УДК:681.3.06:51

ЯЗЫК $\mathcal{JP}$ I
(полное описание)

Ю.Н. Корнев, С.В. Пискунов, С.Н. Сергеев

§1. Введение

Язык $\mathcal{JP}$ - I является языком программирования, предназначенным для преобразования слов в произвольном алфавите.

По носителю язык ориентирован на машины с однородной структурой и однородные сети конечных автоматов: итеративные сети, вычислительные среды и т.п.

Для языка характерна динамическая структура программы. Принадлежность слова программы к тому или иному синтаксическому образованию (а следовательно, и его интерпретация) определяется тем, какое подслово программы в данный момент времени обозревается исполнителем и зависит от состояний программы в предыдущие моменты времени.

Так же, как в языках COMIT [1], SNOBOL, (SNOBOL-3 [2]), в $\mathcal{JP}$-I основным средством преобразования слов является операция подстановки.

1.1. Пусть $\mathcal{V}=\{a_1, a_2, \ldots, a_n\}$ есть некоторый алфавит, который состоит из символов языка, $\mathcal{F}(\mathcal{V})$ - свободная полугруппа над $\mathcal{V}$; A, B, C, D - слова из $\mathcal{F}(\mathcal{V})$, $N_0=\{0, 1, 2, \ldots\}$.

Пусть слово A имеет вид $a_1 \dots a_k$, где $a_i \in \mathcal{U}$, $i = 1, \dots, n$, и $\ell : F(\mathcal{U}) \to N_0$ – такая функция, что $\ell(A) = k$. Тогда $\ell(A)$ называется **длиной слова** A.

Слово B называется **подсловом** A, если существуют такие слова C и D, возможно, пустые, что $A = CBD$. Это отношение будем записывать в виде $B \subset A$.

Пусть $A = CBD$, $k = \ell(C)$. Слово B, стоящее в A после слова C с длиной k, называется **вхождением** $(k, B \subset A)$.

Пусть множество всех вхождений слова B в слово A записано в виде множества $\{(k_1, B \subset A), \dots, (k_n, B \subset A)\}$, при этом $k_1 < k_2 < \dots < k_n$. В этом случае индекс символа k назовем **номером вхождения** слова B в слово A, вхождение $(k_1, B \subset A)$ назовем первым вхождением, $(k_2, B \subset A)$ – вторым и т.д.

Вхождение $(0, B \subset A)$ называется **началом слова** A.

Пусть $A = CB$, $\ell(C) = k$. Вхождение $(k, B \subset A)$ называется **концом слова** A.

Пусть начало программы, конец программы – символы из $\mathcal{U}$. **Программой** называется слово, началом которого служит слово начало программы, концом – конец программы. Других вхождений символов начало программы, конец программы в программу нет.

I.2. Правила чтения программы

I.2.I. Программа читается слева направо.

I.2.2. Если при чтении встречается оператор языка и если этот оператор входит в свою правильную структуру, то он выполняется.

Программа, которая в процессе выполнения каждого оператора является правильной структурой относительно выполняемого оператора, называется **правильной программой**.

Результатом выполнения оператора является либо изменение программы, либо указание символа, с которого следует продолжать чтение программы, либо то и другое вместе.

Если характер чтения оператором не меняется, то после его выполнения программа читается, либо, начиная с символа, стоящего за просмотренным оператором, либо, если просмотренный оператор был оператором чистки слов, расположенным внутри стираемой области, с символа, стоящего за правой границей стираемой области.

I.3. Синтаксические определения языка даются в бэкусовской форме.

I.4. Семантика операторов определяется только для операторов, находящихся в своих правильных структурах.

I.5. Некоторые определения.

<слово B, ограниченное непустым словом A> ::= <слово вида ABA, где слово B не содержит вхождений непустого слова A>

<пусто> ::=

§ 2. Символы языка

2.I. Синтаксис

<символ> ::= <символ алфавита> | <основной символ>

<основной символ> ::= <ограничитель> | <спецификатор>

<ограничитель> ::= заменить | на | ; | : | (|) | < | > | если | применима | то | не применима | есть | нет | определить вхождение | вхождение | читать | стереть | стоп | переписать | к | в | стереть слева | стереть справа | *

<спецификаторы> ::= символ | слово | разделитель | | | граница | Γ | значения переменных

<программные ограничители> ::= начало программы | конец программы

2.2. Семантика

Под символом понимается неделимая единица информации. Неделимость символа означает, что при построении языка и его развитии не используются части символа.

