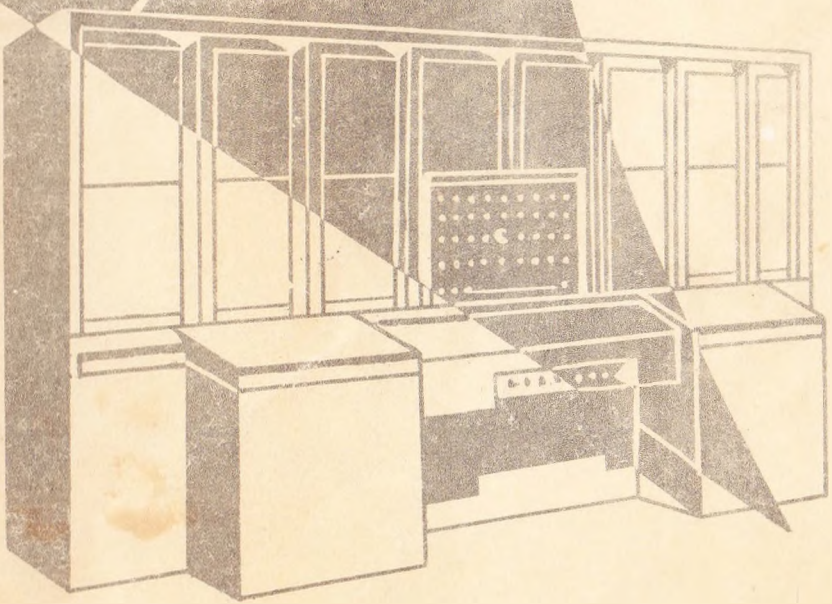
КИЕВСКИЙ ДОМ НАУЧНО-ТЕХНИЧЕСКОЙ ПРОПАГАНДЫ

КИБЕРНЕТИКА НА ТРАНСПОРТЕ

ЗАОЧНЫЙ СЕМИНАР



НТО
радиотехники
и электросвязи

Киевский дом
научно-технической
пропаганды

ДорНТО
Юго-Западной
железной дороги

Дом техники
Юго-Западной
железной дороги

Кандидат физико-математических наук
ЮЩЕНКО Е.Л.

Тема 5

АДРЕСНЫЙ ЯЗЫК

1962

Непосредственное программирование задач для конкретных машин сопряжено с рядом неудобств. Программисту приходится работать с огромным вниманием с тем, чтобы учесть недопустимость одновременного использования одного и того же адреса в различных целях или стирания результата, еще необходимого в вычислениях, проследить за правильным применением констант и т.д. Трудоемкий и однообразный труд программиста уже на самых ранних порах развития вычислительной техники поставил как одну из наиболее насущных проблем задачу разработки определенных методов программирования, получивших название методов автоматизации программирования.

Одним из наиболее перспективных направлений в развитии автоматизации программирования является разработка так называемых универсальных программирующих программ и метод библиотечных подпрограмм.

Метод универсальных программирующих программ предполагает широкое использование самой машины для составления программ.

С этой целью весь порядок выполнения расчетов - алгоритм решения задачи - строго формально записывается некоторым образом. Кроме того, раз и навсегда разрабатывается программа - программирующая программа, которая из любой такой формальной записи создает программу в кодах данной конкретной машины. Разумеется, что чем проще правила формальной записи алгоритмов, чем ближе они к общепринятым в математике способам их описания, тем они удобнее для того, кто собирается ставить

задачу на машину. Однако, наоборот, чем ближе эти правила записи к машинному коду программ, тем легче будет составить программирующую программу и тем проще она будет.

В основу построения современных вычислительных машин положены два основных принципа: принцип программного управления и принцип адресности. Первый из них означает полную автоматизацию вычислительных работ после того, как в машину введена программа задачи. Второй – означает, что раз составленная для определенных исходных данных программа решения некоторой конкретной задачи является готовой программой для решения любых задач данного класса, поскольку при составлении программ используются лишь адреса, в которые помещаются исходные данные и промежуточные и окончательные результаты, а не сами эти величины. Это позволяет поставить вопрос о накоплении программ, которые для удобства оформляются надлежащим образом – делаются стандартными. Затем программы других задач, более сложных, уже могут строиться из готовых, полностью проверенных кусков, что значительно ускоряет и облегчает процесс подготовки задач и их решение на машинах.

Отсюда возникает и третий подход к вопросу автоматизации программирования. А нельзя ли создать такие правила записи алгоритмов, такой "формальный язык", запись алгоритмов в котором могла бы быть использована программирующими программами машин различных типов? В этом случае программы, подготовленные в этом языке с целью их постановки на одних машинах, можно было бы успешно использовать на других машинах, что ускорило бы процесс накопления библиотечных программ.

Ниже будет дано описание языка, пригодного для указанных целей для машин, независимо от их конкретных технических особенностей, так называе-

мого адресного языка, разработанного в ВЦ АН УССР. Именно этот язык используется программирующими программами машин "Урал" и "Киев", эксплуатируемых в ВЦ АН УССР.

Существенно заметить, что адресный язык позволяет расширить круг лиц, способных непосредственно, без помощи программиста, подготовить задачу для ее решения на машине. От этих лиц при такой постановке вопроса вовсе не требуется знания специфических особенностей решения задач на данной конкретной машине. Близость же адресного языка к общепринятому языку формул делает его доступным для любого инженера.

Адресный язык

Адресный язык создан специально для записи алгоритмов. Он весьма близок к обычному языку математических формул и использует принятые в математике символы и обозначения. Однако в систему понятий адресного языка введены специальные понятия и соответствующие им символы, благодаря чему язык становится более приспособленным для описания алгоритмов. Введение новых понятий продиктовано опытом постановки задач на электронных вычислительных машинах с программным управлением.

С т р о к и

Алгоритм в адресном языке записывается в виде ряда строк, размещенных одна под другой. В каждой строке записывается одно или несколько действий, называемых формулами. Между действиями в строке, если их несколько, ставится один из знаков – точка с запятой ";" или запятая ",". Знак ";" ставится в том случае, когда порядок выполнения действий, разделяемых этим знаком, безразличен.

Знак "," ставится в том случае, когда требуется первое из указанных действий выполнить раньше, чем второе /первым считается стоящее слева от запятой/.

Меченые строки

Для указания порядка выполнения алгоритма те или иные его строки могут помечаться метками. Меткой может быть тот или иной набор символов - букв или цифр или других знаков. Для того, чтобы метка не могла быть спутана с другими выражениями языка в случае возможных недоразумений, метки можно специально выделять, например, подчеркивать.

