он видит цифру 9, то заменяет ее на 0, сдвигается влево, оставаясь в состоянии q_1. Так продолжается до тех пор, пока автомат не встретит цифру меньше 9. Если же все цифры были равны 9, то он заменит их нулями, запишет 0 на месте старшей цифры, сдвинется влево и в пустой клетке запишет 1. Затем перейдет в состояние q_z, т.е. остановится.

Задание 3. Дан массив из открывающих и закрывающих скобок. Построить машину Тьюринга, которая удаляла бы пары взаимных скобок, т.е. расположенных подряд “( )”.

Например, дано “) (( ) (( )”, надо получить “) (. . (”. Автомат в состоянии q_1 обозревает крайний левый символ строки.

Состояние q_1: если встретили “(”, то сдвиг вправо и переход в состояние q_2. Состояние q_2: анализ символа “(” на парность, в случае парности должны увидеть “)”. Если парная, то возврат влево и переход в состояние q_3. Состояние q_3: стираем сначала “(”, затем “)” и переходим в q_1.

Самостоятельно:

Задание 1. На ленте машины Тьюринга записаны два числа в унарной системе счисления, разделенные *. Построить машину Тьюринга, которая выполнит вычитание. Первое число больше второго.

Задание 2. Дан алфавит A = {0, 1}. Построить машину Тьюринга, которая подсчитывает количество 1.

Вопросы к модулю 1

1. Что такое «алгоритм»?
2. Перечислите и охарактеризуйте основные свойства алгоритма.
3. Какие существуют подходы к уточнению понятия алгоритма?
4. Почему возникла необходимость уточнить данное понятие?
5. Какая функция называется вычислимой, эффективно вычислимой?
6. Чем отличается описание алгоритма от механизма его реализации?
7. Что собой представляет блок-схема?

8. Как называются вершины в блок-схеме?

9. Можно ли записать несколько алгоритмов с помощью блок-схем?

10. Дайте характеристику блок-схем (назначение, свойства и др.).

11. Что называется декартовым произведением множеств?

12. Что называется областью определения функции? множеством значений?

13. Какая функция называется сюръективной? инъективной? биективной?

14. Что является ограничением функции?

15. Дать определение прообраза, композиции.

16. Какая пара объектов a и b называется неупорядоченной?

17. Какие отношения называются бинарными?

18. Какие бинарные отношения называются транзитивными, рефлексивными, симметричными, отношениями эквивалентности?

19. Что такое «класс эквивалентности»?

20. Какое бинарное отношение называется частичным порядком?

21. Перечислить свойства частичного порядка.

22. Сформулировать теорему Бома и Джакопини.

23. Охарактеризовать основные управляющие структуры.

24. Дайте определение машины Тьюринга.

25. Охарактеризовать каждую составляющую машины Тьюринга.

26. Из чего состоит память машины Тьюринга?

27. Бесконечна ли лента в одну сторону? в обе?

28. Сколько символов можно записать в одну ячейку?

29. Конечно ли множество ячеек, заполненных на ленте в любой момент времени?

30. Что обозначают следующие символы:

a) λ; b) q_1; c) q_z?

31. Что происходит с машиной Тьюринга, когда она попадает в состояние q_z?

32. Дать определение функции вычислимой по Тьюрингу.

33. Как выполняются требования к алгоритмам на примере МТ?

34. Сформулировать проблему остановки. Смысл?

35. Существует ли универсальная МТ?

36. Существует ли универсальная МТ с двумя состояниями и двумя символами на ленте?

МОДУЛЬ 2. ПРИМИТИВНО-РЕКУРСИВНЫЕ ФУНКЦИИ. НОРМАЛЬНЫЕ АЛГОРИТМЫ МАРКОВА

Практическое занятие 4

Примитивно-рекурсивные функции

Теория рекурсивных функций появилась в 30-х годах прошлого века. В этой теории, как и вообще в теории алгоритмов, принят конструктивный (финитный) подход, основной чертой которого является то, что все множество исследуемых объектов (в данном случае функций) строится из конечного числа конечных объектов – базиса с помощью простых операций, эффективная вычислимость которых достаточно очевидна. Операции над функциями будем в дальнейшем называть операторами.

В базис включим: константу 0; функцию следования $x' = x+1$; функцию проекции $U_m^n(x_1, x_2, ..., x_n) = x_m \, (m \le n)$.

Оператором суперпозиции S_m^n называется подстановка в функцию от m переменных m функций от n одних и тех же переменных:

$$S_m^n(h, g_1, ..., g_m) = h(g_1(x_1, ..., x_n), ..., g_m(x_1, ..., x_n)) = f(x_1, ..., x_n).$$

Суперпозиция дает новую функцию от n переменных.

Таким образом, если заданы функции U_m^n и операторы S_m^n, то можно считать заданными всевозможные операторы подстановки функций в функции, а такие переименования, перестановки и отождествления переменных.

Пример 1.

$$f(x_2, x_1, x_3, ..., x_n) = f(U_2^2(x_1, x_2), U_1^2(x_1, x_2), x_3, ..., x_n);$$

$$f(x_1, x_1, x_3, ..., x_n) = f(U_1^2(x_1, x_2), U_1^2(x_1, x_2), x_3, ..., x_n).$$

Это определение порождает семейство операторов суперпозиции $\{S_m^n\}$. Благодаря функциям проекции стандартизация суперпозиции не уменьшает ее возможностей: любую подстановку функций в функцию можно выразить через S_m^n, U_m^n.

Оператор примитивной рекурсии R_n определяет $(n+1)$-местную функцию f через n-местную функцию g и $(n+2)$-местную функцию h так:

$$\begin{cases} f(x_1, ..., x_n, 0) = g(x_1, ..., x_n); \\ f(x_1, ..., x_n, y+1) = h(x_1, ..., x_n, y, f(x_1, ..., x_n, y)). \end{cases}$$

Функция называется примитивно-рекурсивной, если она может быть получена из константы 0, функции x′ и функции   с помощью конечного числа применений операторов суперпозиции и примитивной рекурсии.