§ 3. Алфавит

Алфавитом называется произвольная конечная совокупность символов, именуемых в дальнейшем символами алфавита.

Например, элементами алфавита могут быть буквы русского алфавита, цифры, нотные знаки и т.п.

Пересечение множества основных символов с множеством символов алфавита должно быть пустым.

§ 4. Слово

4.I. Синтаксис

< слово > :: = < пусто > | < символ > | < слово > < символ >

§ 5. Операторы языка

5.I. Синтаксис

< оператор > :: = < оператор типа подстановки > | < условный оператор > | < оператор чтения > | < оператор переписи > | < оператор чистки > | < оператор конца >

5.2. Операторы типа подстановки

5.2.I. Синтаксис

< указатель границ > :: = < указатель постоянной границы > | < указатель переменной границы >

< указатель постоянной границы > :: = в < имя границы >

< указатель переменной границы > :: = в < переменное имя границы >

< имя границы > :: = < символ алфавита >

< переменное имя границы > :: = < переменный символ >

< переменный символ > :: = символ < символ алфавита >

< операционная часть оператора подстановки > :: = заменить < левая часть подстановки > на < правая часть подстановки >

< допустимый символ слова подстановки > :: = < ограничитель > | < символ алфавита > | разделитель | ⊥ | граница | Г | значения переменных | < переменный символ > | < переменное слово >

< слово подстановки > :: = < допустимый символ слова подстановки > | < слово подстановки > < допустимый символ слова подстановки >

< левая часть подстановки > :: = < пусто > | < слово подстановки, не содержащее символа на и не начинающееся с символов ⊥ или разделитель > | < слово подстановки, ограниченное разделителем >

< правая часть подстановки > :: = < пусто > | < слово подстановки, не содержащее символов ; , применима, не применима и не начинающееся с символов ⊥ или разделитель > | < слово подстановки, ограниченное разделителем >

< переменное слово > :: = слово < символ алфавита >

< разделитель > :: = разделитель < символ алфавита > | ⊥ < символ алфавита >

< операционная часть оператора определения вхождения > :: = определить вхождение < слово оператора вхождения > | вхождение < слово оператора вхождения >

< слово оператора вхождения > :: = < слово подстановки, не содержащее символов ; , есть, нет и не начинающееся с символов ⊥ или разделитель > | < непустое слово подстановки, ограниченное разделителем >

< операционная часть оператора подстановки без переменных > :: = < операционная часть оператора подстановки, не содержащая символов символ и слово >

< операционная часть оператора подстановки с переменными > :: = < операционная часть оператора подстановки, содержащая или символ символ или символ слово или то и другое вместе >

< операционная часть оператора определения вхождения без переменных > :: = < операционная часть оператора определения вхождения, не содержащая символов символ и слово >

< операционная часть оператора определения вхождения с переменными > :: = < операционная часть оператора определения вхождения, содержащая или символ символ, или символ слово, или то или другое вместе >

< оператор подстановки без переменных > :: = < указатель постоянной границы > < операционная часть оператора подстановки без переменных >

< оператор подстановки с переменными и постоянной границей > :: = < указатель постоянной границы > < операционная часть оператора подстановки с переменными >

<оператор подстановки с переменными>::= <указатель переменной границы> <операционная часть оператора подстановки>

<оператор определения вхождения без переменных>::= <указатель постоянной границы> <операционная часть оператора определения вхождения без переменных>

<оператор определения вхождения с переменными и постоянной границей>::= <указатель постоянной границы> <операционная часть оператора определения вхождения с переменными>

<оператор определения вхождения с переменными>::= <указатель переменной границы> <операционная часть оператора определения вхождения>

<операционная часть оператора типа подстановки без переменных>::= <операционная часть оператора подстановки без переменных> | <операционная часть оператора определения вхождения без переменных>

<операционная часть оператора типа подстановки с переменными>::= <операционная часть оператора подстановки с переменными> | <операционная часть оператора определения вхождения с переменными>

<операционная часть оператора типа подстановки>::= <операционная часть оператора подстановки> | <операционная часть оператора определения вхождения>

<оператор типа подстановки без переменных>::= <указатель постоянной границы> <операционная часть оператора типа подстановки без переменных>

<оператор типа подстановки с переменными и постоянной границей>::= <указатель постоянной границы> <операционная часть оператора типа подстановки с переменными>

<оператор типа подстановки с переменными>::= <указатель переменной границы> <операционная часть оператора типа подстановки>

5.2.2. Правильная структура относительно оператора типа подстановки

Пусть α, β - любые символы алфавита, неизменные в определениях пункта 5.2.2.