Метка, помечающая строку алгоритма, ставится справа и отделяется знаком "..." /троеточие/. Строка может метиться несколькими метками.

Отдельно стоящая в строке метка

Метка без троеточия может составлять отдельную строку алгоритма. В этом случае она называется формулой безусловного перехода. Действие такой метки состоит в том, что она указывает строку алгоритма, к которой следует перейти для продолжения выполнения алгоритма.

Разумеется, что такая строка имеет смысл только в том случае, когда в алгоритме имеется строка, помеченная той же меткой.

Штрих-операция

В адресном языке вводится операция выделения содержимого адреса. Пусть a - элемент множества, называемого множеством адресов. Каждому элементу этого множества ставится в соответствие некоторый элемент в /код/, называемый содержимым адреса. Записываем

'a=в

Символ $'$ /штрих сверху слева/ является символом операции, выполнение которой означает извлечение содержимого по заданному адресу. Операция, обозначенная символом " $'$ ", может рассматриваться как функция от аргумента, который называется адресом.

Поскольку адреса принято обозначать числами, можно предположить, что содержимое некоторого адреса, в свою очередь, также является адресом. В этом случае имеем:

$$'в = с$$

Будем говорить, что а является адресом второго ранга элемента с и записывать это так:

$$''а = {}^{2}а = {}'в = с, \quad {}^{2}а = с$$

Понятие ранга обобщается на ранги более высокого порядка. О величине в будем говорить, что она является адресом нулевого ранга в.

Штрих-функция существенно однозначна: одному адресу соответствует одно содержимое, хотя различным адресам может соответствовать одно и то же содержимое. Штрих-функция задана, если установлено соответствие между адресами и их содержимым, называемое адресным отображением.

Вместе с тем будут заданы и содержимые по адресам высших рангов.

Операция засылки

Для изменения содержимого адресов - адресного отображения - применяется операция засылки, символом которой является " $\Longrightarrow$ " /стрелка/, соединяющая два элемента. Запись операции

$$а \Longrightarrow в$$

/читаем: а заслать по адресу в/ означает, что устанавливается соответствие вида $''в = а$, а все ранее установленные соответствия вида $'х = у$, где $х \neq в$, остаются неизменными.

Таким образом, после выполнения засылки $а \Longrightarrow в$

изменяется значение всех функций, в которые входят выражения $'в$, $^2в$ и т.д. Например, пусть $'2=31$; $'31 = 42$; $'8 = 12$; $'12 = 0$ /т.е. $^2 2 = 42$, $^2 8 = 0$/. Выполним операцию засылки $12 \Longrightarrow 2$; после выполнения этой операции будет $'2 = 12$; $^2 2 = '12 = 0$. Содержимое адресов 8,12 и 31 не изменилось.

Адресная функция

Функция, построенная, как обычно, с помощью применения к аргументам операций сложения, вычитания, взятия синуса и др. и взятия функции от функции, а также с помощью применения операции взятия содержимого адреса – "штрих"-операции, называется адресной функцией. Например:

$$f = \frac{10 + {}^2a + 'b}{\cos 'c}$$

Значение этой функции может быть вычислено, если задано адресное соответствие. Если $'a = в$; $'в = 3$; $'c = 0$, то $f = 16$.

Формулы засылки

Две адресные функции, соединенные знаком операции засылки, называются формулой засылки или адресной формулой. При этом требуется, чтобы значение функции /в частном случае это может быть просто адрес/, стоящей справа от стрелки, было адресом. Применение операции засылки состоит в следующем: вычисляются значения функций, стоящих справа и слева от символа $\Longrightarrow$, и адресу, являющемуся значением правой функции, присваивается значение стоящей слева функции.

Таким образом, формула засылки является операцией засылки по адресу.

Одна или несколько формул засылки могут составлять строку адресного алгоритма.

Формула безусловного перехода

Строку адресного алгоритма может составлять отдельно стоящая адресная функция. В этом случае она называется формулой безусловного перехода или также меткой и означает следующее. Вычисляется значение этой функции, которое принимается в качестве отдельно стоящей в строке метки; далее, как и в случае отдельно стоящей в строке метки, разыскивается строка алгоритма, помеченная меткой, равной этому значению, и совершается переход к выполнению этой строки. Разумеется, что возможными значениями адресной функции, составляющей отдельную строку алгоритма, должны быть метки, употребленные в алгоритме. Пусть, например, отдельную строку алгоритма составляет адресная функция.

$$\mathfrak{A} + 3$$

Если $\mathfrak{a} = 18$, $\mathfrak{16} = 4$, то такая строка будет означать переход к строке алгоритма с меткой 7.

Формула останова

Формула останова обозначается символом "!" или " Я " и может составлять отдельную строку алгоритма. Если при реализации алгоритма выполняется формула останова "!", то тем самым алгоритмический процесс заканчивается. Если выполняется формула останова Я , которой не соответствовал в алгоритме символ П /см.далее/, то также процесс заканчивается. Действие формулы Я , для которой соответствовал символ П, описано в пункте "Формулы вхождения и подпрограммы".

Формула обмена

Формулы обмена записываются с помощью символа $\Longrightarrow$ /двойной стрелки/ в виде

а ⟸⟹ в

Здесь а и в – адресные функции, воспринимаемые как адреса. Формула обмена означает обмен содержимых этих адресов. Например, если '$a = 3$, '$в = 1{,}1$, то после выполнения формулы

а ⟹⟸ в

получаем

$$'a = 1{,}1, \quad 'в = 3.$$

Формулы вхождения и подпрограммы

Формулы вхождения применяются для выполнения в ходе алгоритма того или иного преобразования – подпрограммы, которое непосредственно в алгоритме не описывается, но которое некоторым образом описано и, значит, может быть выполнено. Формула вхождения записывается в виде

$$\Pi\{a_1, a_2, \ldots, a_n\}\alpha, \beta$$

Здесь α и β – формулы безусловного перехода /в частном случае – просто метки/, из которых α – метка подпрограммы, β – метка некоторой строки алгоритма; $a_1, a_2, \ldots, a_n$ – список адресных функций. Формула означает:

а/ перейти к подпрограмме с меткой α ;

б/ первые выражения из списка $a_1, a_2, \ldots, a_n$ считать упорядоченным списком аргументов подпрограммы;

в/ выполнив подпрограмму, переслать упорядоченный список результатов в остальные адреса списка $a_1, a_2, \ldots, a_n$ формулы Π /число членов списка должно быть равно суммарному числу входных и выходных параметров данной подпрограммы/;

г/ перейти к строке алгоритма с меткой β .

