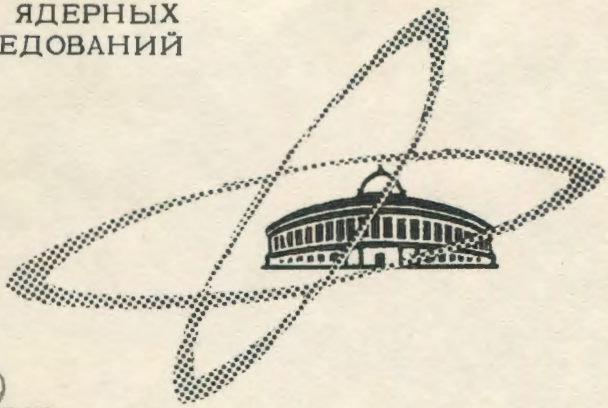
ОБЪЕДИНЕННЫЙ
ИНСТИТУТ
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ

Дубна

1668



В.И. Шаронов

АЛГОРИТМИЧЕСКИЙ ЯЗЫК ДЛЯ ОПЕРАЦИЙ НАД СЛОВАМИ, ОПИРАЮЩИЙСЯ НА АЛГОЛ-60

ВЫЧИСЛИТЕЛЬНЫЙ ЦЕНТР

1964

В.И. Шаронов

АЛГОРИТМИЧЕСКИЙ ЯЗЫК ДЛЯ ОПЕРАЦИЙ НАД СЛОВАМИ, ОПИРАЮЩИЙСЯ НА АЛГОЛ-60

Дубна 1964

В настоящее время известен ряд задач, к которым трудно применять существующие алгоритмические языки для описания вычислительных процессов (примером языка такого рода может служить АЛГОЛ-60 [1], [2]).

К таким задачам можно отнести: задачи формального преобразования формул - формальное дифференцирование и интегрирование и др.; задачи моделирования процессов творчества (перевод с одного языка на другой, композиция музыки, сочинение стихов на ЭВМ и др.); некоторые задачи теории игр и т.п.

В то же время существующие алгоритмические языки для задач такого типа либо удалены от общепринятых методов описания алгоритмов, как, например, адресное программирование [3], либо не обладают достаточно гибким аппаратом, как, например, языки строк АЛГОЛ-60 [4], [5], что в значительной мере снижает эффективность применения этих языков и затрудняет обозримость программ, написанных на таких языках.

Цель настоящей работы - построить алгоритмический язык, по возможности свободный от этих недостатков.

Построение предлагаемого языка велось на основе языка АЛГОЛ-60 при помощи добавления к нему новых понятий. (Синтаксис и семантика предлагаемого языка не изменяют синтаксиса и семантики языка АЛГОЛ-60).

Таким образом, предлагаемый язык является некоторым расширением языка АЛГОЛ-60. Расширение было выбрано так, что любая программа, написанная на языке АЛГОЛ-60, является программой на предлагаемом языке. Для того, чтобы сделать программу на предлагаемом языке программой на языке АЛГОЛ-60, необходимы некоторые преобразования исходной программы, которые могут быть формализованы (см. заключение). В отличие от работ [4], [5] уточнение понятия строк не производится.

Отличия предлагаемого языка от языка АЛГОЛ-60 состоят в том, что введены:

- тип word для переменных, принимающих "буквенное" значение; описания для таких переменных и массивов и возможность указания в них количества символов в словах;
- описания ранга для слов, определяющие операции отношения между словами;
- операция $\in$, позволяющая определять вхождение символов и совокупностей символов в другие совокупности символов;
- операции соединения слов $\&$ и $\vdash$;
- оператор сдвига;
- оператор засылки;

- ряд новых стандартных функций;
- новые виды скобок ⊄ ⊅, { } и разделитель * .

§ I. Формальное описание вводимых понятий

Нумерация разделов этого параграфа совпадает с нумерацией разделов сообщения об АЛГОЛ-60 [I] . В разделах, обозначенных римскими цифрами, даются неформальные разъяснения соответствующих разделов формального описания. В разделах, соответствующих разделам "Сообщения о языке АЛГОЛ-60", описываются дополнения к этим разделам, предлагающие сохранение существующего содержания этих разделов в "Сообщении о языке АЛГОЛ-60".

2.3. ОГРАНИЧИТЕЛИ

<операция> ::= <операция соединения> | <операция отношения для слов>

<операция соединения> ::= & | ↓

<операция отношения для слов> ::= ∈

<операция следования> ::= shift | send

<разделитель> ::= * | in | left | right

<скобка> ::= ⊄ | ⊅ | { | }

<описатель> ::= word | rank

Перевод вводимых слов см.[I)]

2.8. ЗНАЧЕНИЯ, КЛАССЫ ЗНАЧЕНИЙ И ТИПЫ

Значением является некоторое упорядоченное множество простых или составных слов (специальный случай - одно простое или составное слово). Различаются следующие классы значений: числа, логические значения, метки, простые слова и составные слова.

Значения типа word относятся либо к классу простых слов, либо к классу составных слов.

2.9. СЛОВА

2.9.I. С и н т а к с и с

<элемент типа адрес> ::= {<идентификатор>}

<пустой элемент> ::= <пусто>

<элемент правильного слова> ::= <любой основной символ, отличный от ⊄ или ⊅ или { или } или *> | <элемент типа адрес>

<правильное слово> ::= <элемент правильного слова> | <правильное слово><элемент правильного слова>

<пустое слово> ::= <пустой элемент> | <пустое слово><пустой элемент>

<элемент простого слова> ::= <элемент правильного слова> | <пустой элемент>

I) shift - сдвиг, send - заслать, in - в, left - влево, right - вправо, rank - ранг, word - слово (англ.).

<простое слово> :: = <правильное слово> | <пустое слово> | <правильное слово>
<пустое слово>
<элемент составного слова> :: = <простое слово>
<составное слово> :: = <элемент составного слова> * <элемент составного слова> |
<составное слово> * <элемент составного слова>
<элемент слова> :: = <элемент простого слова> | <элемент составного слова>
<слово> :: = <простое слово> | <составное слово>

2.9.2. Примеры

```
alpha
if x=0 then A else C
this is a word * termin 2 * 128
 {a12}{a13} {a14}
      * *
ab *cd *e* {x} i
```

2.9.3. Семантика

Слова применяются при построении первичных буквенных выражений. Элемент типа адрес применяется для запоминания в значениях слов адреса переменной. Количество элементов в словах задается при помощи описаний (см. разд.5.I.). На последних позициях в простых словах могут находиться пустые элементы.

2.9.4. Типы

Все слова имеют тип **word** .

3. ВЫРАЖЕНИЯ

<выражение> :: = <буквенное выражение>

3.2. ФУНКЦИИ

3.2.2. Примеры

```
subs ( ¢ ab ¢ , 2, 3, A[i,k] )
fix (¢a112¢)
pt (i,k+m, R & T)
```

3.2.3. Семантика

Функция определяет одно буквенное значение - слово.

3.2.4. Стандартные функции

pt (A1, A2, W) I) - для выделения части слова - значения выражения W , начиная с элемента[2] с номером A1 и кончая элементом о номером A2. Первые два параметра должны восприниматься как параметры, имеющие

спецификацию integer , третий параметр - как имеющий спецификацию word . Значение функции - выделенная часть слова. Если значение выражения A1 больше значения выражения A2, то значение функции не определено. Число элементов в значении функции определяется по формуле A2-A1+1.

opt(A1,A2,W) [3] - для выделения элемента с номером A1 из элемента с номером A2 в составном слове - значения выражения W . Первые два параметра должны восприниматься как имеющие спецификацию integer , третий - спецификацию word . Значение функции - выделенный элемент; функция не определена, если значение выражения W относится к классу простых слов.

subs(W1,A1,A2,W2) [4] - для замены в значении буквенного выражения W2 совокупности элементов, начиная с элемента с номером A1 и кончая элементом с номером A2, на значение буквенного выражения W1 .