Пример 2. Сложение $f_+(x, y) = x + y$ примитивно-рекурсивно:

$$f\ (x,0) = x = U_1^1(x);$$

$$f\ (x, y+1) = f_+(x, y) + 1 = (f_+(x, y))'$$

Задание 1. Доказать, что если функция $f(x_1, x_2, ..., x_n)$ примитивно-рекурсивная, то следующие функции примитивно-рекурсивны:

а) $g(x_1, x_2, x_3, ..., x_n) = f(x_2, x_1, x_3, ..., x_n)$ – перестановка аргументов;

б) $\psi(x_1, x_2, \ldots, x_n) = f(x_2, \ldots, x_n, x_1,)$ – циклическая перестановка аргументов.

Задание 2. Доказать, что следующие функции примитивно-рекурсивны:

а) $f(x) = x + 2$;

б) $f(x) = 2^x$;

в) $f(x, y) = 10^{x \cdot y}$;

г) $f(x, y) = [x / y]$.

Задание 3. Какая функция получится с помощью схемы примитивной рекурсии:

а) $f(x, 0) = x$;

$f(x, y+1) = x^{f(x, y)}$;

б) $f(x, 0) = x$;

$f(x, y+1) = (f(x, y))^x$.

Задание 4. Доказать, что следующие функции примитивно-рекурсивны:

а) $f(x) = x \dot{-} 1 = \begin{cases} 0, & \text{если } x = 0; \\ x-1, & \text{если } x > 0. \end{cases}$

б) $f(x, y) = x \dot{-} y = \begin{cases} 0, & \text{если } x \le y; \\ x-y, & \text{если } x > y. \end{cases}$

в) $f(x, y) = \begin{cases} x-y, & \text{если } x \ge y; \\ y-x, & \text{если } x < y. \end{cases}$.

Самостоятельно:

1) $f(x, y) = x \dot{-} (x \dot{-} y)$;

2) $f(x, y) = y + x \dot{-} y$;

3) $g(x, y) = \begin{cases} 0, & \text{если } x = y; \\ 1, & \text{если } x \ne y. \end{cases}$

4) $\overline{g}(x, y) = \begin{cases} 1, & \text{если } x = y; \\ 0, & \text{если } x \ne y. \end{cases}$

Практическое занятие 5

Примитивно-рекурсивные функции от *n*-переменных

1. Доказать, что если функция $f(x_1, x_2, ..., x_n)$ примитивно рекурсивная, то следующие функции примитивно рекурсивны:

а) $h(x_1, ..., x_n, x_{n+1}) = f(x_1, ..., x_n)$ – введение фиктивного аргумента;

б) $\varphi(x_1, ..., x_{n-1}) = f(x_1, x_1, x_2, ..., x_{n-1})$ – отождествление аргументов.

2. Доказать, что следующие функции примитивно рекурсивны:

а) $f(x) = x - 2$;

б) $f(x) = 2^{x-1}$.

3. Пусть g – примитивно рекурсивная функция. Доказать, что следующие функции примитивно рекурсивны:

а) $f(x_1, ..., x_n, x_{n+1}) = \sum_{i=0}^{x_{n+1}} g(x_1, ..., x_n, i)$;

б) $f(x_1, ..., x_n, y, z) = \begin{cases} \sum_{i=y}^{z} g(x_1, ..., x_n, i), & если\ y \le z; \\ 0, & если\ y > z; \end{cases}$

Самостоятельно:

1) $f(x_1, ..., x_n, x_{n+1}) = \prod_{i=0}^{x_{n+1}} g(x_1, ..., x_n, i)$;

2) $f(x_1, ..., x_n, y, z) = \begin{cases} \prod_{i=y}^{z} g(x_1, ..., x_n, i), & если\ y \le z; \\ 0, & если\ y > z. \end{cases}$

Индивидуальное задание 3

Примитивно-рекурсивные функции

1. Доказать, что следующие функции примитивно-рекурсивны:

1. $f(x) = x + n$; $f(x) = e^x$; $f(x, y) = (x \dot{-} y) + y$.

2. $f(x) = x!$; $f(x) = x\sqrt{2}$ ·; $f(x, y) = (x + y) \dot{-} y$.

3. $f(x) = 10^x$; $f(x) = a + b \cdot x$, a – const; $f(x, y) = (x + y) \dot{-} (y + x)$.

4. $f(x) = x + 2$; $f(x) = 10^{y-x}$; $f(x, y) = y \dot{-} (y \dot{-} x)$.

5. $f(x) = 2^x$; $f(x) = x - n$, n – const; $f(x, y) = (x \dot{-} y) + (y \dot{-} x)$.

6. $f(x) = x \dot{-} (x \dot{-} y)$; $f(x, y) = x^y$; $f(x) = 2 \cdot x$.

7. $sg(x) = \begin{cases} 0, x = 0, \\ 1, x > 0 \end{cases}$; $f(x) = 2 \cdot x - 1$; $f(x) = 100^{x-1}$.

8. $\overline{sg}(x) = \begin{cases} 1, x = 0, \\ 0, x > 0 \end{cases}$; $f(x) = x - 10$; $f(x, y) = 2^{x \cdot y}$.

9. $x \dot{-} 1 = \begin{cases} 0, x = 0, \\ x - 1, x > 0 \end{cases}$; $f(x) = c \cdot x - d$, c, d – const; $f(x) = x \cdot \sqrt{3}$.

10. $f(x)=3^x$; $f(x)=a - b \cdot x$; a, b – const; $f(x, y) = (x + y) \dot{-} (x \dot{-} y)$.