Если эта строка является следующей строкой алгоритма, то вторая метка может быть опущена.

Подпрограммы, записанные в адресном языке

/что не обязательно, можно предполагать, что они записаны в каком-либо другом языке, например, в машинных кодах/, начинаются строкой адресных формул с "пустыми" /символ " ϕ "/ левыми частями; вместо них в ходе выполнения подпрограммы ставятся элементы списка из формулы вхождения. Конец работы подпрограммы обозначается символом δ . В этом случае формула вхождения означает переход к строке с меткой α , включение /с сохранением порядка/ списка адресных выражений $a_1, a_2, \ldots, a_n$, на места знаков пусто ϕ левых частей формул первой строки подпрограммы в замену символа δ меткой β /замены не в записи алгоритма, а в ходе его выполнения/.

Предикатные функции

Выражения вида $P\{Z\}$ называются предикатными функциями: здесь P - символ функции. Аргументом предикатной функции является Z - некоторое высказывание, т.е. выражение, о котором имеет смысл говорить, что оно либо истинно, либо ложно. Высказываниями будут, например, утверждения:

$$a \leqslant 'b \; ; \quad 'a < 'b \leqslant 'c \; ; \quad 'a \times 'b \leqslant n \; ; \quad 2 \cdot 'b + 2 = 'c$$

Если, например, $a = 1$; $'a = 1 = 10$; $'b = 3$; $'c = 8$; $n = 20$, то первое и четвертое выражения - истинны, а второе и третье - ложны.

Предикатная функция принимает только два значения - "истина" и "ложь", в зависимости от истинности или ложности своего аргумента, которые можно обозначить некоторыми символами, например, 1 и 0. Так, при определенных выше значениях букв предикатные функции

$$P\{a < 'b\} \; ; \quad P\{'b = 3\}$$

принимают значение "истина", а предикатные функции

$$P\{'b = 0\} \; , \quad P\{2 < 1 < 3\} \; , \quad P\{'a = 'b\}$$

значение "ложь".

Чертой над высказыванием обозначается отрицание высказывания, т.е. высказывание с чертой сверху означает ему противоположное высказывание. Противоположными высказываниями будут, например:

$$P\{a<b\} \text{ и } P\{a\geq b\}$$

Предикатные формулы

Выполнение многих алгоритмов зависит от условий, которые заранее не могут быть признаны истинными или ложными. Так бывает, когда в высказывание входят элементы, значение которых определяется в самом алгоритме. В этих случаях для изменения порядка выполнения алгоритма, в зависимости от выполнения необходимых условий /для разветвления вычислительного процесса/ используются предикатные формулы, которые записываются следующим образом

$$P\{z\}\alpha\,|\,\beta \quad \text{или} \quad P\{z\}^{\alpha}_{\beta}$$

где $P\{z\}$ – предикатная функция;

α и β – какие-либо из следующих выражений;

а/ формула останова;

б/ формула безусловного перехода /в частности, просто метка/;

в/ формула вхождения;

г/ одна или несколько адресных формул или формул обмена;

д/ строка адресных формул или формул обмена, за которой после знака "," следует формула останова или формула безусловного перехода.

Смысл предикатной формулы состоит в том, что она воспринимается как выражение α, если значение предикатной функции "истина", и как выражение β, если это значение "ложь".

Так, формула

$$P\{1<2\}^{2}a \Longrightarrow b \,|\, c \Longleftrightarrow a$$

эквивалентна формуле

$$'a \Longrightarrow 'b ,$$

а формула

$$P\{'a='c\}\,f-g \Longrightarrow \alpha \downarrow k$$

при определенных выше значениях букв эквивалентна формуле, состоящей из одной метки, т.е. означает переход на строку алгоритма, помеченную меткой K .

Если значение предикатной функции всегда "истина", то ее можно не писать, так же, как выражение β . В этом случае предикатная формула просто эквивалентна выражению α . Если предикатная формула всегда "ложь", то ее также можно не писать, а писать лишь выражение $\bar{P}$.

Если α или β – метка следующей строки алгоритма, то эту метку можно опускать, сохраняя, однако, разделительный знак " $\downarrow$ " на своем месте.

Формулы циклирования

Пусть требуется для ряда значений переменной x, размещенных по адресам

$$\alpha+1, \alpha+2, \ldots, \alpha+n , \qquad /1/$$

вычислить значение некоторой функции $f(x)$ и поместить в адреса

$$\beta+1, \beta+2, \ldots, \beta+n \qquad /2/$$

Адресная программа такого алгоритма может быть записана в виде

$$1 \Longrightarrow \pi$$
$$k\ldots\ P\{'\pi \le n\}\downarrow l$$
$$f('(\alpha+'\pi)) \Longrightarrow \beta+'\pi$$
$$'\pi+1 \Longrightarrow \pi$$
$$k$$

Для сокращения записи таких программ вводится формула циклирования

$$Ц\{a(b)\,c \Longrightarrow \pi\}\,\alpha,\beta ,$$

где Ц – символ формулы;
a – первое значение переменной;

ℓ – шаг ее изменения;

c – последнее значение;

– все значения переменной помещаются по адресу $\overline{\pi}$;

α – метка строки, по которой распространяется формула циклирования;

β – метка строки алгоритма, к которой следует перейти после выполнения данного цикла вычислений для всех значений переменной $\overline{\pi}$.

Та же программа вычислений с помощью формулы циклирования запишется в виде:

$$\mathrm{Ц}\{1(1)n \Longrightarrow \pi\}\, k, \ell$$
$$f('(\alpha + '\pi)) \Longrightarrow \beta + '\pi$$
$$k \ldots$$

В формуле циклирования в общем случае a, в, c могут быть адресными функциями, изменяющими свое значение в результате выполнения программы. Если цикл вычислений требуется повторить не до указанного значения, а вплоть до нарушения некоторого условия $P\{z\}$, то формула циклирования записывается в виде:

$$\mathrm{Ц}\{\alpha(b)\, P\{z\} == \pi\}\, \alpha, \beta$$

Например, для вычисления квадратов первых положительных чисел, размещенных в адресах /1/, вплоть до числа $x_i \leqslant \varepsilon$, можно написать программу

$$\mathrm{Ц}\{1(1)P\{'(\alpha + '\pi) \leqslant \varepsilon\} \Longrightarrow \pi\}\, k, \ell$$
$$('(\alpha + '\pi))^2 \Longrightarrow \beta + '\pi$$
$$k \ldots$$

/результаты помещаются в последовательность адресов /2/.

Допускаются следующие сокращения записи формулы циклирования.