Первый и четвертый параметры должны восприниматься как имеющие спецификацию word , второй и третий - спецификацию integer .

Если значение выражения A1 больше значения выражения A2, то значение функции не определено. Замена понимается в смысле операции объединения, если значение выражения W2 относится к классу простых слов и в смысле операции присоединения - в противном случае. Если значения выражений A1 и A2 оба выходят за пределы слова - значения выражения W2 , то значение функции равно значению выражения W2 с присоединенным к нему (в указанном выше смысле) значением выражения W1 слева, если значение выражения A2 не больше нуля, и справа, если значение выражения A1 больше числа элементов в значении выражения W2 .

Число элементов в значении функции определяется следующим образом:

---

1) от part - часть (сноска на стр.5)

2) здесь и далее всюду, где специально не оговорено, под понятием "элемент" подразумевается понятие "элемент слова", определенное в разд.2.9.1. (см.стр.5)

3) от option - выбор

4) от substition - замена (англ.)

- Если значения выражений W1 и W2 относятся к одному и тому же классу значений, то оно равно значению выражения

(if A1 < 1 then 0 else if A1 > A then A else A1-1)+B+
(if A2 < 1 then A else if A2 ≤ A then A - A2 else 0) ,

где: A - число элементов в значении выражения W2 , B - число элементов в значении выражения W1 .

- Если значение выражения W1 относится к классу простых слов, а значение выражения W2 - к классу составных слов, то оно определяется по предыдущей формуле, где вместо B следует поставить I.

- Если значение выражения W1 относится к классу составных слов, а значение выражения W2 - к классу простых слов, то оно равно числу элементов в значении W1 .

Максимальная длина элемента в значении функции (если оно относится к классу составных слов) определяется по правилам раздела 3.6.4.

extr (A,W) I) - для извлечения значения переменной, входящей в значение выражения W как элемент типа адрес (элемент с номером A). Первый параметр должен восприниматься как имеющий спецификацию integer , второй - спецификацию word .

Значение функции не определено:

- Если значение функции pt(A,A,W) не является элементом типа адрес.

- Если соответствующая данному элементу переменная не является переменной типа word .

extrreal (A,W) - для извлечения значения переменной, соответствующей элементу типа адрес в том случае, когда тип этой переменной real .

extrint (A,W) - для извлечения значения переменной, соответствующей элементу типа адрес в том случае, когда тип этой переменной integer .

fix (W) 2) - для преобразования значения буквенного выражения W , относящегося к классу простых слов, в элемент типа адрес. Значение функции не определено, если значение аргумента не является идентификатором.

I) от extraction - извлечение (англ.)

2) от fix - фиксировать (англ.)

3.2.5. Функции преобразования

Рекомендуется добавить к стандартным функциям функцию valoir(W) , преобразующую значение буквенного выражения W , состоящее из одних цифр, в положительное число типа integer , и обратную ей функцию corde(E) , преобразующую арифметические выражения типа integer в простое слово, состоящее из соответствующих цифр.

Ш.П. Приведем примеры, поясняющие, как получаются значения стандартных функций (о длине значений переменных см. разд. 5.I.).

ПРИМЕРЫ

Букв. вырах.	Значение букв.вырах.	Длина знач.букв. выражения	Букв. вырах.	Значен. букв. вырах.	Длина знач.букв. выражения	Функция	Значение функции	Длин. знач. функ.
A	abcd«	5	8	–	–	pt(1,4,A)	abcd	4
A1	ab«cd«ef«iq	4:2	–	–	–	pt(2,3,A1)	cd«ef	2:2
A2	abcdef	6	–	–	–	pt(0,2,A2)	ab	3
						pt(0,0,A2)	пустое слово*)	1
						pt(6,7,A2)	f	2
						pt(9,11,A2)	пустое слово*)	3
						pt(-7,-5,A2)	пустое слово*)	3
B	ab«cd«ef	3:2	–	–	–	opt(2,1,B)	b	1
						opt(3,2,B)	пустое слово*)	1
						opt(2,3,B)	f	1
C	abcde	5	D	ab	2	subs(D,3,3,C)	ababde	6
						subs(D,7,7,C)	abcdeab	7
						subs(D,-5,-5,C)	ababcde	7
						subs(‹W›,2,2,C)	aWcde	5
						subs(‹›,1,1,C)	bcde	5
			D1	{x1}	1	subs(D1,3,3,C)	ab{x1}de	5
			D2	a«b«c	3:1	subs(D2,2,2,C)	aacde«b«c	3:5
C1	ab«cd«ef	3:2	D3	uvw	3	subs(D3,2,2,C1)	ab«uvw«ef	3:3
			D4	xy«uz	2:2	subs(D4,2,2,C1)	ab«xy«uz«ef	4:2
			D5	x	1	subs(D5,0,0,C1)	x«ab«cd«ef	4:2
			–	–	–	subs(‹›,2,2,C1)	ab«« ef	3:2
E	abcy	4	F	kl	2	subs(F,2,3,E)	akly	4
						subs(F,-2,7,E)	kl	2
						subs(F,-6,1,E)	klbcy	5
						subs(F,2,8,E)	akl	3
						subs(F,8,10,E)	abcykl	6
			F1	k«l«m	3:1	subs(F1,3,4,E)	abk«lmm	3:3

*) Пустым словом называется простое слово, состоящее из пустых элементов. Количество элементов в пустом слове определяется в каждом случае содержательно.

Букв. вырaж.	Значение букв.вырaж.	Длина знач.букв. выражения	Букв. вырaж.	Значен. букв. вырaж.	Длина знач.букв. выражения	Функция	Значение функции	Длин. знач. функ.
			-	-	-	subs(≮{x1}≯,3,4,E)	ab{x1}	3
			-	-	-	subs(≮ ≯,2,4,E)	a	2
E1	xy*uw*zv	3:2	F2	a*b*c	3:1	subs(F2,2,3,E1)	xy*a*b*c	4:2
			F3	abc	3	subs(F3,1,2,E1)	abc*zv	2:3
			-	-	-	subs(≮ ≯,1,2,E1)	*zv	2:2
G	a1	2	-	-	-	fix(G)	{a1}	1
G1	x61	3	-	-	-	fix(G1)	{x61}	1
H	ab{y1}cd	5	y1	xy*zw	2:2	extr(3,H)	xy*zw	2:2
			y1	285	число	extrint(3,H)	285 (число)	-
			y1	2.07	число	extrreal(3,H)	2.07 (число)	-
I	02854	5	-	-	-	valoir(I)	2854 (число)	-
I1	1247	число	-	-	-	corde(I1)	1247 (слово)	4

3.4. БУЛЕВСКИЕ ВЫРАЖЕНИЯ

3.4.1. С и н т а к с и с

<операция отношения для слов> ::= ∈

<отношение> ::= <буквенное выражение> <операция отношения>

<буквенное выражение> | <буквенное выражение> <операция отношения для слов>

<буквенное выражение>

3.4.2. П р и м е р ы

≮ab≯ ∈ S

N2 = ≮acf≯

M <Q

D(C ≠ A) B∧pt(1,k,L1)> R

3.4.3. С е м а н т и к а

В случае отношений с обычными операциями отношения значение отношения не определено, если значения буквенных выражений в левой и правой частях отношения относятся к разным классам значений. Числа элементов в значениях левой и правой частей отношения считаются одинаковыми (если это не выполнено, то они уравниваются путем фиктивного добавления нужно-

го количества пустых элементов к слову меньшей длины). Такое же правило соблюдается и в отношении максимальных длин элементов данных значений.