5.2.2.1. Синтаксис

<слово, открытое для оператора типа подстановки>::= <слово не содержащее слов ((,))>

<слово, закрытое для оператора типа подстановки>::= ((<слово, не содержащее слов ((,)) >))

<промежуточное слово>::= <слово, открытое для оператора типа подстановки> | <слово, закрытое для оператора типа подстановки> | <промежуточное слово> <слово, открытое для оператора типа подстановки> | <промежуточное слово> <слово, закрытое для оператора типа подстановки>

<область действия оператора подстановки>::= <промежуточное слово>

<описание значений переменных символов и слов>::= значение переменных <пусто> | значение переменных <список значений переменных>

<список значений переменных>::= <список значений переменных> <элемент списка значений переменных>

<элемент списка значений переменных>::= <переменный символ> * <значение переменного символа> * | <переменное слово> * * | <переменное слово> * <значение переменного слова> * | <значение переменной границы>

<значение переменного слова>::= <слово, не содержащее символов *>

<значение переменной границы>::= <переменное имя границы: * <имя границы> *

<значение переменного символа>::= <символ алфавита> | <ограничитель> | <спецификаторы>

<слово A>::= <промежуточное слово, не содержащее в словах, открытых для оператора типа подстановки, слов граница α и $\Gamma\alpha$>

<слово B>::= <область действия оператора типа подстановки, не содержащая в словах, открытых для оператора типа подстановки слов граница α и $\Gamma\alpha$>

<слово C>::=<промежуточное слово, не содержащее в словах, открытых для оператора типа подстановки, слов граница α и $\Gamma\alpha$ и символа значения переменных>

<слово D>::=<промежуточное слово, не содержащее в словах, открытых для оператора типа подстановки, символа значения

переменных >

< правильная структура относительно оператора типа подстановки без переменных > : : = начало программы < слово > в α < операционная часть оператора типа подстановки без переменных > ; < слово A > граница α < слово B > граница α < слово > конец программы

< правильная структура относительно оператора типа подстановки с переменными и постоянной границей > : : = начало программы < слово > в α < операционная часть оператора типа подстановки с переменными > ; < слово C > < описание значений переменных символов и слов > ; < слово A > граница α < слово B > граница α < слово > конец программы

< правильная структура относительно оператора типа подстановки с переменными > : : = начало программы < слово > в символ β < операционная часть оператора типа подстановки > ; < слово D > < описание значений переменных символов и слов, содержащее слово символ $\beta * \alpha *$ > ; < слово A > граница α < слово B > граница α < слово > конец программы

5.2.2.2. Семантика

Представим программу в виде слова ABC, где B -оператор подстановки.

Для облегчения описания правил выполнения оператора подстановки введем дополнительные символы i_1, i_2, i_3, i_4, i_5, i_6, i_7, i_8, слово', символ' не являющиеся символами языка, и символы x, y, z, обозначающие произвольные символы языка.

В слове C в направлении слева направо все подслова, ограниченные словами ((,)) и не содержащие вхождений этих слов, нумеруются и называются закрытыми отрезками. Пусть максимальный номер закрытого отрезка n.

Если $n=0$, то слово C называется начальным открытым отрезком оператора B.

Если $n>0$, то непустое подслово слова C, ограниченное слева оператором B, справа - первым закрытым отрезком, называется начальным открытым отрезком оператора B; непустое подслово слова C, ограниченное слева первым закрытым отрезком, справа - вторым, называется первым открытым отрезком оператора B и т.д.; непустое подслово слова C, ограниченное слева n -м закрытым отрезком, справа пустым концом слова C, называется заключительным отрезком.

Реализацию оператора подстановки можно свести к выполнению следующих действий.