11. $f(x, y) = (y - x) + x$; $f(x, y)=x^{1-y}$; $f(x) = x / 2$;

12. $f(x) = n - x$; $f(x) = 2^{1-x}$; $f(x, y) = (y \dot{-} x) + x$.

* считать везде "–" урезанным вычитанием "$\dot{-}$".

2. Какая функция получится с помощью схемы примитивной рекурсии (по данной схеме примитивной рекурсии восстановить функцию):

1. $f(x, 0) = x$, $f(x, y+1) = f(x, y)^x$.
2. $f(x, 0) = x$, $f(x, y+1) = x^{f(x, y)}$.
3. $g(x, 0) = 2$, $g(x, y+1) = 2^{g(x, y)}$.
4. $g(x, 0) = 2$, $g(x, y+1) = g(x, y)^2$.
5. $f(x, 0) = x$, $f(x, y+1) = f(x, y) * x$.
6. $f(x, 0) = x$, $f(x, y+1) = f(x, y) + x$.
7. $g(x, 0) = 2$, $g(x, y+1) = g(x, y) * 2$.
8. $g(x, 0) = 2$, $g(x, y+1) = g(x, y) + 2$.
9. $f(x, 0) = x$, $f(x, y+1) = f(x, y)^2$.
10. $f(x, 0) = x$, $f(x, y+1) = 2^{f(x,y)}$.
11. $f(x, 0) = 10$, $f(x, y+1) = f(x, y) + 10$.
12. $g(x, 0) = 2^x$, $g(x,y+1) = g(x, y) * 2^x$.

П р а к т и ч е с к о е   з а н я т и е   6

Нормальные алгоритмы Маркова (НАМ)

Марковской подстановкой называется операция над словами, задаваемыми с помощью упорядоченной пары слов (*P, Q*), состоящая в следующем. В заданном слове *R* находят первое вхождение слова *P* (если оно есть) и, не изменяя остальных частей слова R, заменяют в нем это вхождение словом *Q*. Полученное слово называется результатом применения марковской подстановки (*P, Q*) к слову *R*. Если же первого вхождения *P* в слово *R* (и, следовательно, вообще нет ни одного вхождения *P* в *R*), то считается, что марковская подстановка (*P, Q*) не применима к слову *R*.

Запись $P \to Q$ называется формулой подстановки (*P*,*Q*). *P* называется левой частью, *Q* – правой частью в формуле подстановки. Некоторые подстановки называются заключительными. Для обозначения таких подстановок будем использовать запись $P \to \cdot Q$, называя ее формулой заключительной подстановки.

Упорядоченный конечный список формул подстановок в алфавите А

$$\begin{cases} P_1 \to (\cdot)Q_1 \\ P_2 \to (\cdot)Q_2 \\ \vdots \\ P_n \to (\cdot)Q_n \end{cases}$$

называется схемой нормального алгоритма в алфавите А. Запись точки в скобках означает, что она может стоять на этом месте, а может отсутствовать.

Пример. Построить нормальный алгоритм Маркова, заменив в алфавите А = {*a, b, c*} все буквы *a* на *c*. Используем символ α для расширения алфавита А. В = $\{\alpha\} \cup$А. Схема *Z* нормального алгоритма будет иметь следующий вид:

$$
Z: \begin{cases} \alpha a \longrightarrow c\alpha \\ \alpha b \longrightarrow b\alpha \\ \alpha c \longrightarrow c\alpha \\ \alpha \longrightarrow \cdot\Lambda \\ \Lambda \longrightarrow \alpha \end{cases}
$$

Например, $aacbab \Rightarrow \alpha aacbab \Rightarrow c\alpha acbab \Rightarrow cc\alpha cbab \Rightarrow ccc\alpha bab \Rightarrow cccb\alpha ab \Rightarrow cccbc\alpha b \Rightarrow cccbcb\alpha \Rightarrow cccbcb.$

Этот алгоритм может быть реализован так же следующей схемой:

$$
Z_1: \begin{cases} a \to c \\ \Lambda \to \cdot\Lambda \end{cases}.
$$

Задание 1. Что получится в результате следующих марковских подстановок в слово «апельсин»:

а) (Λ, к);

б) (пельс, спир);

в) (ль, Λ).

Задание 2. Построить нормальный алгоритм Маркова для вычисления функции $f(x) = x - 1\ (x>1)$ в унарной системе счисления.

Задание 3. Исполнить алгоритм вычисления функции $f(x) = x + 1$ в десятичной системе счисления.

Задание 4. Построить нормальный алгоритм Маркова для вычисления функции $f(x) = x - 1$ в десятичной системе счисления.

Задание 5. Дано слово в алфавите $A = \{a, b, c\}$. Построить алгоритм Маркова, присоединяющий слово Q к данному слову.

Решение:

$$\begin{cases} \varepsilon a \to a\varepsilon \\ \varepsilon b \to b\varepsilon \\ \varepsilon c \to c\varepsilon \\ \varepsilon \to \cdot Q \\ \Lambda \to \varepsilon \end{cases}$$

Задание 6. Построить нормальный алгоритм Маркова, удваивающий слово в унарной системе счисления.

Задание 7. Построить нормальный алгоритм Маркова для перевода числа из двоичной системы счисления в четверичную систему счисления.

Самостоятельно:

1. Дан алфавит А = {*a, b, c*}. Заменить все *a* на *bc*.
2. Дан алфавит А = {*a, b, c*}. Удалить все *b*.
3. Построить нормальный алгоритм Маркова для вычисления функции $f(x) = x - 1$ в троичной системе счисления.
4. Построить нормальный алгоритм Маркова для перевода числа из четверичной системы счисления в двоичную систему счисления.