Значения буквенных выражений, входящих в отношения с обычными операциями отношения, сравниваются поэлементно слева направо. Первая же пара неравных между собой элементов дает отношению значения true , false либо неопределенное значение. Отношения между неравными элементами слова определяются при помощи описаний ранга (см.разд.5.5). Если неравные элементы не входят в описания ранга, то в зависимости от операций отношения значение соответствующего отношения будет равно:

для операций $< \leqslant \geqslant >$ неопределенное значение

для операции $=$ false

для операции $\neq$ true

Операции отношения для слов $\in$ позволяет установить отношение между словами без дополнительных описаний. Отношение с данной операцией имеет значение true в том и только том случае, когда имеется хотя бы одно вхождение значения буквенного выражения - левого операнда в значение буквенного выражения - правого операнда. При выполнении данной операции выравнивание длин в значениях соотносящихся выражений не производится. При определении вхождения звездочка в значениях выражений должна восприниматься как элемент соответствующего слова - значения соотносящегося выражения, а пустой элемент - как символ, отличный от всех основных символов.

II.IУ. Рассмотрим несколько более подробно, как происходит нахождение значения отношения W1$\in$ W2 . Пусть α и β символы, отличные от всех основных символов. Заменим в значениях W1 и W2 звездочку всюду, где она встречается, символом α , а пустой элемент символом β . Пусть после такой замены в левой части получилось m элементов, а в правой -n . Тогда значение отношения W1$\in$ W2 совпадает со значением булевской переменной B, получающимся в результате работы следующей программы на предлагаемом языке:

```
B := false;
for i := 0 step 1 until n-m do
    begin B := B∨W1 = pt(i+1,i+m,W2);
        if B then  go to  значение отношения найдено.
    end
```

В последующих примерах мы будем исходить из наличия стандартных описаний ранга (см.разд.5.5.2.).

отношение	значение отношения
‹abcde› =‹abcdf›	false

отношение	значение отношения
⟨abcde⟩ ≤ ⟨abcdf⟩	true
⟨abcd⟩ = ⟨abc{x}y⟩	false
⟨abc⟩ ∈ ⟨ambabcpq⟩	true
⟨ab*cd⟩ < ⟨ab*sz⟩	true
⟨am*ef⟩ < ⟨ak*ab⟩	false
⟨stl⟩ > ⟨tls⟩	false
⟨⟩ ∈ ⟨**a⟩	true
⟨⟩ ∈ ⟨ab⟩	false

3.6. БУКВЕННЫЕ ВЫРАЖЕНИЯ

3.6.1. Синтаксис

⟨первичное буквенное выражение⟩ ::= ⟨⟨слово⟩⟩ | ⟨переменная⟩ | ⟨функция⟩ | (⟨буквенное выражение⟩)

⟨буквенный терм⟩ ::= ⟨первичное буквенное выражение⟩ | ⟨первичное буквенное выражение⟩ & ⟨буквенный терм⟩

⟨простое буквенное выражение⟩ ::= ⟨буквенный терм⟩ | ⟨простое буквенное выражение⟩ ↓ ⟨буквенный терм⟩

⟨буквенное выражение⟩ ::= ⟨простое буквенное выражение⟩ | ⟨условие⟩ ⟨простое буквенное выражение⟩ else ⟨буквенное выражение⟩

3.6.2. Примеры

Первичные буквенные выражения

```
N [i,k]
⟨basic⟩
subs(A1 & A2,1,1,X)
(if x=0 then A & B else A & C)
⟨25⟩
⟨⟩
```

Буквенные термы

```
⟨128⟩
⟨**⟩ & Q & ⟨**⟩ & L & ⟨**⟩
(R1 ↓ R2 & G & (if B then ⟨x⟩ else ⟨y⟩)) & A
B[i] & ⟨{a52}⟩ & pt(i,j,C)
```

Простые буквенные выражения

```
A1 & A2 & A3 & A4 ↓ ⟨number⟩
(if A[y+2,x]>⟨a⟩↓T then ⟨2⟩ else ⟨3⟩) ↓ T1
```

го количества пустых элементов к слову меньшей длины). Такое же правило соблюдается и в отношении максимальных длин элементов данных значений.

Значения буквенных выражений, входящих в отношения с обычными операциями отношения, сравниваются поэлементно слева направо. Первая же пара неравных между собой элементов дает отношению значения true , false либо неопределенное значение. Отношения между неравными элементами слова определяются при помощи описаний ранга (см.разд.5.5). Если неравные элементы не входят в описания ранга, то в зависимости от операций отношения значение соответствующего отношения будет равно:

для операций $< \leq > \geq$ неопределенное значение

для операции $=$ false

для операции $\neq$ true

Операция отношения для слов $\in$ позволяет установить отношение между словами без дополнительных описаний. Отношение с данной операцией имеет значение true в том и только том случае, когда имеется хотя бы одно вхождение значения буквенного выражения - левого операнда в значение буквенного выражения - правого операнда. При выполнении данной операции выравнивание длин в значениях соотносящихся выражений не производится. При определении вхождения звездочка в значениях выражений должна восприниматься как элемент соответствующего слова - значения соотносящегося выражения, а пустой элемент - как символ, отличный от всех основных символов.

П.IУ. Рассмотрим несколько более подробно, как происходит нахождение значения отношения W1$\in$ W2 . Пусть α и β символы, отличные от всех основных символов. Заменим в значениях W1 и W2 звездочку всюду, где она встречается, символом α , а пустой элемент символом β . Пусть после такой замены в левой части получилось m элементов, а в правой -n . Тогда значение отношения W1$\in$ W2 совпадает со значением булевской переменной B, получающимся в результате работы следующей программы на предлагаемом языке:

```
B : = false;
for i : = 0 step 1 until n-m do
    begin B : = B∨W1 = pt(i+1,i+m,W2);
          if B then  go to  значение отношения найдено.
    end
```

В последующих примерах мы будем исходить из наличия стандартных описаний ранга (см.разд.5.5.2.).

отношение	значение отношения
‹abcde› =‹abcdf›	false

отношение	значение отношения
⟨abcde⟩ ≤ ⟨abcdf⟩	true
⟨abcd⟩ = ⟨abc{x}y⟩	false
⟨abc⟩ ∈ ⟨ambabcpq⟩	true
⟨ab*cd⟩ < ⟨ab*sz⟩	true
⟨am*ef⟩ < ⟨ak*ab⟩	false
⟨stl⟩ > ⟨tls⟩	false
⟨⟩ ∈ ⟨**a⟩	true
⟨⟩ ∈ ⟨ab⟩	false

3.6. БУКВЕННЫЕ ВЫРАЖЕНИЯ

3.6.1. Синтаксис

<первичное буквенное выражение> ::= ⟨<слово>⟩ | <переменная> | <функция> | (⟨буквенное выражение>)

<буквенный терм> ::= <первичное буквенное выражение> | <первичное буквенное выражение> & <буквенный терм>

<простое буквенное выражение> ::= <буквенный терм> | <простое буквенное выражение> ↓ <буквенный терм>

<буквенное выражение> ::= <простое буквенное выражение> | <условие> <простое буквенное выражение> else <буквенное выражение>

3.6.2. Примеры

Первичные буквенные выражения

```
N [1,k]
⟨basic⟩
subs(A1 & A2,1,1,X)
(if x=0 then A & B else A & C)
⟨25⟩
⟨⟩
```

Буквенные термы

```
⟨128⟩
⟨**⟩ & Q & ⟨**⟩ & L & ⟨**⟩
(R1 ↓ R2 & G & (if B then ⟨x⟩ else ⟨y⟩ )) & A
B[1] & ⟨{a52}⟩ & pt(i,j,C)
```

Простые буквенные выражения

```
A1 & A2 & A3 & A4 ↓ ⟨number⟩
(if A[y+2,x]>⟨a⟩ ↓ T then ⟨2⟩ else ⟨3⟩ ) ↓ T1
```

```
LK ‹‹128«result›
if ‹ › ∈ subs(N,k,l,M) then M1 else M2
if if x=z then1 < y else m>y then (if pt(i,i,ST1)= ‹ ›
        then ‹ab› else ‹ac› ) else RT1
```

3.6.3. С е м а н т и к а

Буквенные выражения являются правилом для нахождения буквенного значения - слова. В случае простого буквенного выражения это значение получается путем последовательного выполнения операций соединения (см.3.6.4.). Фактическим значением конструкции ‹‹слово›› является последовательность основных символов, находящаяся внутри буквенных скобок ‹ › . При этом длина простого слова считается равной числу основных символов, находящихся внутри буквенных скобок. (Если основные символы отсутствуют, то значение данной конструкции есть простое слово, состоящее из одного пустого элемента). Число элементов в составном слове считается равным числу звездочек, увеличенному на единицу, максимальная длина элемента - равной наибольшему числу основных символов в элементах данного слова.(Если основные символы в элементах отсутствуют, то максимальная длина элемента считается равной I).