Если формула циклирования распространяется только на одну строку алгоритма, то первая метка может опускаться, однако, знак ",", разделяющий метки формулы, сохраняется. Если после выполнения

формулы циклирования требуется перейти к строке алгоритма, непосредственно расположенной за последней строкой, на которую она распространена, то может опускаться вторая метка вместе со знаком ",", стоящим между метками. Если условие, до которого требуется продолжать перебор значений переменной π, называемой параметром цикла, определяется в цикле, то формулу циклирования можно записывать в виде:

$$Ц\{a(b) \Longrightarrow \pi\}$$

Областью действия формулы циклирования называются следующие строки программы: все строки, вплоть до строки с меткой α; если строка, помеченная меткой α /или первая строка после формулы циклирования, если метка α в формуле отсутствует/, является снова формулой циклирования или формулой вхождения, то их области действия также входят в область действия первой формулы циклирования.

Формула циклирования может быть помечена меткой. Если переход по метке к формуле циклирования совершен извне ее области действия, то перебор значений начинается с первого; если же переход осуществлен из области действия, то перебор продолжается.

Формулы замены

Формула замены имеет вид:

$$З\{a_1 \to b_1, a_2 \to b_2, \ldots, a_n \to b_n\}\alpha, \beta, s$$

где З - символ формулы замены; $a_1 \to b_1, \ldots, a_n \to b_n$ - список замен; α, β, s - метки.

По формуле замены строки алгоритма, начиная от строки, помеченной меткой α, по метку β, выполняются с заменой в них символов b_i на символы a_i. Замена производится только в ходе выполнения программы, запись остается неизменной.

После выполнения формулы замены совершается переход к строке с меткой S . Формула замены позволяет использовать готовые части алгоритма при других значениях переменных или даже символов операций. Например, если ранее вычислялась сумма векторов по программе

$$M\ldots\ \mathcal{U}\{1(1)n \Longrightarrow \bar{\pi}\}$$
$$N\ldots\ '(\alpha+'\pi)+'(\beta+'\pi) \Longrightarrow \gamma+'\pi$$

где $\alpha+i$, $\beta+i$ $(i=1,2,\ldots,n)$ – адреса компонент векторов, $\gamma+i$ $(i=1,2,\ldots,n)$ – адреса для компонент результирующего вектора, то формула замены

$$З\{-\rightarrow+\}M,N$$

будет означать вычисление разности векторов с помещением компонент результирующего вектора в адреса $\gamma+i$ $(i=1,2,\ldots,n)$.

Формула замены может быть использована для построения подпрограммы. Для этого вместо строки адресных формул с пустыми левыми частями подпрограмма начинается формулой замены, в списке замен которой в левых частях стоит тот же символ пусто $\emptyset$. Первая строка подпрограммы

$$\emptyset \Longrightarrow \varphi,\ \emptyset \Longrightarrow \psi$$

эквивалентна строке $З\{\emptyset \rightarrow '\varphi,\ \emptyset \rightarrow '\psi\}$

Первая же строка подпрограммы

$$З\{a \rightarrow \varphi, b \rightarrow \psi, c \rightarrow \delta\}$$

не может быть заменена строкой с адресными формулами; такая подпрограмма не может быть записана без формулы замены, если ее не подвергнуть существенным изменениям.

Адресный алгоритм

Адресный алгоритм или адресная программа состоит из конечного числа строк, записанных одна под другой. В каждой строке может стоять:

1. Одна или несколько формул обмена, замены или адресных.

2. Формула безусловного перехода.
3. Формула останова.
4. Формула вхождения.
5. Формула циклирования.
6. Предикатная формула.
7. Одна из предыдущих строк с одной или несколькими метками.
8. Пустая строка с одной или несколькими метками /т.е. пустая строка всегда должна быть помечена меткой; пустая строка означает переход к следующей строке/.

Строки алгоритма выполняются, начиная с некоторой строки, метка которой специально указывается. Если такого указания нет, то выполняется первая /верхняя/ в записи строка. После строки с адресными формулами или формулами обмена и после пустой строки выполняется следующая в записи строка. Формулы вхождения, предикатные, формулы безусловного перехода и формулы циклирования указывают себе преемниц. Формула останова означает конец работы алгоритма.

Алгоритм определяется заданием программы, исходного адресного отображения /заполнения адресов исходными данными/ и указанием на множество адресов, содержащих результаты вычислений. Работа алгоритма состоит в преобразовании адресного соответствия вплоть до получения окончательного результата по указанному множеству адресов.

Примеры. **Линейный алгоритм.** Если строки выполняются одна за другой, то алгоритм называется линейным. Вычислим значение величины

$$x = \frac{a+b}{c} \cdot d - \frac{e}{a+b} f ,$$

если дано адресное отображение и результат должен быть получен по адресу τ_1:

$'\alpha = a$ $\quad '\gamma = c$ $\quad \tau = e$

$'\beta = b$ $\quad '\delta = d$ $\quad \rho = f$

Алгоритм можно записать в виде

$$'\alpha + a \Rightarrow z_1 \; ; \; '\delta : '\gamma \Rightarrow z_2 \; ; \; '\tau \times 'p \Rightarrow z_3$$
$$'\beta + b \Rightarrow z_2 \; ; \; 'z_3 : 'z_1 \Rightarrow z_3 \; ; \; 'z_2 - 'z_3 \Rightarrow z_1$$

Разбиение алгоритма на отдельные строки в этом случае произвольно: можно принимать более крупные или более мелкие разбиения. Линейный адресный алгоритм можно записать одной формулой:

$$\frac{'\alpha + '\beta}{'\gamma} \times '\delta - \frac{'\tau}{'\alpha + '\beta} \times 'p \Rightarrow z_1 \; .$$

Алгоритмы с разветвлением. Если алгоритм зависит от некоторого условия $\mathcal{Z}$, то он реализуется с помощью предикатной функции $P\{\mathcal{Z}\}$. Вычислим, например, величину

$$y = \begin{cases} x^4 : (x^2+1), & \text{если} \quad x < 0 \\ x^4 \times (x^2+1), & \text{если} \quad x \geqslant 0 \end{cases}$$

Пусть $'\alpha = x$ и результат должен быть получен по адресу β. Алгоритм вычисления может быть записан в виде

$$B\ldots \; P\{'\alpha < 0\} \; ('\alpha)^4 : (('\alpha)^2+1) \Rightarrow \beta \; | \; ('\alpha)^4 \cdot (('\alpha)^2+1) \Rightarrow \beta$$

или в виде

$$B\ldots \; ('\alpha)^2 + 1 \Rightarrow z \; ; \; ('\alpha)^4 \Rightarrow z_1$$
$$P\{'\alpha < 0\} \; 'z_1 : 'z \Rightarrow \beta \; | \; 'z_1 \times 'z \Rightarrow \beta$$

Алгоритмы с использованием формул вхождения. Формула вхождения может применяться при многократном использовании одного участка программы /подпрограммы/ с переходом после этого на разные строки.