А. Разметка оператора

1) Если левая часть подстановки не является словом подстановки, ограниченным разделителем, то левая часть подстановки ограничивается слева и справа символом i_1.

2) Если левая часть подстановки является словом подстановки, ограниченным разделителем, то символ i_1 ставится после начального разделителя и перед конечным разделителем левой части подстановки.

3) Аналогично правилам 1), 2) производится разметка правой части оператора подстановки.

ОПРЕДЕЛЕНИЕ 1. Слово, стоящее между символами i_1 в левой (правой) части подстановки, называется собственной левой (правой) частью подстановки.

Б. Выделение области описания значений переменных символов и слов

Если оператор подстановки является оператором с постоянной границей с переменными или оператором с переменными, т.е. в оператор подстановки входят или символ символ, или символ слово или то и другое вместе, то в слове C в начальном открытом отрезке определяется первое вхождение символа значения переменных, если его нет, то в первом отрезке и т.д., пока вхождение символа значения переменных не будет найдено. Это вхождение заменяется словом значения переменных i_2, затем в программе справа от i_2 первое вхождение символа ;, не входящее ни в какой элемент списка значений переменных символов и слов,

заменяется на слово i_2 ;

ОПРЕДЕЛЕНИЕ 2. Слово, стоящее между символами i_2, называется собственной областью описания значений переменных символов и слов.

В. Выделение области действия оператора подстановки

I) Если оператор подстановки является оператором с постоянной границей, например, x, где x – символ алфавита, то в слове C в начальном открытом отрезке определяется первое вхождение слова граница x ($\lceil x$), если его там нет, то эта процедура продолжается для первого открытого отрезка и т.д., пока вхождение слова граница x ($\lceil x$) не будет найдено. Это вхождение слова граница x ($\lceil x$) заменяется словом граница $x\, i_4$ ($\lceil x\, i_4$); правее символа i_4 аналогичным образом отыскивается вхождение слова граница x ($\lceil x$) и заменяется словом i_4 граница x ($i_4 \lceil x$).

2) Если оператор подстановки является оператором с переменной границей, например символ y, где y – символ алфавита, то в собственной области описания значений переменных символов и слов определяется первое вхождение слова символ y, затем выбирается символ, являющийся значением этого переменного символа, выбранному символу придается смысл постоянной границы и относительно этой постоянной границы выполняется пункт В I).

ОПРЕДЕЛЕНИЕ 3. Слово, стоящее между символами i_4, называется собственной областью действий оператора типа подстановки.

Г. Определение вхождения левой части оператора типа подстановки

I) В слове, ограниченном символами i_4, определяются открытые отрезки для оператора аналогично тому, как это делалось

в слове C.

2) Начальный (или первый, если нет начального) открытый отрезок ограничивается слева и справа символами i_3.

3) За первым вхождением символа i_3 ставятся два символа i_5 и i_7; символ i_5 имеет смысл метки начала вхождения, i_7 – метки конца вхождения.

4) За первым вхождением символа i_7 ставится символ i_9.

5) Если просмотр собственной левой части подстановки закончен или если левая часть подстановки пустая, то в этом случае будет иметь место вхождение слова $i_9\, i_7$ в левую часть подстановки. В этом случае символ i_9 заменяется пустым и выполняется пункт 24).

6) Если в собственной левой части подстановки за символом i_9 стоит переменное слово, т.е. имеет место вхождение слова i_9 слово x, где x является именем переменного слова, то символ i_9 ставится после переменного слова, старый символ i_9 заменяется пустым и далее выполняется пункт I4).

7) Проверяется, не стоит ли символ i_7 перед концом области действия оператора подстановки; если имеет место вхождение слова $i_7\, i_3$, то это означает, что вхождения левой части подстановки в открытый отрезок, ограниченный символами i_3, не существует, и в этом случае выполняется пункт 28.

8) Если после символа i_9 стоит переменный символ, т. е. имеет место вхождение слова i_9 символ y, где y – имя переменного символа, то символ i_9 переносится через два символа вправо и выполняется пункт II.

9) Если символ i_9 стоит перед символом z, который не является ни символом символ, ни символом слово, то символ i_9 перебрасывается через этот символ.