Практическое занятие 7

Нормальные алгоритмы Маркова (работа с эмулятором)

Рассмотрим примеры, в которых демонстрируются типичные приёмы составления нормальных алгоритмов Маркова (НАМ).

Как и в случае машины Тьюринга, для сокращения формулировки задач будем использовать следующие соглашения:

– буквой *P* будем обозначать входное слово;

– буквой *A* будем обозначать алфавит входного слова, т.е. набор тех символов, которые могут входить во входное слово *P* (но в процессе выполнения НАМ в обрабатываемых словах могут появляться и другие символы).

На рисунке 2.1 приведен пример работы эмулятора.

Рассмотрен пример добавления к слову в алфавите $A = \{a, b, c, d\}$ слова $Q = aaaa$.

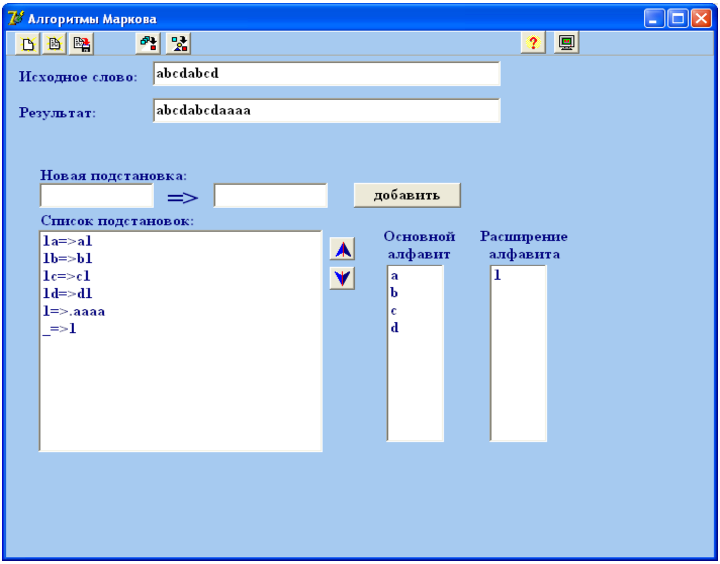

Рис. 2.1. Главное окно эмулятора НАМ

Задание 1. Вставка и удаление символов.

$A = \{a, b, c, d\}$. В слове P требуется заменить первое вхождение подслова bb на ddd и удалить все вхождения символа c. Исполнить по шагам.

Например: $abbcabbca \to adddabba$.

Задание 2. Перестановка символов.

$A = \{a, b\}$. Преобразовать слово P так, чтобы в начале оказались все символы a, а в конце – все символы b.

Например: $babba \to aabbb$.

Задание 3. Использование спецзнака.

$A = \{a, b\}$. Удалить из непустого слова P его первый символ. Пустое слово не менять.

Задание 4. Фиксация спецзнаком заменяемого символа.

$A = \{0, 1, 2, 3\}$. Пусть P – непустое слово. Трактуя его как запись неотрицательного целого числа в четверичной системе счисления, требуется получить запись этого же числа, но в двоичной системе.

Например: 0123 → 00011011.

Самостоятельно:

1. Составить НАМ, проверяющий деление десятичного числа на 5.
2. $A = \{a, b, c\}$. Составить НАМ, перемещающий все буквы а в конец слова.
3. Построить нормальный алгоритм Маркова для вычисления разности двух чисел, представленных в унарной системе счисления.

Индивидуальное задание 4

Нормальные алгоритмы Маркова

Задание 1.

1. Построить нормальный алгоритм Маркова, который бы в слове из алфавита $A = \{a, b, c, d, e, f\}$ все вхождения последовательности *abc* заменял на символ *f* и удалял первое вхождение пары *cf*.
2. Построить нормальный алгоритм Маркова, который бы в слове из алфавита $A = \{a, b, c, d, e, f\}$ удалял все вхождения последовательности *bc* и удваивал гласные буквы.
3. Построить нормальный алгоритм Маркова, который бы в слове из алфавита $A = \{a, b, c, d, e, f\}$ все символы *a* заменял на *f*, а все *f* – на *af*.

4. Построить нормальный алгоритм Маркова, который бы в любом слове из алфавита $A = \{a, b, c, d, e, \ldots, x, y, z\}$ удваивал все буквы, стоящие на четных местах в исходном слове.

5. Построить нормальный алгоритм Маркова, который упорядочивает любое слово в алфавите $A = \{a, b, c, d\}$.

6. Поменять местами первый и последний символы слова в алфавите $A = \{a, b, c, d, e\}$.

7. Даны два слова из букв, записанные через * в алфавите $A = \{a, b, c, d, e, f\}$. Вместо пробела поставить « = ».

8. Построить нормальный алгоритм Маркова, реализующий циклический сдвиг первого символа слова в алфавите $A = \{a, b, c, d, e, f, g\}$.

9. Построить нормальный алгоритм Маркова, который бы в слове из алфавита $A = \{a, b, c, d, e, f\}$ все вхождения последовательности *cde* заменял на символ *a* и удваивал согласные буквы.

10. Построить нормальный алгоритм Маркова, который бы в слове из алфавита $A = \{a, b, c, d, e, f\}$ удалял все вхождения последовательности *fe* и заменял первое вхождение *da* на *b*.

11. Построить нормальный алгоритм Маркова, который бы переворачивал любое заданное слово в алфавите $A = \{a, b, c, d\}$.

12. Построить нормальный алгоритм Маркова, который бы в слове из алфавита $A = \{a, b, c, d, e, f\}$ все символы *e* заменял на *d*, а все *d* – на *de*.

13. Построить нормальный алгоритм Маркова, который бы в любом слове из алфавита $A = \{a, b, c, d, e, \ldots, x, y, z\}$ удваивал все гласные буквы.