В остальном семантика этого раздела совпадает с семантикой разд.3.3.I., если заменить везде слово "число" на "слово", а слово "арифметический" - на "буквенный"

3.6.4. О п е р а ц и и и т и п ы

Компоненты простого буквенного выражения должны иметь тип word .

Операция & носит название операции объединения. Данная операция соединяет значения двух операндов в одно слово по следующим правилам:

- Если значения обоих операндов относятся к классу простых слов, то результат - простое слово, полученное путем приписывания к значению левого операнда значения правого операнда справа. Число элементов получившегося значения равно сумме количеств элементов в значениях операндов.

- Если значения обоих операндов относятся к классу составных слов, то слово с меньшим числом элементов дополняется до размеров второго слова пустыми элементами путем фиктивного приписывания справа нужного количества звездочек. Затем все элементы первого слова объединяются попарно по предыдущему правилу с соответствующими элементами второго слова, начиная слева. Максимальная длина элемента полученного значения равна сумме максимальных длин элементов в значениях операндов.

- Если значения операндов относятся к разным классам значений, то значение, принадлежащее к классу простых слов, должно фиктивно восприниматься как двухэлементное составное

слово со вторым пустым элементом. Для этого в данном случае следует считать, что сначала к этому значению справа приписывается звездочка; максимальной длиной элемента в полученном двухэлементном составном слове следует считать число элементов соответствующего простого слова, а затем происходит выполнение операции по предыдущему правилу.

Операция ↓ носит название операции присоединения. Данная операция соединяет значения двух операндов в одно составное слово по следующему правилу: выписывается значение первого операнда, к нему справа приписывается звездочка и значение второго операнда. Число элементов в получившемся таким образом значении находится очевидными действиями, максимальная длина элемента равна наибольшей из длин элементов получившегося составного слова.

3.6.5. С т а р ш и н с т в о о п е р а ц и й

Последовательность операций в выражении, вообще говоря, выполняется слева направо с учетом того, что порядок старшинства операций по отношению к синтаксису раздела 3.6.I. таков:

первый: ↓

второй: ↓

Значение выражения между левой скобкой и соответствующей правой скобкой находится самостоятельно.

Ш.УI. Следующие примеры поясняют, каким образом происходит нахождение значения буквенных выражений.

Значение операнда A	Длина знач. операнда A	Значение операнда B	Длина знач. операнда B	Значение выражения	Длина знач. выражения
		Буквенное выражение: A↓B			
abcd	4	xyzw	4	abcdxyzw	8
ab	4	cde	3	abcde	7
ab*c	2:2	xyzw	4	abxyzw*c	2:6
a*b*c	3:2	l*k*m	3:1	al*bk*cm	3:3
a*b*c	3:1	l*k*m*n	4:1	al*bk*cm*n	4:2
ab**cd	3:2	****	5:2	ab**cd**	5:4
пустое слово	4	am	2	am	6
{x1}{x2}	2	128	3	{x1}{x2}128	5
m+n*axb	2:3	-k*+c	2:2	m+n-k*axb+c	2:5
lkm	3	ax*by*cz	3:2	lkmax*by*cz	3:5

Значение операнда A	Длина знач. операнда A	Значение операнда B	Длина знач. операнда B	Значение выражения	Длина знач. выражения
		Буквенное выражение: A ↓ B			
ab	2	cd	2	ab*cd	2:2
ab*cd	2:2	xyz	3	ab*cd*xyz	3:3
a*b*c	3:1	lm*kn	2:3	a*b*c*lm*kn	5:3
ayz	6	an*bkcz	2:4	ayz*an*bkcz	3:6
{a}*{b}n	2:5	ab	2	{a}*{b}n*ab	3:5

Поскольку пустые элементы в простых словах всегда стоят в конце слова (см.разд. 2.9.3.), то значение выражения $\prec \succ \ell$ A равно значению выражения A $\ell \prec \succ$ в тех олучаях, когда значение A относится к классу простых слов.

4.I. СОСТАВНЫЕ ОПЕРАТОРЫ И БЛОКИ

<оператор> :: = <оператор сдвига> | <оператор засылки>

4.2. ОПЕРАТОРЫ ПРИСВАИВАНИЯ

4.2.I. С и н т а к с и с

<оператор присваивания> :: = <список левой части> < буквенное выражение>

4.2.2. П р и м е р ы

N[1] : = AℓB↓C

M : = A[i,j,k+2] : = pt(l,m,A1)

4.2.3.3. Если оператор присваивания включает в себя буквенное выражение, то значение выражения присваивается всем переменным списка левой чаоти с учетом ограничений на количество элементов в их значениях, указанных в описаниях (см.разд.5.I.).

Выполнение операторов присваивания происходит по следующим правилам (рассматривается случай, когда в списке левой части находится лишь одна переменная):

- Если значения переменной левой части и выражения одновременно должны относиться к классоу простых слов, то предварительно путем добавления пустых элементов либо отбрасывания последних элементов в значении выражения происходит уравнивание числа элементов в значениях выражения и переменной. Затем полученное значение выражения присваивается переменной.

- Если значения переменной левой части и выражения должны относиться к классу состав-

ных слов, то сначала происходит аналогичное предыдущему уравнивание числа элементов в значении выражения с числом элементов в значении переменной. После этого также аналогично происходит уравнивание числа элементов в каждом из элементов значения выражения (простом слове) с максимальной длиной элемента в значении переменной.

Полученное таким образом значение выражения присваивается переменной левой части.

- Если значение переменной левой части должно относиться к классу простых слов, а значение выражения относится к классу составных слов, то в значении выражения отбрасываются все элементы, кроме первого, и получившееся таким образом значение рассматривается как простое слово с числом элементов, равным максимальной длине элемента в значении выражения.

- Если значение переменной левой части должно относиться к классу составных слов, а значение выражения относится к классу простых слов, то значение выражения воспринимается как двухэлементное составное слово, в котором второй элемент - пустое слово, а максимальная длина элемента равна числу элементов в данном простом слове.

В общем случае, когда в списке левой части находится несколько переменных, значение буквенного выражения присваивается параллельно всем переменным списка левой части по вышеописанным правилам.

Если в списке левой части имеется идентификатор процедуры либо переменная с неуказанной в описании длиной, то им присваивается точное значение буквенного выражения.

4.2.4. Т и п ы

Если переменные и идентификаторы процедур списка левой части имеют тип word , выражение должно быть буквенным.

IУ.П. Прежде всего приведем примеры.

Оператор присваивания A : = W

Значение выражения	Длина значения выражения	Длина значения переменной	Значение переменной
abcd	4	4	abcd
abcde	5	4	abcd
ab	2	4	ab
ab*cd*ef	3:2	4	ab
xpyqr*lm	2:5	4	xpyq
{a1}1218s	6	4	{a1}121
пустое слово	любая	4	пустое слово
ab*cd	2:2	4:2	ab*cd
abc*efg*hij*klm	4:3	4:2	ab*ef*hi*kl
a*b*c*d*e	5:1	4:2	a*b*c*d
a*abc*lk	4:3	4:2	a*ab*lk*
пустое слово	любая	4:2	* * *

Изложим, каким образом, по нашему мнению, может происходить трансляция введенных операций на трехадресной машине. Предположим, что каждый элемент простого олова будет находиться в отдельной ячейке памяти. Тогда каждое слово, вообще говоря, будет занимать некоторый массив ячеек. После того, как произойдет трансляторное распределение памяти, станет известно расположение этого массива в памяти машины.