Составим программу вычисления значения функции

$$f(f(a) + f(b)), \qquad /3/$$

где $f(x) = \begin{cases} \cos x - 1, & \text{если} \quad x < 0 \\ x^2, & \text{если} \quad x \geqslant 0 \end{cases}$.

Предварительно составим подпрограмму вычисления значения функции f по заданному значению ее аргумента.

Пусть адрес ψ используется для хранения аргумента, а в адресе ψ будет находиться адрес, по которому выдается значение функции.

Подпрограмма может быть записана в виде:

$$f\ldots\ \overset{\phi\Rightarrow\varphi,\ \phi\Rightarrow\psi}{\underset{a}{P\{'\varphi<0\}\cos'\varphi-1\Rightarrow'\psi}}\ \ '\psi+('\varphi)^2\Rightarrow'\psi$$

Здесь знаками ϕ /"пусто"/ обозначены места для помещения в них формальных параметров подпрограммы. Первая строка подпрограммы означает, что обращение к подпрограмме производится по формуле вхождения $\Pi\{\alpha,\beta\}f$ с двумя фактическими параметрами α и β, при этом α /аргумент/ и β /адрес для помещения результата/ ставятся на места символов ϕ.

Эта же подпрограмма с применением формулы замены запишется в виде $f\ldots\ 3\{\phi-\varphi,\ \phi-\psi\}$

$$\underset{a}{P\{\varphi<0\}\cos\varphi-1\Rightarrow\psi};\quad\varphi^2\Rightarrow\psi$$

Алгоритм для вычисления значений функции /3/ запишется в виде:

$$\Pi\{a,\rho\}f$$
$$\Pi\{b,\rho_1\}f$$
$$'\rho+'\rho_1\Rightarrow\rho$$
$$\Pi\{'\rho,\rho\}f$$

Ответ - в адресе ρ. Алгоритм не зависит от того, к какой из двух подпрограмм, помеченных меткой f, мы обращаемся.

Алгоритм с циклом. Использование формулы циклирования позволяет участки алгоритма повторять нужное число раз. Пусть для всевозможных сочетаний параметров a, b, c, каждый из которых задан набором значений, размещенных соответственно по адресам

$$\alpha+1,\ \alpha+2,\ \ldots,\ \alpha+n;$$
$$\beta+1,\ \beta+2,\ \ldots,\ \beta+m;$$
$$\gamma+1,\ \gamma+2,\ \ldots,\ \gamma+p,$$

требуется вычислить значение функции от трех аргументов $f(a,b,c)$ и все результаты последовательно выдать на печать.

Пусть имеется подпрограмма вычисления значений этой функции f с меткой f, по которой по заданным значениям аргументов a, в, c может быть получено значение функции f. Формула вхождения на эту подпрограмму пусть имеет вид $\Pi\{a,b,c;\beta\}f$ что означает, что если a,в,c – аргументы функции, то, выполнив подпрограмму по адресу β, будет получено соответствующее значение $f/a,в,c/$.

Программа вычисления значений заданной функции с выдачей результатов на печать может быть записана в виде:

$$Ц\{1(1)n \Rightarrow \pi_1\}$$
$$Ц\{1(1)m \Rightarrow \pi_2\}$$
$$Ц\{1(1)p \Rightarrow \pi_3\}\,\alpha$$
$$\Pi\{'(\alpha+'\pi_1),'(\beta+\pi_2),'(\gamma+'\pi_3);d\}f$$
$$\alpha\ldots \quad \text{Печать } 'd$$
$$\beta$$

Формулы циклирования могут быть переставлены местами, в связи с чем изменится порядок выдачи результатов на печать.

Решение системы линейных алгебраических уравнений с симметричной матрицей коэффициентов усовершенствованным методом Гаусса

Пусть $A=\{a_{ij}\}$ $(i,j=1,2,\ldots,n,\ a_{ij}=a_{ji})$ – исходная матрица и $\mathcal{F}\{a_{i,n+1}\}$ – вектор правых частей.

Согласно данному методу алгоритм распадается на два этапа – "прямой ход" и "обратный ход".

Расчетные формулы

Прямой ход. Элементы расширенной матрицы $a_{ij}\{1\leq i\leq n;\ 1\leq j\leq n+1\}$ преобразуются последовательно по строкам по рекурентным формулам

$$\bar{a}_{1j}=a_{1j}\ ,\quad 1\leq j\leq n+1$$

$$\bar{a}_{ij}=a_{ij}-\sum_{k=1}^{i-1}\frac{\bar{a}_{ki}\cdot\bar{a}_{kj}}{\bar{a}_{kk}}\ ,\quad 2\leq i\leq n,\ i\leq j\leq n+1$$

Обратный ход. Обратный ход состоит в решении системы уравнений с треугольной матрицей $\{\bar{a}_{ij}\}$, $i=1,2,\ldots,n$; $i\leq j\leq n+1$, полученной в результате прямого хода. Расчетные формулы имеют вид:

$$x_i=\frac{\bar{a}_{i,n+1}-\sum_{k=i+1}^{n}\bar{a}_{ik}\,x_k}{\bar{a}_{ii}}\ ,\quad i=n,n-1,\ldots,1$$

Адресные программы на общеалгоритмическом уровне

Заметив, что вновь получаемым элементам $\bar{a}_{ij}$ могут быть поставлены в соответствие те же адреса, которые отводились элементам a_{ij} , и приняв, что

$$'\alpha_{ij}=a_{ij}\quad (i=1,2,\ldots,n;\ 1\leq j\leq n+1);$$

S_i – адреса, отведенные для компонент вектора решений

$$x_i\ (i=1,2,\ldots,n),$$

можем написать алгоритм прямого и обратного хода в виде.

Алгоритм прямого хода

$$U\{2(1)n\Rightarrow i\},\ 8$$
$$U\{'i(1)n+1\Rightarrow j\}$$
$$U\{1(1)'i-1\Rightarrow K\}$$
$$'\alpha_{ij}-\frac{\alpha_{ki}\cdot\alpha_{kj}}{\alpha_{kk}}\Rightarrow\alpha_{ij}$$

Исходное адресное соответствие

$'\alpha_{ij}=a_{ij}$ $(i=1,2,\ldots,n,$ $j=1,2,\ldots,n+1);$ $a_{ij}=a_{ji}$

Результативное множество адресов совпадает с исходным.