14. Поменять местами первый и последний символа слова в алфавите $A = \{0, 1, 2, 3, 4, 5\}$.

15. Построить нормальный алгоритм Маркова, реализующий циклический сдвиг последнего символа слова в алфавите $A = \{a, b, c, d, e, f\}$.

16. Построить нормальный алгоритм Маркова для вычисления функции $f(x) = x \bmod 3$ в унарной системе счисления.

17. Построить нормальный алгоритм Маркова для вычисления функции $f(x) = x\ div\ 3$ в унарной системе счисления.

18. Построить нормальный алгоритм Маркова для перевода числа из двоичной системы счисления в восьмеричную систему счисления.

19. Построить нормальный алгоритм Маркова для перевода числа из двоичной системы счисления в шестнадцатеричную систему счисления.

20. Построить нормальный алгоритм Маркова, который бы в слове из алфавита $A = \{a, b, c, d, e, f\}$ все символы e заменял на d, а все d – на de.

21. Построить нормальный алгоритм Маркова, который бы в любом слове из алфавита $A = \{a, b, c, d, e, ..., x, y, z\}$ удваивал все гласные буквы и удалял согласные буквы.

22. Построить нормальный алгоритм Маркова, который в любом слове в алфавите $A = \{a, b, c\}$ переносит все буквы a в конец слова, а буквы b – в конец слова.

23. Поменять местами первый и последний символы слова в алфавите $A = \{0, 1, 2, 3, 4, 5\}$.

24. Построить нормальный алгоритм Маркова для перевода числа из шестнадцатеричной системы счисления в двоичную систему счисления.

Задание 2 (+ 0,2 балла).

1. Даны два слова в унарной системе, записанные через *. Вместо звездочки поставить >, <, =.

2. Построить нормальный алгоритм Маркова для определения системы счисления (найти возможное минимальное основание), в которой записано натуральное число (предполагается, что основание не может превышать 16_{10}).

3. Считая непустое слово P записью двоичного числа в алфавите $A = \{0, 1\}$, определить, является ли это число степенью 2 (1, 2, 4, …). Ответ: слово 1, если является, или слово 0 иначе.

4. Считая непустое слово P в алфавите $A = \{0, 1, 2, 3\}$ записью четверичного числа, проверить, чётное оно или нет. Ответ: слово 0, если чётное, и слово 1 иначе.

5. Считая непустое слово P в алфавите $A = \{0, 1, 2, 3\}$ записью четверичного числа, получить остаток от деления этого числа на 4.

6. Считая непустое слово P в алфавите $A = \{0, 1\}$ записью двоичного числа, получить это же число, но в четверичной системе. (*Замечание*: учесть, что в двоичном числе может быть нечётное количество цифр.)

7. Считая непустое слово P в алфавите $A = \{0, 1, 2\}$ записью троичного числа, увеличить это число на 3.

8. Считая непустое слово P в алфавите $A = \{0, 1, 2\}$ записью положительного троичного числа, уменьшить это число на 3.

9. Считая слово P записью числа в единичной системе счисления, получить запись этого числа в троичной системе. (*Рекомендация*: следует в цикле удалять из «единичного» числа по единице и каждый раз прибавлять 1 к троичному числу, которое вначале положить равным 0).

10. Считая непустое слово P в алфавите $A = \{0, 1, 2\}$ записью числа в троичной системе, получить запись этого числа в единичной системе.

11. Определить, входит ли первый символ непустого слова P в алфавите $A = \{a, b, c\}$ ещё раз в это слово. Ответ: слово a, если входит, или пустое слово иначе.

12. Если в непустом слове P в алфавите $A = \{a, b\}$ совпадают первый и последний символы, то удалить оба этих символа, а иначе слово не менять.

13. Определить, является ли слово P в алфавите $A = \{a, b\}$ палиндромом (перевёртышем, симметричным словом). Ответ: слово a, если является, или пустое слово иначе.

14. Пусть слово P в алфавите $A = \{a, b\}$ имеет нечётную длину. Удалить из него средний символ.

15. Пусть P имеет вид $Q = R$, где Q и R – любые слова из символов a и b. Выдать ответ a, если слова Q и R одинаковы, и пустое слово иначе.

16. Пусть P имеет вид $Q = R$, где Q и R – непустые слова из символов 0 и 1. Трактуя Q и R как записи двоичных чисел (возможно, с незначащими нулями), выдать в качестве ответа слово 1, если эти числа равны, и слово 0 иначе.

17. Пусть P имеет вид $Q > R$, где Q и R – непустые слова из символов 0 и 1. Трактуя Q и R как записи двоичных чисел (возможно, с незначащими нулями), выдать в качестве ответа слово 1, если число Q больше числа R, и слово 1 иначе.

18. Определить, сбалансировано ли слово P в алфавите $A = \{(,)\}$ по круглым скобкам. Ответ: y (да) или n (нет).

Вопросы к модулю 2

1. В чем заключается конструктивный подход, принятый в теории алгоритмов?
2. Как называются операции над функциями?
3. Что входит в базис?
4. Дать определение функции проекции, суперпозиции, примитивной рекурсии.
5. Дать определение примитивно-рекурсивной функции.
6. Как доказать примитивную рекурсию сложения? вычитания?
7. Какое отношение называется примитивно-рекурсивным?
8. Какой предикат называется примитивно-рекурсивным?
9. Какой оператор называется примитивно-рекурсивным?
10. Как задается циклическая перестановка аргументов, отождествление и др.?
11. Как определяется ограниченный оператор наименьшего числа (μ-оператор)?
12. Какой оператор называется оператором обращения?
13. Где определены примитивно-рекурсивные функции?

14. Как задается функция Аккермана?

15. Является ли функция Аккермана примитивно-рекурсивной? общерекурсивной?