Для большего удобства оперирования со словами в машине желательно завести для каждого слова еще одну ячейку, которую мы будем называть определяющей ячейкой.

Содержимое этой ячейки следующее:

КОП	IA	ПА	ША
признаки	n	α	m

α - указание о расположении массива ячеек, соответствующего данному слову, в памяти машины;

n - число элементов в слове;

m - максимальная длина элемента (положим для простых слов m = 0). КОП отводится под признаки. Признаки могут быть такие:

- определяющая ячейка соответствует простой переменной;
- определяющая ячейка соответствует переменной с индексами;
- в определяющей ячейке находится само значение слова (например, в случае одноэлементного проотого слова);
- определяющая ячейка соответствует "рабочему" значению олова (аналогу обычных рабочих ячеек); и т.п.

Будем предполагать, что в памяти машины существует рабочее поле для значений буквенных выражений. При нахождении значения буквенного выражения во время выполнения какой-либо операции предварительно выбираются не сами значения операндов, а их определяющие ячейки, по ним размечается рабочее поле для выполнения данной операции, значения операндов переносятся на отведенные места на рабочем поле и затем выполняется операция. Для операторов присваивания после нахождения значения буквенного выражения происходит выборка определяющей ячейки, соответствующей данной переменной списка левой части, и по указанной в ней информации выполняется присваивание переменной значения выражения по правилам разд. 4.2.2.3. Если в списке левой части находится идентификатор процедуры либо переменная с неуказанной в описании длиной, то соответствующая определяющая ячейка заполняется после нахождения значения буквенного выражения.

4.8. ОПЕРАТОР СДВИГА

4.8.1. Синтаксис

<простое первичное буквенное выражение> ::= ↓<простое слово>↑|<переменная>|
<функция>|(<правильное простое буквенное выражение>)

<простой буквенный множитель> ::= <простое первичное буквенное выражение>|
<простой буквенный множитель>↑<простое первичное буквенное выражение>

<правильное простое буквенное выражение> ::= <простой буквенный множитель>↑<условие>
<простой буквенный множитель> else <правильное простое буквенное выражение>

<составное первичное буквенное выражение> ::= ↓<слово>↑|<переменная>|<функция>|
(<правильное составное буквенное выражение>)

<составной буквенный множитель> ::= <составное первичное буквенное выражение>|
<составное первичное буквенное выражение>↓<составной буквенный множитель>

<правильное составное буквенное выражение> ::= <составной буквенный множитель>|
<условие><составной буквенный множитель> else <правильное составное буквенное выражение>

<правильное буквенное выражение> ::= <правильное простое буквенное выражение>|
<правильное составное буквенное выражение>

<направление> ::= left | right

<количество сдвигаемых элементов> ::= <арифметическое выражение>

<оператор сдвига> ::= shift <правильное буквенное выражение> <направление>
<количество сдвигаемых элементов> | <метка>: <оператор сдвига>

4.8.2. Примеры

```
shift M left 4
shift A↓B↓C right n+1
shift I[1] ↓ N left 8
shift if a < b then P↓N else Q right s-n
shift T↓R↓T right i
shift N ↓ (if x ≤ y then Q else Q1) left if r = ↓ ↑ then 2 else k1
```

4.8.3. Семантика

Оператор сдвига присваивает новые значения переменным, указанным в правильном буквенном выражении. Значение простого первичного буквенного выражения должно относиться к классу простых слов. Если значение арифметического выражения, указывающего количество

сдвигаемых элементов, есть отрицательное число, то направление сдвига меняется одновременно со знаком арифметического выражения. В том случае, когда данное арифметическое выражение не есть выражение типа integer , автоматически подключается преобразующая функция entier(E+0.5) , где E - данное арифметическое выражение.

Оператор сдвига выполняется следующим образом. Определяется, какие первичные компоненты (переменные, слова, функции) участвуют в построении правильного буквенного выражения при данном наборе значений булевских выражений, входящих в условия рассматриваемого буквенного выражения. Переменным с неопределенным значением присваиваются в качестве значений пустые слова. Затем производится сдвиг значений в компонентах путем выталкивания крайних элементов из значения каждой компоненты в значение компоненты, следующей за ней в направлении сдвига. Для переменных в полученных значениях происходит упорядочение элементов в соответствии с синтаксисом разд.2.9.

Позиции элементов в значениях компонент, расположенных в конце правильного буквенного выражения, противоположном направлению сдвига, после выполнения сдвига заполняются пустыми элементами, если только какая-либо компонента не встречается в правильном буквенном выражении дважды. Результат действия оператора сдвига не определен, если одна и та же компонента встречается дважды с той стороны от операции соединения, которая совпадает с направлением сдвига.

В этом случае, когда в правильном буквенном выражении встречается переменная с неуказанной в описании длиной, то перед выполнением операции сдвига ей присваивается в качестве значения одноэлементное пустое слово, если до выполнения оператора сдвига неизвестно значение данной переменной.

При выполнении оператора сдвига, включающего в себя составное правильное буквенное выражение, сдвиг элементов происходит с учетом максимальных длин элементов в значениях компонент. Для простых слов здесь роль максимальной длины элемента играет число элементов.

IУ.УШ. Примеры

Переменная	Значение переменной	Длина знач. перем.	Оператор сдвига	Длина зн. прав.букв. выражения	Перемен.	Значение перем. после выполнен. опер. сдвига
A	xyzw	4	shift A left 2	4	4	zw
A1	ab∗cd	2:2	shift A1 left 1	2:2	A1	cd∗
A2	abc	3	shift A2&B2&C2 left 4	10	A2	efg
B2	defg	4			B2	hij
C2	hij	3			C2	пустое слово
D	{a1}x	2	shift D right 1	2	D	{a1}

Переменная	Значение переменной	Длина знач. перем.	Оператор сдвига	Длина зн. прав.букв. выражения	Перемен.	Значение перемен. после выпол. опер. сдвига
E	yzw▫pq▫sn	3:3	shift E right 2	2	E	▫▫yzw
F	abcd	4	shift F⫷G⫷G left 4	6	F	zzzz
G	z	1			G	пустое слово
F1	klm	4	shift F1⫷<np> left 2	6	F1	mnp
G1	aaaa	4	shift Q1⫸H1 right 1	2:4	G1	пустое слово
H1	bb	2			H1	bb
G2	ab▫bc▫cd	3:2	shift G2⫸H2 left 1	5:3	G2	bc▫cd▫xy
H2	xyz▫uvw	2:3			H2	uvw
L	lk	3	shift <⫸L right 1	4	L	lk
L1	любое простое слово	n-любое	shift L1⫷<left 1	n+1	L1	пустое слово

4.9. ОПЕРАТОР ЗАСЫЛКИ

4.9.1. С и н т а к с и с

<засылаемая часть> ::= <буквенное выражение> | <арифметическое выражение>

<место засылки> ::= <правильное простое буквенное выражение>

<оператор засылки> ::= send <засылаемая часть> in <место засылки> | <метка>: <оператор засылки>

4.9.2. П р и м е р ы

```
send  A⫷B⫸C[i]   in pt(i,i,M)
send  25*sin(x)   in if x < 16 then A else <{z2}⫸
send  <xy▫s▫d▫c>  in opt(i,1,M⫷N⫷Q)
send  284         in pt(1,1,extr(1[s],P)
```

4.9.3. С е м а н т и к а

Оператор засылки служит для присваивания нового значения переменной, входящей в состав слова в качестве элемента типа адрес. Значение простого правильного буквенного выражения, указывающего место засылки, должно быть одноэлементным простым словом с элементом типа адрес. После выполнения оператора засылки переменной, помещающейся в данном элементе, будет присвоено значение выражения, указанного в засылаемой части. Типы данной переменной и выражения засылаемой части должны совпадать. Присваивание значения переменной происходит в смысле разд.4.2. (Операторы присваивания). Результат действия оператора засылки не определен, если значение простого буквенного выражения, указывающего место засылки, не есть элемент типа адрес.