Алгоритм обратного хода

$U_i\{n(1)1 \Longrightarrow i\}$ M

$\alpha'_{i\,n+1} \Longrightarrow S_i$

$U_k\{i+1(1)n \Longrightarrow k\}$

$'S_i - \alpha_{ik} \cdot S_k \Longrightarrow S_i$

$\frac{'S_i}{\alpha_{ii}} \Longrightarrow S_i$

M...

Исходное адресное соответствие

$'\alpha_{ij} = \bar{a}_{ij}$

$(i = 1,2,\ldots,n;\ j = 1,2,\ldots,n+1;\ i \leq j)$

Результативное множество адресов

В такой записи алгоритмы, хотя и содержат точную постановку задачи, неудобны для их реализации на машинах в силу того, что здесь нами использовано двумерное множество адресов α_{ij} . В существующих же машинах множество адресов является линейным, обычно представляющим собой отрезок натурального ряда чисел.

Уровень условных адресов

Учитывая это, перейдем от множества двумерного α_{ij} к некоторому множеству линейному $\alpha+1, \alpha+2, \ldots$

Такой переход /кодирование/ может быть осуществлен различными способами, в связи с чем будут получаться различного вида алгоритмы.

Рассмотрим для примера два способа кодирования.

а/ Элементы исходной матрицы кодируются по столбцам в одну последовательность адресов, начиная с адреса $\alpha+1$, т.е., как легко заметить, имеет место следующее соответствие между элементами матрицы и их адресами:

$$a_{ij} = '(\alpha + (j-1)n + i)$$

Это позволяет переписать программы прямого и обратного хода в виде:

Прямой ход

$$U\{2(1)n \Rightarrow i\},\ 8*$$
$$U\{'i\,(1)\,n+1 \Rightarrow j\}$$
$$U\{1\,(1)\,'i-1 \Rightarrow \kappa\}$$
$$'(\alpha+('j-1)n+'i)-\frac{'(\alpha+('i-1)n+'\kappa)-'(\alpha+('j-1)n+'\kappa)}{'(\alpha+('\kappa-1)n+'\kappa)} \Rightarrow$$
$$\Rightarrow \alpha+('j-1)n+'i$$

Исходное адресное соответствие

$$a_{ij}='(\alpha+(j-1)n+i)$$
$$1\leqslant i\leqslant n\ ;\ i\leqslant j\leqslant n+1$$

Результативное множество адресов совпадает с исходным

Обратный ход....

$$U\{n(-1)1 \Rightarrow i\}M$$
$$'(\alpha+n^2+'i) \Rightarrow S+i$$
$$U\{'i+1\,(1)\,n \Rightarrow \kappa\}$$
$$'(S+i)-'(\alpha+('\kappa-1)n+'i)-'(S+'\kappa) \Rightarrow S+'i$$
$$\frac{'(S+'i)}{'(\alpha+('i-1)n+'i)} \Rightarrow S+'i$$
$$M...\ 8$$

Исходное адресное соответствие

$$a_{ij}='(\alpha+(j-1)+i)$$
$$1\leqslant i\leqslant n\ ;\ i\leqslant j\leqslant n+1$$

Результативное множество адресов

$$'(S+i)=x_i\ ,\ i=1,2,\ldots,n$$

б/ В связи с тем, что исходная матрица коэффициентов - симметрична, напрашивается способ кодирования, при котором в последовательность адресов по столбцам /по строкам/ кодируются лишь те элементы матрицы, для которых $i \leqslant j$ т.е. диагональные и наддиагональные элементы. Так, для $n=4$ принимается следующая таблица размещения исходных данных по адресам:

$$'(\alpha+1)=a_{11}\quad '(\alpha+2)=a_{12}\quad '(\alpha+4)=a_{13}\quad '(\alpha+7)=a_{14}\quad '(\alpha+11)=a_{15}$$
$$'(\alpha+3)=a_{22}\quad '(\alpha+5)=a_{23}\quad '(\alpha+8)=a_{28}\quad '(\alpha+12)=a_{25}$$
$$'(\alpha+6)=a_{33}\quad '(\alpha+9)=a_{34}\quad '(\alpha+13)=a_{35}$$
$$'(\alpha+10)=a_{44}\quad '(\alpha+14)=a_{45}$$

Нетрудно убедиться в том, что

$$a_{ij}=\left(\alpha+i+\frac{i}{2}(j-1)\right)$$

Поэтому программы, в которых мы для исключения их зависимости от параметров n, S и α. положили

$$'\alpha=n\cdot,\quad '\varphi=\alpha\quad,\quad '\psi=S$$

переписываются в виде:

Прямой ход ...

$$U\{2(1)''\varphi \Longrightarrow i\},s$$
$$U\{i(1)''\varphi+1\Longrightarrow j\}$$
$$U\{1(1)'i-1\Longrightarrow k\}$$
$$'(''\varphi+i+\frac{i}{2}('j-1))-$$
$$-\frac{'(''\varphi+'k+\frac{i}{2}(i-1))\cdot'(''\varphi+'k+\frac{i}{2}('j-1))}{'(''\varphi+\frac{'k}{2}('k+1)+'k)}\Longrightarrow$$
$$\Longrightarrow''\varphi+i+\frac{i}{2}('j-1)$$

Исходное адресное соответствие

$$'\alpha=n;\ '\varphi=\alpha$$
$$'(\alpha+i+\frac{i}{2}(j-1))=a_{ij}$$
$$(1\leqslant i\leqslant n;\ i\leqslant j\leqslant n+i)$$

Результативное множество адресов совпадает с исходным

Обратный ход ....

$$U\{''\varphi(-1)1\Longrightarrow i\}M,s$$
$$'(''\varphi+i+\frac{''\varphi+1}{2}\cdot''\varphi)\Longrightarrow''\psi+'i$$
$$U\{'i+1(1)''\varphi\Longrightarrow k\}$$
$$(''\psi+'i)-'(''\varphi+'i+\frac{'k}{2}('k-1))\cdot'(''\varphi+'k)\Longrightarrow''\psi+'i$$
$$\frac{'(''\psi+'i)}{'(''\varphi+\frac{'i('i+1)}{2})}\Longrightarrow''\psi+'i$$

Исходное адресное соответствие

$$''\varphi=\alpha;\ '\alpha=n;\ ''\psi=S$$
$$'(\alpha+i+\frac{i}{2}(j-1))=a_{ij}$$
$$(1<i<n;\ i\leqslant j\leqslant n+1)$$

Результат – по множеству адресов

$$S+i\ \ (i=1,2,\ldots,n)$$

Каждый из приведенных алгоритмов может быть оформлен в виде подпрограммы.