16. Дать определение частично-рекурсивной функции.

17. Какая частично-рекурсивная функция называется общерекурсивной?

18. Как выполняются свойства детерминированности, массовости и дискретности для частично-рекурсивных функций?

19. Что собой представляют нормальные алгоритмы Маркова?

20. Дать определение алфавита, слова.

21. Какая операция называется «Марковской подстановкой»?

22. Как записывается формула подстановки? Конечной подстановки?

23. Как записывается схема нормального алгоритма Маркова?

24. Что означает, что алгоритм М применим к слову Р?

25. В чем заключается правило построения последовательности $\{P_i\}$ слов в алфавите А?

26. Что является результатом применения нормального алгоритма к слову Р?

27. Какая функция называется частично вычислимой по Маркову?

28. Сформулировать принцип нормализации Маркова.

29. Что называется расширением алфавита?

30. Какая функция называется вычислимой по Маркову?

31. В каком случае можно говорить о нормальном алгоритме, что он замкнут?

32. Что называется естественным распространением алгоритма Маркова?

33. В каком случае композиция алгоритмов называется нормальной композицией алгоритмов M_1 и M_2?

34. Сформулировать принцип нормализации Маркова.

МОДУЛЬ 3. МАШИНА С НЕОГРАНИЧЕННЫМИ РЕГИСТРАМИ

Практическое занятие 8

Машина с неограниченными регистрами (МНР)

Машина с неограниченными регистрами (МНР) состоит из:

1) ленты, содержащей бесконечное число регистров, обозначаемых через R_1, R_2, R_3, …, каждый из которых в любой момент времени содержит некоторое натуральное число. Число, содержащееся в R_n, мы будем обозначать через r_n;

2) программы P, состоящей из конечного списка команд. Команды бывают следующих четырех видов:

a) команда обнуления – Z(n) заменяет содержание R_n на 0; обозначается также $0 \to R_n$ или $r_n := 0$ (читается, как r_n присваивается 0);

b) команды прибавления единицы – S(n) увеличивает содержимое R_n на 1; обозначается также $r_n + 1 \to R_n$ или $r_n := r_n + 1$ (r_n присваивается $r_n + 1$);

c) команда переадресации – T(m, n) заменяет содержимое R_n числом r_m, содержащимся в R_m; обозначается также $r_m \to R_n$ или $r_n := r_m$ (r_n присваивается r_m);

d) команда условного перехода – J(m, n, q) сравнивает содержимое регистров R_m и R_n, далее, если $r_m = r_n$, то МНР переходит к выполнению q-й команды программы P, если $r_m \neq r_n$, то МНР переходит к выполнению следующей команды в P. Если условный переход невозможен ввиду того, что в P меньше, чем q команд, то МНР прекращает работу.

Команды обнуления, прибавления единицы и переадресации называются арифметическими.

Пример. $f(x, y) = x - y$. Мы получим $x - y$, прибавляя 1 к y (используя соответствующие команды) y раз. Вычисление $x - y$ начинается с начальной конфигурации x, y, 0, 0, …; наша программа продолжает прибавление 1 к r_2, используя R_3 как счетчик числа прибавлений 1 к r_2. Типичной конфигурацией в процессе вычисления является:

R_1	R_2	R_3	R_4	R_5	
x	y	k	0	0	

Программа спроектирована на останов, когда $x = y$, при этом в регистре R_1 оказывается число k, что и требовалось. Итак, нашу задачу решает МНР, программа которой следующая:

I_1 J (1,2,6)
I_2 S (2)
I_3 S (3)
I_4 J (1,2,6)
I_5 J (1,1,2)
I_6 T (3,1)

Таким образом, функция $f(x, y) = x - y$ оказывается вычислимой.

Задания

1. Построить МНР, вычисляющую функцию $f(x, y) = x + y$. Исполнить для чисел 3 и 2.

R_1	R_2	R_3	R_4	R_5	
x+k	y	k	0	0	

I_1 J(3,2,5)	3	2	0
I_2 S(1)	4	2	1
I_3 S(3)	5	2	2
I_4 J(1,1,1)			

2. Построить МНР, вычисляющую функцию

$$f(x) = \begin{cases} \frac{1}{2}x, & \text{если } x \text{ четно}, \\ \text{неопределена}, & \text{если } x \text{ нечетно}. \end{cases}$$

Исполнить алгоритм для числа 6.

R_1	R_2	R_3	R_4	R_5	
x	2k	k	0	0	…

I_1	J(1,2,6)	6	0	0	0
I_2	S(3)	6	1	1	0
I_3	S(2)	6	2	1	0
I_4	S(2)	6	3	2	0
I_5	J(1,1,1)	6	4	2	0
I_6	T(3,1)	6	5	3	0
		6	6	3	0
		3	6	3	0

3. Построить МНР, вычисляющую функцию $f(x, y) = x - 2$.
4. Построить МНР, вычисляющую функцию $f(x, y) = x * y$.
5. Построить МНР, вычисляющую функцию $f(x, y) = x / y$.

Самостоятельно:

1. Построить МНР, вычисляющую функцию

$$f(x) = \begin{cases} \frac{1}{3}x, & \text{если } x \text{ делится на } 3, \\ \text{неопределена}, & \text{в остальных случаях}. \end{cases}$$

2. Построить МНР, вычисляющую функцию f(x) = 2 * x.

3. Построить МНР, вычисляющую функцию f(x) = x – 2, если x > 2.

4. Построить МНР, вычисляющую функцию

$$f(x, y) = (x + y) / 2.$$

Практическое занятие 9

Машина с неограниченными регистрами (работа с эмулятором)

Задание 1. Знакомство с эмулятором «Машина с неограниченными регистрами».