IУ.IX. Оператор засылки необходим для работы с адресами любого ранга. Отличие его от оператора присваивания состоит в том, что здесь не указывается непосредственно адрес переменной, а указывается лишь место, откуда взять этот адрес.

5. О П И С А Н И Я

<описание> :: = <описание ранга>

5.I. ОПИСАНИЕ ТИПОВ

5.I.I. С и н т а к с и с

<число элементов в слове> :: = <арифметическое выражение>

<максимальная длина элемента слова> :: = <арифметическое выражение>

<длина слова> :: = <число элементов в слове> | <число элементов в слове> : <максимальная длина элемента слова>

<тип> :: = word

<новый список идентификаторов> :: = <список идентификаторов> (<длина слова>)

<новый список типа> :: = <новый список идентификаторов> | <новый список идентификаторов>, <новый список типа>

<описание типа> :: = word <новый список типа> | own word <новый список типа>

5.I.2. П р и м е р ы

```
word A(6)
word A,B,C(3:2)
own word E1,F(4),G(2:5)
word P,N,K
```

5.I.3. С е м а н т и к а

Значения арифметических выражений, определяющих длину слова, вычисляются так же, как и индексные выражения.

Описания типа служат для указания того, что значения некоторых переменных могут быть отнесены к значениям определенного класса.

Переменные типа word могут принимать значения либо из класса простых слов, либо из класса составных слов.

Если значения некоторых переменных должны принадлежать к классу простых слов, то для них в конце нового списка идентификаторов в круглых скобках должно быть указано число элементов, допускаемое в данных значениях. Все переменные, указанные в данном списке, будут иметь значения, состоящие из одинакового числа элементов.

Если значения некоторых переменных должны принадлежать к классу составных слов, то для них в конце нового списка идентификаторов в круглых скобках указывается как число элементов в их значениях, так и максимальная длина элемента значения. Все переменные данного списка будут иметь значения, состоящие из одинакового числа элементов, обладающих одинаковой максимальной длиной.

Если в описании типа не указана длина значения буквенной переменной, то принадлежность ее к определенному классу значений определяется после каждого присваивания значения данной переменной.

5.I.3.I. Переменная типа word не определена, если значения арифметичеоких выражений, определяющих число элементов в простом слове и максимальную длину элемента слова, являются числами, меньшими единицы, а значения арифметичеоких выражений, определяющих число элементов в составных словах, - числами, меньшими двух.

5.2. ОПИСАНИЕ МАССИВОВ

5.2.I. С и н т а к с и с

<новый сегмент массива> ::= <новый список идентификаторов> [<список граничных пар>] | <новый список идентификаторов>, <новый сегмент массива>

<список массивов> ::= <новый сегмент массива> | <список массивов>, <новый сегмент массива>

5.2.2. П р и м е р ы

```
word array T(6)[1:15]
own word array Q(2),L(14:25)[1:4],P,S,O(3),M(2:5)[m:n]
word array A,B(1)[1:12,1:20],C(1)[1:60],D(4:2)[1:q]
word array L,P(2:4)[1:7],N,N1[6:11]
```

5.5. ОПИСАНИЕ РАНГА

5.5.I. С и н т а к с и с

<операция ранга> ::= <

<сегмент ранга> ::= <<элемент слова>> <операция ранга> <<элемент слова>> |
<сегмент ранга> <операция ранга> <<элемент слова>>

<список ранга> ::= <сегмент ранга> | <список ранга>, <сегмент ранга>

<описание ранга> ::= rank <список ранга>

5.5.2. Примеры

rank ‹ g› ‹ a› , ‹ 0› ‹ 1›

rank ‹ ,› ‹ .› ‹ :› , ‹ a1› ‹ b1›

rank ‹ abd› ‹ bca› ‹ ceg› , ‹ G2› ‹ A2›

5.5.3. Семантика

Описания ранга служат для определения отношений между элементами слов и используются при нахождении значений булевских выражений (см.разд.3.4.).

Если описание ранга противоречиво внутри себя, то результат соответствующего отношения не определен.

Для двух взаимно противоречивых описаний ранга действительным считается последнее из них.

5.5.3.1. Отыскание сравниваемых элементов слов в описаниях ранга. Пара элементов считается входящей в описание ранга, если оба элемента находятся в одном из сегментов данного описания. Если в описаниях ранга отыскиваются элементы составного слова, то в каждом из таких элементов предварительно фиктивно отбрасываются последние пустые элементы.

5.5.3.2. Стандартное описание ранга

Рекомендуется иметь стандартное описание ранга, которое всегда будет считаться включенным в описание самого внешнего блока программы. Если данный блок имеет свои описания ранга, то они будут восприниматься как стоящие после стандартного описания ранга.

5.5.3.2.1. Стандартный список ранга для простых слов

В основу всех сегментов данного списка положен лексикографический принцип (‹ › ‹ *› ‹ a› ‹ b› и т.д. , ‹ › ‹ *› ‹ A› ‹ B› и т.д.)

Следующий простой сегмент ранга содержится в данном списке:

‹ › ‹ *› ‹ 0› ‹ 1› ‹ 2› ‹ 3› ‹ 4› ‹ 5› ‹ 6› ‹ 7› ‹ 8› ‹ 9›

5.5.3.2.2. Стандартный список ранга для составных слов

Правила построения всех сегментов данного списка следующие:

- В качестве элементов составных слов любого данного сегмента выписываются все простые слова таким образом, чтобы значение отношения с операцией $<$, в котором в качестве первого операнда взято любое простое слово из данного сегмента, а в качестве второго операнда - любое последующее простое слово того же сегмента всегда равнялось логическому значению true . При нахождении значения отношения считается действующим описание ранга со стандартным списком ранга простых слов. Если в данном отношении значения операндов име-

ют разное число элементов, то значение с меньшим числом элементов при вычислении отношения дополняется пустыми элементами.

– Каждый сегмент данного стандартного списка строится для элементов составного слова с максимальной длиной элемента, постоянной для данного сегмента и определяющейся порядковым номером этого сегмента в стандартном списке ранга составных слов.

§ 2. Примеры описаний процедур на предлагаемом языке

Пример 1

```
procedure ИНВЕРСИЯ (S,m,n); value m,n; word S; integer m,n;
    comment Эта процедура меняет на обратный порядок элементов в составном слове S с
     количеством элементов n и максимальной длиной элемента m ;
  begin integer i; word T(m),V(n:m);
        for i:=1 step 1 until n do
            begin shift T ¦ S left 1;
                  shift T ¦ V right 1
            end ;
        S := V
  end   ИНВЕРСИЯ
```

Например, пусть значение переменной P с количеством элементов 3 и максимальной длиной элемента 5 равно: это « есть « слово. После выполнения оператора процедуры ИНВЕРСИЯ (P,5,3) значение P станет равным : слово « есть « это.

Пример 2

```
procedure     ПОДРАЖАНИЕ  (P,K,a,i) ; value i; word P,K,a;
    comment   Эта процедура описывает алгоритм подражания работе машины Тьюринга, опи-
     санный в [6] . K – начальная конфигурация, P – функциональная схема, заданная по-
     следовательностью пятерок символов (см. ниже). a – элемент внутреннего алфавита,
     определяющий исходное состояние, i – номер ячейки машины Тьюринга, под которой он
     подписывается в начальной конфигурации.
```

Примечание. Алфавит языка считается дополненным символами внешнего и внутреннего алфавита машины Тьюринга.

Пример. Содержание фактических параметров при обращении к процедуре для алгоритма сложения, описанного в [6] , должно быть следующим:

первый: $|q_0 \wedge q_2 \sqcap | q_2 | q_1 \wedge | q_1 | q_2 \sqcap \wedge q_1 \wedge q_0 \sqcap \wedge q_1 \wedge q_0 \sqcap \wedge q_2 | q_1 H \oplus q_0 \wedge | H \oplus q_1 \oplus q_1 \wedge \oplus q_2 \oplus q_2 \sqcap$

второй: ||||||⊛||||

третий: %.