При принятом способе кодирования исходной матрицы и ее порядка для получения подпрограмм остается только к приведенным программам, соответственно, прибавить в начале строки

ПРЯМОЙ ХОД ..... $\phi \Rightarrow \varphi$

ОБРАТНЫЙ ХОД .... $\phi \Rightarrow \varphi$, $\phi \Rightarrow \psi$

/к которым отнесены начальные метки/.

Формулы вхождения на эти подпрограммы будут иметь вид:

$\Pi\{\alpha\}$ ПРЯМОЙ ХОД

$\Pi\{\alpha,S\}$ ОБРАТНЫЙ ХОД

Еще раз подчеркнем, что как запись формул вхождения, так и запись алгоритмов согласуется со способом кодирования информации.

Адресное программирование : построение программ в кодах конкретных машин

Для перевода адресных программ в программы конкретной машины могут быть разработаны формальные правила, учитывающие специфику выполнения операций данной машиной. Проиллюстрируем это на примере перевода адресных формул в программы для машины "Урал" /описание машины "Урал" см.в Б.Б.Гнеденко, В.С.Королюк, Е.Л.Ющенко "Элементы программирования", Физматгиз, 1961, а также Китов А.П. и Криницкий Н.А. "Электронные цифровые машины и программирование", Физматгиз, 1959/. Приведенная далее таблица соответствия может быть использована для построения программ, переводящих адресные программы в программы для "Урала". Такого рода программирующие программы работают в ВЦ АН УССР на машинах "Киев" и "Урал".

Можно показать, что произвольный адресный алгоритм может быть представлен в эквивалентной

ему форме, в которой в адресных функциях, входящих в алгоритм, операция-штрих встречается над результатом, содержащим эту операцию не более одного раза /такие адресные функции будем называть функциями ранга не выше второго/.

Например, адресные формулы

$$^{n}\alpha \Rightarrow \beta \qquad (n \geq 3)$$

или

$$\alpha \Rightarrow {}^{m}\beta \qquad (m \geq 2)$$

могут быть записаны соответственно в виде

$$\left.\begin{array}{l} {}^{2}\alpha \Rightarrow \beta \\ {}^{2}\beta \Rightarrow \beta \\ \vdots \\ {}^{2}\beta \Rightarrow \beta \\ {}^{2}\beta \Rightarrow \beta \end{array}\right\} \; n-2 \text{ раза} \qquad \left.\begin{array}{l} {}^{2}\beta \Rightarrow \omega \\ {}^{2}\omega \Rightarrow \omega \\ \vdots \\ {}^{2}\omega \Rightarrow \omega \\ \alpha \Rightarrow {}'\omega \end{array}\right\} \; m-2 \text{ раза}$$

Таким образом, задача построения адресных формул на "Урале" может быть разбита на две части:

а/ получение значения адресной функции на сумматоре S : $f \Rightarrow S$;

б/ перенос содержимого сумматора S по адресу, равному значению некоторой адресной функции f:

$${}'S \Rightarrow f$$

Наиболее существенной особенностью набора операций "Урала" является возможность выполнения на нем операций по адресам второго ранга,[x] например, операций:

$${}'({}'\alpha + a) \Rightarrow S, {}^{2}\alpha \Rightarrow S \text{ или } {}'S \Rightarrow {}'\alpha + a, \; {}'S \Rightarrow {}'\alpha \qquad /1/$$

лишь в том случае, когда α является регистром команд K, а именно, имеет место следующее соответствие приведенных адресных программ /1/ программам "Урала":

---

x/ не считая случая использования групповых операций.

Адресная программа	Эквивалентная ей адресная программа, реализуемая "Уралом"	Программа в кодах "Урала"
1	2	3
1. $'('\alpha + \beta) \Rightarrow S$	$'\alpha \Rightarrow K$ $('K+\beta) \Rightarrow S$	30 α 02 β
2. $'\alpha \Rightarrow S$	$'\alpha \Rightarrow K$ $'K \Rightarrow S$	30 α 02 0000
3. $'S \Rightarrow '\alpha + a$	$'\alpha \Rightarrow K$ $'S \Rightarrow K+a$	30 α 16 a
4. $'S \Rightarrow '\alpha$	$'\alpha \Rightarrow K$ $S \Rightarrow 'K$	30 α 16 0000

Для упрощения перехода от адресных программ к программам "Урала" в средней колонке мы приводим эквивалентные исходным адресные программы, реализуемые машиной.

Рассмотрим ряд типовых примеров. Пусть θ обозначает любую из операций "Урала": 01, 03, 04, 06, 07, 10, 12, 13, 14.

а/ получение значений адресных функций на сумматоре:

1	2	3
1. Константу a заслать на сумматор S $a \Rightarrow S$	$a \Rightarrow S$	20 a
2. Содержимое адреса α заслать на сумматор S $'\alpha \Rightarrow S$	$'\alpha \Rightarrow S$	02 α
3. Выполнить одну из операций θ, первым аргументом которой является константа a, а вторым - содержимое адреса β $a\,\theta\,'\beta \Rightarrow S$	$a \Rightarrow S$ $'S\,\theta\,'\beta \Rightarrow S$	20 a θ β

4. То же, что и в 3, но аргументы переставлены местами. В случае перестановочной операции следует аргументы переставить местами и применять п.3; в противном случае, следует воспользоваться данной программой

$'\beta\theta\alpha \Rightarrow S$

$\alpha \Rightarrow S$ 20 α
$'S \Rightarrow \omega$ 16 ω
$'\beta \Rightarrow S$ 02 β
$'S\theta'\omega \Rightarrow S$ θ ω

5. Выполнить одну из операций θ, первым аргументом которой является константа α, а вторым - содержимое адреса β по второму рангу

$\alpha\,\theta^2{}'\beta \Rightarrow S$

$\alpha \Rightarrow S$ 20 α
$'\beta \Rightarrow K$ 30 β
$'S\theta^2K \Rightarrow S$ θ 0000

6. $\alpha\,\theta'('\beta + c) \Rightarrow S$

$\alpha \Rightarrow S$ 20 α
$'\beta \Rightarrow K$ 30 β
$'S\theta'('K+C) \Rightarrow S$ θ C

7. Выполнить одну из операций θ, аргументом которых - содержимое адресов α и β

$'\alpha\,\theta'\beta \Rightarrow S$

$\alpha \Rightarrow S$ 02 α
$'S\theta'\beta \Rightarrow S$ θ β

8. Выполнить одну из операций θ, первый аргумент которой - содержимое адреса α, а второй - содержимое по адресу второго ранга β