1. Запустить эмулятор «МНР» с внутреннего. Портала ЮУрГГПУ: ЮУрГГПУ>Учебно-метод. матер. >IITiMOI>ISiT> 3k>TA>Emuliytors>MNR.swf

2. Познакомиться с интерфейсной частью программы, рассмотреть возможности меню (рис. 3.1);

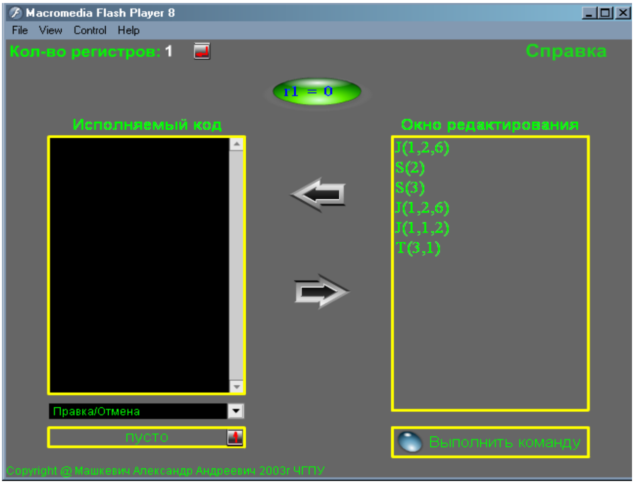

Рис. 3.1. Внешний вид эмулятора МНР

Задание 2. Реализация готовых алгоритмов в эмуляторе «МНР».

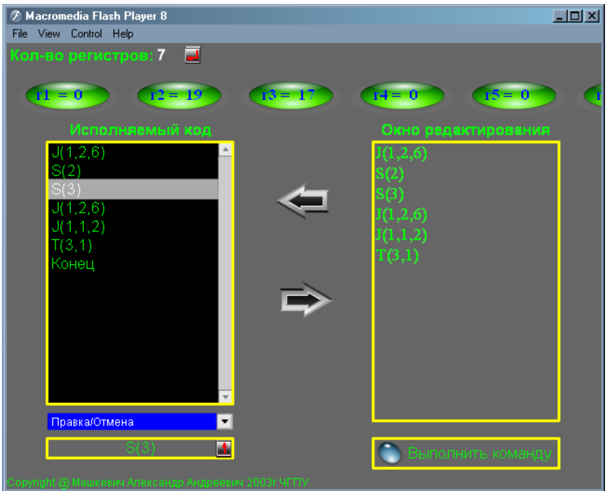

Рис. 3.2. Пример вычислений в эмуляторе «МНР»

1. Рассмотреть пошагово работу эмулятора для задачи, представленной на рисунке 3.2.

2. Какой алгоритм реализован с помощью данной МНР?

Задание 1. Построить МНР, вычисляющую функцию $f(x) = 4 \cdot x$.

Задание 2. Составить программу для вычисления функции

$$f(x) = \begin{cases} 0, & \text{если} \quad x = 0, \\ x - 1, & \text{если} x <> 0 \end{cases}$$

Самостоятельно:

1. Составить МНР-программу, вычисляющую f(x) = a · x + b.
2. Построить МНР, вычисляющую остаток от деления x на y.

Индивидуальное задание 5

Машина с неограниченными регистрами

Задание 1. Построить машину с неограниченными регистрами, вычисляющую функцию $y(x)$:

1. $y = a * x + b$.
2. $y = x \ div\ 2$.
3. $y = b – a * x$.
4. $y = x^2$.
5. $y = 2 * x – 1$.
6. $y = a * x$.
7. $y = a * x – b$.
8. $y = x \ mod\ 2$.
9. $y = a^x$.
10. Среднее 2-х чисел.
11. $y = 2^x$.
12. Проверка на четность.
13. $y = x^3$.
14. Проверка деления на 3.
15. $y = x / 3$.
16. Проверка на нечетность.

17. $y = x / 2$.

18. Проверка деления на 5.

19. $y = x\ div\ 3$.

20. $y = 3^x$.

21. $y = x\ mod\ 3$.

22. $y = x / a - d$.

23. $y = b - x / a$.

24. Проверка деления на 10.

Задание 2. Построить машину с неограниченными регистрами (не из списка) +0,2 балла.

Вопросы к модулю 3

1. Из чего состоит машина с неограниченными регистрами?
2. Охарактеризовать каждую составляющую МНР.
3. Из чего состоит память машины с неограниченными регистрами?
4. Бесконечна ли лента в одну сторону? в обе?
5. Сколько символов можно записать в одну ячейку?
6. Объяснить команды машины: Z, S, T, J.
7. Какие команды относятся к арифметическим?
8. Что является данными МНР?
9. Что собой представляет память, элементарные шаги, детерминированность МНР?
10. Что является результатом работы МНР?
11. Какая функция называется МНР-вычислимой?
12. Как выполняются требования к алгоритмам на примере МНР?
13. В каком случае вычисление $P(a_1, a_2, \ldots, a_n)$ сходится к b?
14. Что обозначает запись $P(a_1, a_2, a_3, \ldots)$?
15. Что является результатом работы МНР?

МОДУЛЬ 4. ВЫЧИСЛИМОСТЬ И РАЗРЕШИМОСТЬ

Практическое занятие 10

Нумерация программ

Биекция π: $N*N \to N$ определяется равенством $\pi(m, n) = 2^m(2 \cdot n + 1) - 1$.

Величину π^{-1} можно задать как $\pi^{-1}(x) = (\pi_1(x), \pi_2(x))$, где π_1 и π_2 – вычислимые функции, определяемые равенством: π_1 = минимальной степени 2 в двоичном разложении числа x;

$$\pi_2 = \frac{1}{2}\left(\frac{x+1}{2^{\pi_1(x)}} - 1\right).$$

Для явного задания биекции $\zeta : N^+ * N^+ * N^+ \to N$ используем функцию π:

$$\zeta(m, n, q) = \pi(\pi(m-1, n-1), q-1).$$

Тогда $\zeta^{-1}(x) = (\pi_1(\pi_1(x)) + 1, \pi_2(\pi_1(x)) + 1, \pi_2(x) + 1)$.