четвертый: I

Здесь элемент ⊛ представляет обычную звездочку работы [6] ;

```
begin  word B(1),D(2) ; integer j ;
       D:= pt(i,i,K) & a;
   L1: j:= -5;
   L2: j:= j+5;
       if pt(j+1,j+2,f) ≠ D then go to L2 ;
       K:= subs(pt(j+3,j+3,F),i,i,K) ;
       B:= pt(j+5,j+5,F) ;
       if B = ⟨!⟩ then go to M else
         i:= if B= ⟨П⟩ then i+1 else if B= ⟨Л⟩ then i-1 else i ;
       D:= pt(j+3,j+4,F) ;
       go to L1 ;
M; end      ПОДРАЖАНИЕ
```

По нашему мнению, программа алгоритма подражания машине Тьюринга может служить качественной характеристикой алгоритмического языка. Сравнение программ подражания, написанных на разных языках, может оказаться удобным для определения сравнительной эффективности данного языка (при этом меру эффективности можно связывать с длиной настоящей программы).

Пример 3

```
procedure  ПОЛЬСКАЯ (M,m,N,n) ; value m;integer m,n; word M,N;
   comment Эта процедура преобразует в польскую запись формулу M. Считается, что формула
     составлена верно и содержит лишь бинарные операции. m - количество элементов в фор-
     муле M , N , n соответственно - результирующая формула и количество элементов в ней;
   begin word A(1),B(5),D((m-1)*4),E((m+1)*2) ; integer i,j ;
       j:= 0 ;
       B:= ⟨+-x/*⟩ ;
       shift D & ⟨**⟩ & ⟨**⟩ left(m-1)*4 ;
       E:= ⟨⟩ ;
       for i:=1 step 1 until m do
          begin A:= pt(i,i,M) ;
                if A = ⟨(⟩ then j:= j+1 else
```

```
                if A ∈ B then D:= subs(A,j,j,D) else
                    if A = ‹ › then begin E:= E ∉ pt(j,j,D); j:=j-1 end
                                else E:= E ∉ A
            end просмотра;
        N:= E ;
        n:= (m+1)÷2
    end  ПОЛЬСКАЯ
```

Следующие операторы переводят формулу A в формулу B с числом элементов n ;

		Значение A	Значение B	n
ПОЛЬСКАЯ	(A,13,B,n)	(((a+b)-c)xd)	ab+c-dx	7
ПОЛЬСКАЯ	(A,17,B,n)	(((a+b)-(cxd))÷e)	ab+cdx-e÷	9
ПОЛЬСКАЯ	(A,25,B,n)	(((a+(b-c)x((d÷e)+f))/g	abc-+de÷f+xg/	13

Пример 4

```
procedure  ГАММА МЮ ШПУР  (A,n,T) ; value n ; integer n ; word A,T ;
  comment  Эта процедура находит формулу для вычисления шпуров вида tr b₁b₂... bₙ для
    случаев, когда сомножители  b₁b₂...bₙ  имеют вид  ∗^â_μ  (см. [7] ). n - число со-
    множителей в произведении (предполагается, что каждый сомножитель представлен одной
    буквой), A - последовательность букв в произведении, T - результат. Каждое слагае-
    мое формулы находится при помощи вспомогательной последовательности чисел  α_i
    (см. [7] , генерация перестановок);
  begin integer i,j,k,l,m; integer array a [1:n] ; Boolean S ;
        T:= ‹ › ;  S:= true ;
        for i:=1 step 1 until n do a [i] := i ; comment начальная перестановка;
        go to L3 ;
    L1: i:= n ;
    L2: i:= i-2 ; if i=0 then go to N ;
        l:= n+1 ;
        for k:= i+1 step 1 until n do if a [k]>a [i] ∧ a [k]< l then
                                        begin l:= a [k]  ;  j:= k end ;
        if l=n+1 then go to L2 ;
        a [j] := a [i] ;  a [i] := l ; comment В последовательности a₁a₂... aₙ
      число  a_i  поменялось местами с наибольшим из последующих чисел;
        for k:=i+1 step 1 until n-1 do
          for m:=k+1 step 1 until n do  if a [k]> a [m] then
```

```
                                    begin  l:= a [m] ;
                                           a [m] := a [k] ;
                                           a [k] := l  ;
                                    end ;
      comment  Все числа, следующие за a , расположились в возрастающем порядке;
      S:= ¬ S ;
L3:   T:= T ℓ(if S= true then ⟨ +(⟩  else  ⟨ -(⟩  ;
      j:= 2 ;
      for i:=1 step 1 until n do  begin  T:= T ℓ pt(a [i] , a [j] ,A) ;
                                         if i=j then begin T:= T ℓ⟨)(⟩ ;
                                                           j:= j+2
                                                     end
                                    end i ;
      T:= T ℓ ⟨ )⟩ ;
      go to L1 ;
N:  end                ГАММА МЮ ШПУР
```

Пример. Если необходимо найти формулу для вычисления tr(abcdef) , то после обращения к данной процедуре с оператором ГАММА МЮ ШПУР(⟨abcdef⟩,6,F) получим в F формулу

(ab)(cd)(ef)-(ab)(ce)(df)+(ab)(cf)(de)-(ac)(bd)(ef)+(ac)(be)(df)-
(ac)(bf)(de)+(ad)(bc)(ef)-(ad)(be)(cf)+(ad)(bf)(ce)-(ae)(bc)(df)+
(ae)(bd)(cf)-(ae)(bf)(cd)+(af)(bc)(de)-(af)(bd)(ce)+(af)(be)(cd)