$'\alpha\,\theta^2\beta \Rightarrow S$

$'\alpha \Rightarrow S$ 02 α
$'\beta \Rightarrow K$ 30 β
$'S\theta^2K \Rightarrow S$ θ 0000

9. $^2\alpha\,\theta'\beta \Rightarrow S$

$\alpha \Rightarrow K$ 30 α
$^2K \Rightarrow S$ 02 0000
$'S\theta'\beta \Rightarrow S$ θ β

10. $'('\alpha + \beta)\,\theta'c \Rightarrow S$

$'\alpha \Rightarrow K$ 30 α
$'('K+\beta) \Rightarrow S$ 02 β
$'S\theta'C \Rightarrow S$ θ C

11. Выполнить одну из операций θ оба аргумента которой – содержимые адресов второго ранга a и b

$^{2}a\ \theta\ ^{2}b \Rightarrow s$

$'a \Rightarrow K$	30	a
$^{2}K \Rightarrow S$	02	0000
$'b \Rightarrow K$	30	b
$'S\ \theta\ ^{2}K \Rightarrow S$	θ	0000

12. $''('a + b)\theta\,''('c + d) \Rightarrow S$

$'a \Rightarrow K$	30	a
$''('K + b) \Rightarrow S$	02	b
$'b \Rightarrow K$	30	c
$'S\,\theta\,''('K + d) \Rightarrow S$	θ	d

13. Содержимое по адресу третьего ранга заслать на сумматор

$^{3}\alpha \Rightarrow S$

$'\alpha \Rightarrow K$	30	α
$^{2}K \Rightarrow K$	30	0000
$^{2}K \Rightarrow S$	02	0000

14. Содержимое по адресу n-го ранга заслать на сумматор

$^{n}\alpha \Rightarrow S$

$'\alpha \Rightarrow K$	30	α
$^{2}K \Rightarrow K$	n-2 раза	300000
⋮		⋮
$^{2}K \Rightarrow K$		300000
$^{2}K \Rightarrow S$		020000

15. $''(''('\alpha + a) + b) \Rightarrow S$

$'\alpha \Rightarrow K$	30	α
$''('K + a) \Rightarrow K$	30	a
$''('K + b) \Rightarrow S$	30	b
$^{2}K \Rightarrow S$	02	0000

16. $''('\alpha + '\beta) \Rightarrow S$

$'\alpha \Rightarrow S$	02	α
$'S + '\beta \Rightarrow S$	01	β
$'S \Rightarrow \omega$	16	ω
$'\omega \Rightarrow K$	30	ω
$^{2}K \Rightarrow S$	02	0000

17. $''('\alpha + \beta + a) \Rightarrow S$

$\alpha \Rightarrow S$	02	α
$S + \beta \Rightarrow S$	01	β
$S \Rightarrow \omega$	16	ω
$\omega \Rightarrow K$	30	ω
$''('K + a) \Rightarrow S$	02	a

18. $c + {}'c\,{}'b \Rightarrow s$

$c \Rightarrow s$	20	c
${}'a \Rightarrow r$	17	a
${}'r\,{}'b + {}'s \Rightarrow s$	05	b

r -регистр арифметического устройства

19. ${}'a\,{}'b + {}'c \Rightarrow s$

${}'c \Rightarrow s$	02	c
${}'a \Rightarrow r$	17	a
${}'r\,{}'b + {}'s \Rightarrow s$	05	b

20. ${}'a\,{}'b + {}'c\,{}'d \Rightarrow s$

I вариант	II вариант	I вариант	II вариант
${}'a \Rightarrow s$	$0 \Rightarrow s$	02 a	20 0000
${}'s\,{}'b \Rightarrow s$	$a \Rightarrow r$	06 b	17 a
${}'c \Rightarrow r$	${}'r\,{}'b + {}'s \Rightarrow s$	17 c	05 b
	${}'c \Rightarrow r$	05 d	17 c
${}'r\,{}'d + {}'s \Rightarrow s$	${}'r\,{}'d + {}'s \Rightarrow s$		05 d

б/ перенос содержимого сумматора S по адресу f

$${}'s \Rightarrow f \quad ,$$

где f - некоторая адресная функция.

1	2	3
1. Содержимое сумматора заслать по адресу a ${}'s \Rightarrow a$	${}'s \Rightarrow a$	16 a
2. Содержимое сумматора S заслать по адресу, равному ${}'a + b$ $s \Rightarrow {}'a + b$	${}'a \Rightarrow k$ ${}'s \Rightarrow {}'k + b$	30 a 16 b

3. Содержимое сумматора S заслать по адресу, содержащемуся в адресе a

$'S \Rightarrow {}'a$

$'a \Rightarrow K$	30	a
$'S \Rightarrow {}'K$	16	0000

4. $'S \Rightarrow ('a + b) + c$

$'a \Rightarrow K$	30	a
$'(K+b) \Rightarrow K$	30	b
$'S \Rightarrow {}'K + c$	16	c

5. $'S \Rightarrow {}'a + {}'b + c$

$'S \Rightarrow \omega_1$	16	ω_1
$'a \Rightarrow S$	02	a
$'S + {}'b \Rightarrow S$	01	b
$'S \Rightarrow \omega_2$	16	ω_2
$'\omega_1 \Rightarrow S$	02	ω_1
$'\omega_2 \Rightarrow K$	30	ω_2
$'S \Rightarrow {}'K + c$	16	c

6. $'S \Rightarrow {}^2a$

$'a \Rightarrow K$	30	a
$^2K \Rightarrow K$	30	0000
$'S \Rightarrow {}'K$	16	0000

БФ 07036 Подписано к печати и в свет 23.УШ-1962 г.
Об'ем 2 печ. листа т. 450 зак. 162

Киев, ул. Шота Руставели, 3, ОКМП МСС ЦСУ УССР

Научные руководители

кандидат физико-математических наук
МИХАЛЕВИЧ В.С.

кандидат технических наук
БЕРНАРДО дель РИО

Ответственные за выпуск

от КДНТП КОЗАЧКОВ Л.С.

от Дома техники ЮЗЖД
ШУЛЬГАЙЦЕР Л.М.

Редактор ФРИДМАН С.А.

Корректор БАТУК В.А.

Цена для участников семинара 15 коп. Тир. 600
Первый завод 450


+ 5 8
1 3
: 0
9 × 1
4 − 3
6 : 5
2
4 × 2 7 8 +
1 9 4
3 : 6 ×
+ 7
4 ×