Биекция $\tau : \bigcup_{k>0} N^k \to N$ задается тождеством:

$$\tau(a_1, \ldots, a_к) = 2^{a_1} + 2^{a_1+a_2+1} + 2^{a_1+a_2+a_3+2} + \ldots + 2^{a_1+a_2+\ldots+a_k+k-1} - 1.$$

$\tau^{-1}(x) = (a_1, \ldots, a_k)$, где $a_1 = b_1$, а $a_{i+1} = b_{i+1} - b_i - 1$ $(1 <= i < k)$.

Определим биекцию β: $\mathcal{L} \to N$, которая отображает команды четырех типов в натуральные числа вида $4u$, $4u+1$, $4u+2$, $4u+3$ соответственно; используем функции π и ξ:

$\beta(Z(n)) = 4 \cdot (n - 1)$,

$\beta(S(n)) = 4 \cdot (n - 1) + 1$,

$\beta(T(m, n)) = 4 \cdot \pi(m - 1, n - 1) + 2$,

$\beta(J(m, n, q)) = 4 \cdot \xi(m, n, q) + 3$.

Для вычисления $\beta^{-1}(x)$ найдем сначала такие числа u, r, что $x = 4 \cdot u + r$ при $0 \le r < 4$. Значение r указывает тип команды $\beta^{-1}(x)$, а u позволяет найти конкретно команду данного типа, а именно:

– если $r = 0$, то $\beta^{-1}(x) = Z(u + 1)$;

– если $r = 1$, то $\beta^{-1}(x) = S(u + 1)$;

– если $r = 2$, то $\beta^{-1}(x) = T(\pi_1(u) + 1, \pi_2(u) + 1)$;

– если $r = 3$, то $\beta^{-1}(x) = I(m, n, q)$, где $(m,n,q) = \xi^{-1}(u)$.

Определим биекцию $\gamma : P \to \mathrm{N}$ следующим образом: если $P = I_1, I_2, \ldots, I_s$, то $\gamma(P) = \tau(\beta(I_1), \ldots, \beta(I_s))$.

Примеры.

А. Пусть Р является программой $T(1,3)$, $S(4)$, $Z(6)$. Вычислим $\gamma(P)$.

$\beta(T(1,3)) = 4\pi(0,2) + 2 = 4 \cdot (2^0 \cdot (2*2+1)) + 2 = 18$,

$\beta(S(4)) = 4 \cdot 3 + 1 = 13$,

$\beta(Z(6)) = 4 \cdot 5 = 20$.

Следовательно $\gamma(P) = 2^{18} + 2^{32} + 2^{53} - 1 = 9007203549970431$.

В. Пусть $n = 4127$, найдем P_{4127}. $4127 = 2^5 + 2^{12} - 1$; следовательно, P_{4127} является программой с двумя командами I_1 и I_2, где

$\beta(I_1) = 5 = 4 \cdot 1 + 1$.

$\beta(I_2) = 12 - 5 - 1 = 6 = 4 \cdot 1 + 2 = 4\pi(1,0) + 2$.

Согласно определению β, $I_1 = S(2)$, а $I_2 = T(2,1)$, и, значит, P_{4127} есть $S(2)$, $T(2,1)$.

Задание 1.

1. Пусть Р является программой Z(3), S(2), T(3,1). Вычислить $\gamma(P)$.

2. Пусть Р является программой S(2), S(3), T(3,1). Вычислить $\gamma(P)$.

3. Пусть $n = 1089$, найти P_{1089}.

4. Пусть $n = 523$, найти P_{523}.

Вопросы к модулю 4

1. Кто впервые сформулировал принцип, утверждающий пригодность некоторых конкретных уточнений понятия «алгоритм»?

2. Является ли разрешимой задача распознавания эквивалентности примитивно-рекурсивных описаний?

3. Пусть *X* – множество конечных объектов; в каком случае *X* называется эффективно счетным?

4. Существует ли невычислимая всюду определенная (или тотальная) функция?

5. Кто предложил метод диагональной конструкции для построения функции f?

6. Как можно найти индекс вычислимой функции?

7. Как обозначается область определения $\Phi_a^{(n)} = \{(x_1, \ldots, x_n) | \mathrm{P_a}(x_1, \ldots, x_n)\downarrow\}$?

8. Как обозначается *n*-местная функция, вычислимая по программе $\mathrm{P_a} = f_{P_a}^{(n)}$?

9. Как обозначается множество значений функции $\Phi_a^{(n)}$?

10. Для любого перечисления множества всюду определенных вычислимых функций существует ли общерекурсивная функция, входящая в это перечисление?

11. Что означает, что проблема «$x \in W_x$» неразрешима?

12. Что означает, что проблема «Φ_x всюду определена» неразрешима?

13. Что означает, что проблема «$\Phi_x(y)$ определена» неразрешима?

14. Что означает, что проблема «$\Phi_x = g$», где g – любая фиксированная вычислимая функция, неразрешима?

15. Можно ли проверить, вычисляют ли две программы одну и ту же одноместную функцию?

16. Как называется множество M, если существует общерекурсивная функция $\psi_M(x)$, такая, что $a \in M$ тогда и только тогда, когда для некоторого x $a = \psi_M(x)$?

17. Существует ли множество M, которое перечислимо, но не разрешимо?

18. В чем смысл теоремы Райса (с точки зрения программирования)?

19. Множество квадратов натуральных чисел $M = \{a | a = x^2\}$ перечислимо и разрешимо?