Пример 5

```
procedure ПРИСВОЕНИЕ (M,t) ; value M,t ; word M ; integer t ;
  comment  Эта процедура описывает алгоритм присвоения ладовых функций нотам мелодии M
    (см. [8] ).  t - число тактов мелодии (число элементов в составном слове M). Любой
    элемент слова M есть простое слово, состоящее из элементов типа адрес. Каждый элемент
    типа адрес представляет одну ноту мелодии. Значение переменной, находящейся в этом
    элементе, есть составное слово, в котором первый элемент есть интонационная высота но-
    ты в буквенной записи (если пауза, то пустое слово), второй элемент - длительность но-
    ты, умноженная на 16, третий элемент - ладовая функция этой ноты;
  begin integer i,j,i1,j1,l,j3; word A(1),P,Q,(3:2),T,D,S(6:2); Boolean B;
        comment  предполагается наличие глобальной функции random(a) , значение которой
          есть случайное число, и соответствующей ей переменной a ;
        word procedure N(mju,nju,kappa,R); value mju,nju,kappa,R;
```

```
integer nju,nju,kappa; word R;
  begin  N:= pt(nju,nju,extr(nju,pt(kappa,kappa,R))) end ;
Boolean procedure K; comment эта процедура-функция имеет значение true в
  тех случаях, когда мы уже рассмотрели последнюю ноту данного такта (или
  данный такт вообще не надо рассматривать);
  begin integer r,r1,r2; Boolean C1,C2;
    C1:= opt(j,i,M) = < > V (t=9 ∧ i=1) ;
    C2:= false ;
    if ¬ (t=9 ∧ i=5) V t=8 then go to LK ;
    r1:= 0 ;
первые паузы: for r:= 1,r+1 while N(1,r,1,M)= < > do
                    r1:= r1+valoir(N(2,r,1,M));
    r2:= 0 ;
  затакт: for r:=1 step 1 until j-1 do r2:= r2+valoir(N(2,r,1,M)) ;
    C2:= (r1 = r2);
LK: K:= C1 V C2
  end K ;
procedure ВЫБОР (P1,Q1) ; word P1,Q1 ;
  comment Эта процедура производит выбор двух соседних звучащих нот в мелодии;
  begin integer S ; switch q := LB1,LB2 ;
    S:= 1 ;
LB: if K then begin j:= 1; go to q [S] end ;
пауза: if N(1,j,i,M) = < > then begin j:= j+1; go to LB end ;
LB1: P1:= extr(j,pt(i,i,M)); S:=2; i1:= i; j1:= j; j:= j+1;
    go to LB ;
LB2: Q1:= extr(j,pt(i,i,M))
  end ВЫБОР ;
procedure НОВ (i2,j2,C3) ; integer i2,j2; Boolean C3;
  comment Эта процедура проверяет возможность замены ладовой функции на но-
    вую. Если замена возможна, процедура производит ее. C3 имеет значение
    true в случае, когда замена невозможна;
  begin word A1,A2(1),A3(3); integer l1;
    C3:= true ;
    A3:= < {T} { D } {S} > ;
    A1:= N(3,j2,i2,M) ;
```

```
                for i1:=1 step 1 until 3 do
                  begin shift A2 ⟨ A3 left 1 ;
                        if A1=A2 then go to LN ;
                        if N(1,j2,i2,M) ∈ extr(1,A2) then
замена:                 begin send subs(A2,3,3,extr(j2,pt(i2,i2,M)))
                              in opt(j2,i2,M) ;  C3:= false
                        end;
                LN:   end цикла
                end НОВ;
               procedure ПРОБА  (B2,n) ; Boolean B2; integer n;
                  comment Эта процедура производит проверку заданного правила B2 для всей
                    мелодии с заменой ладовых функций при невыполнении правила. n - номер
                    ноты, с которой надо начинать проверку;
                  begin integer m; Boolean B1 ;
                    i:= 1; j:= n;
                  LT: j:= j+1 ; ВЫБОР  (P,Q) ;
                    if B2 then begin НОВ (i1,j1,B) ; if B then НОВ (i,j,B) end;
                    if i≠T then go to LT ;
последний такт:  B1 := true ;
                    for m:= j+1 , m+1 while opt(m,i,M) ≠ ⟨ ⟩ do
                          B1:= B1  N(1,m,i,M) = ⟨ ⟩ ;
                    if  (B1 ∨ opt(j+1,i,M)= ⟨ ⟩ ) ∧ N(3,j,i,M)≠ ⟨{T}⟩  then
                   НОВ  (i,j,B) else if ¬ (B1 ∨ opt(j+1,i,M)= ⟨ ⟩) then go to LT
                  end ПРОБА;
   начало ПРИСВОЕНИЯ : T:= ⟨ g•c1•e1•g1•c2•e2 ⟩ ;
                       D:= ⟨g•b•d1•g1•b1•d2⟩ ;
                       S:= ⟨a•c1•f1•a1•c2•f2⟩ ;
                       for i:=1 step 1 until t do
                         begin  j:= 0 ;
                           LP:  j:= j+1 ;
                                if K then go to LP2 ;
                         пауза :  if N(1,j,i,M)= ⟨ ⟩ then go to LP ;
                              LP1: a:= random(a) ;
                                   A:= if a < .3333 then ⟨{T}⟩ else
                                        if a < .6666 then ⟨{D}⟩ else  ⟨{S}⟩ ;
```

```
                    if ¬N(1,j,1,M) ∈ extr(1,A) then go to LP1 ;
                    send subs(A,3,3,extr(j,pt(1,1,M))) in opt(j,1,M) ;
                    go to LP ;
LP2:  end  присваивания нотам такта ладовых функций;
Проверка правил:  if t=8∧N(3,1,1,M)≠ <{T}> then НОВ (1,1,B) ;
                  j3:= if t=8∧N(3,1,1,M)= <{T}> then 1 else 0 ;
                  ПРОБА (pt(1,1,P)=pt(1,1,Q)≠pt(3,3,P)=pt(3,3,Q),j3) ;
                  for i:=1 step 1 until 2 do
                    begin ПРОБА ((i1=i-1)∧ pt(3,3,P)=pt(3,3,Q),j3) ;
                          ПРОБА (pt(3,3,P)= <{D}>∧pt(3,3,Q)= <{S}> ,j3)
                    end
end  ПРИСВОЕНИЕ
```

ЗАКЛЮЧЕНИЕ

При трансляции с предлагаемого языка можно воспользоваться уже существующими трансляторами с языка АЛГОЛ-60. Как упоминалось во введении, запись программы на предлагаемом языке можно формально преобразовать так, что она станет записью программы на языке АЛГОЛ-60. Можно создать блок транслятора, который, выполняя эти преобразования, будет работать перед началом работы транслятора с языка АЛГОЛ-60.

Будем считать, что все символы, введенные в разделе 2.3. (§ I), включены в алфавит языка АЛГОЛ-60 в качестве букв. Тогда сущность алгоритма преобразования программы в общих чертах заключается в следующем.

- Заменить каждый минимальный оператор, включающий в себя величины типа word , на соответствующий оператор процедуры, фактическими параметрами которого являются преобразованные выражения, находящиеся между разделителями данного преобразуемого оператора.

- В буквенных выражениях: заменить каждое слово, взятое в буквенные скобки, на процедуру-функцию, фактическим параметром которой служит строка, заключающая в себе данное слово; каждые два операнда с соответствующей им операцией соединения заменить на соответствующую процедуру-функцию, фактическими параметрами которой служат данные операнды.

- В булевских выражениях, использующих отношения между буквенными выражениями: заменить эти отношения на процедуры-функции, фактическими параметрами которых служат данные буквенные выражения.

-В описаниях: для величин типа word заменить всюду символ word на символ integer; запомнить для переменных и массивов типа word длины значений, убрав выражения для них из описаний в специальные операторы процедуры.

Сущность изменений в трансляторе с языка АЛГОЛ-60 вытекает из вышеописанного алго-

ритма:

– Необходимо, чтобы была обеспечена возможность подключения к программе ряда специальных процедур, тела которых есть коды (эти процедуры могут быть оформлены как стандартные программы).

– При обработке рекурсивных процедур иногда нужно запоминать некоторые промежуточные значения (см. [9] , [10]); необходимо учитывать, что некоторые из этих значений (адреса которых являются адресами определяющих ячеек) могут потребовать запоминания массива ячеек.

В заключение автор выражает благодарность Л.С.Нефедьевой и В.П.Ширикову, которые читали эту работу и сделали ряд ценных замечаний.

ЦИТИРОВАННАЯ ЛИТЕРАТУРА

1. Дж.В.Бэкус и др. Сообщение об алгоритмическом языке АЛГОЛ-60. Ж.выч.матем. и мат.физ., 1961, I, № 2, 308-342.

2. Дополнение к "Сообщению об АЛГОЛ-60". Ж.вычисл.матем. и матем.физ., 1964, 4, № I, 195-198.

3. В.С.Королюк. О понятии адресного алгоритма. Сб. проблемы кибернетики, вып.4, Физматгиз, 1960.

4. I.H.Wegstein and W.W.Jouden. A String Language for Symbol Manipulation Based on ALGOL-60. Communs ACM, 1962, 5(1), 54-61.

5. Miriam G. Shoffner and Peter Y. Brown. A Suggested Method of Making Fuller Use of Strings in ALGOL-60. Communs ACM , 1963, 6(4), 169-171.

6. Б.А.Трахтенброт. Алгоритмы и машинное решение задач. М. Физматгиз, 1958.

7. H.I.Kaiser. Trace calculation on electronic computer. Nuclear Physics, 1963, 43, 620-627.

8. М.С.Рытвинская, В.И.Шаронов. Гармонизация песенных мелодий на быстродействующей ЭВМ. Сб. Вероятностные методы и кибернетика, II, Уч.зап. КГУ, т.123, кн.6, 103-111, Казань, 1963.

9. В.И.Собельман, М.Р.Шура-Бура. Реализация рекурсивных процедур в языке АЛГОЛ-60. Ж.вычисл.матем. и матем.физ., 1962, 2, № 2, 303-316.

10. М.Р.Шура-Бура, Э.З.Любимский. Транслятор АЛГОЛ-60. Ж.вычисл.матем. и матем.физ., 1964, 4, № I, 96-112.

Рукопись поступила в издательский отдел
6 мая 1964 года.