I0) Проверяется вхождение слова $i_7\, z$ в собственную область действия оператора подстановки. Если это вхождение есть, то в области оператора подстановки вхождение слова $i_7\, z$ заменяется словом $z\, i_7$, затем выполняется пункт 5. Если же вхождение слова $i_7\, z$ нет, то выполняется пункт I9.

II) Пункты II, I2, I3 являются пунктами определения значения переменного символа. Пусть именем переменного символа является тот же символ y, что и в пункте 8. В собственной области значений переменных определяется вхождение слова символ

y .Если вхождение есть,то со значением этого переменного символа, например, z выполняется пункт I0.

I2) В собственной области действия оператора подстановки определяется вхождение слова символ' y . Если это вхождение есть, то символ, следующий за этим вхождением,например, z является значением переменного символа символ y ,и с этим значением выполняется пункт I0.

I3) Этот пункт выполняется в том случае, когда значение переменного символа не определено. В этом случае перед символом i_7 ставится слово символ' y и символ i_7 переносится вперед через один символ. Последнему символу придается смысл значения переменного символа символ y . Далее выполняется пункт 5.

I4) Пункты I4,I5 и I6 являются пунктами определенного значения переменного слова. Именем переменного слова в этих пунктах принимается символ x , такой же, как и в пункте 6.

В собственной области значений переменных определяется вхождение слова слово x . Если значение переменного слова слово x определено, то выполняется пункт I7.

I5) В собственной области действия оператора подстановки определяется вхождение слова слово' x , слову, стоящему между этим вхождением и символом i_7 , придается смысл значения переменного слова слово x . Если это значение переменного слова определено, то выполняется пункт I7.

I6) Этот пункт выполняется в том случае, когда значение переменного слова не определено.В этом случае выполняется следующее. Первое вхождение символа i_7 заменяется на пустое,первое вхождение символа i_9 заменяется на слово $i_7\, i_9$. В собственной области действия оператора подстановки вхождение символа i_6 , если оно есть, заменяется на i_7 , а вхождение символа i_7 заменяется на слово' $x\, i_6\, i_7$. Далее выполняется пункт 5).

I7) Если начало слова, стоящего между символами i_2 и i_3, равно значению переменного слова слово x , то символ i_7 переносится через это начало вправо и выполняется пункт 5, иначе выполняется следующий пункт.

I8) Если имеет место вхождение в программу слова $i_2\, i_3$, то выполняется пункт 28.

I9) Если в области действия оператора подстановки есть вхождение символа i_6 , то справа от него ставится символ i_9, далее выполняется пункт 2I.

20) В области действия оператора подстановки символ i_5 заменяется на $i_5\, i_9$.

2I) В слове, ограниченном символами i_9 , i_{11}, заменяются все слова, имеющие вид символ' y или слово' x , пустым словом.

22) В собственной левой части подстановки символ i_9 заменяется пустым символом, а первое вхождение символа i_1 заменяется словом $i_1\, i_9$.

23) В собственной области действия оператора подстановки символ i_7 заменяется пустым символом, а слово вида $x\, i_9\, y$,где x – один из символов i_5 или i_6 , а y – произвольный символ, стоящий после i_9 , заменяется на слово $y\, x\, i_7$. Далее выполняется пункт 5.

24) В собственной области действия оператора подстановки символ i_6 заменяется на символ i_7 .

25) Каждое вхождение слова вида символ' xy , где x является символом алфавита, а y – произвольный символ языка , заменяется на символ y , одновременно с этим после вхождения символа i_2 в программу вписывается слово символ $x*y*$.

26) Каждое вхождение слова вида слово' $xy\, i_7$, где x – символ алфавита, а y – слово, не содержащее символ i_7 , заменяется на слово y ; одновременно с этим после первого вхождения символа i_2 в программу вписывается слово слово $x*y*$.

27) В левой части подстановки все вхождения символа i_7 заменяются пустым символом, далее выполняется Д. Это означает, что оператор подстановки применим.

28) В собственной области действия оператора подстановки символ i_6 заменяется на символ i_7 .

29) Каждое вхождение слова вида слово' $x\, y\, i_7$, где x является символом алфавита, а y – слово, не содержащее символа i_7 , заменяется на слово вида y .

30)Каждое вхождение слова вида символ' xy , где x является символом алфавита, а y – произвольный символ языка, заменяется на символ вида y .

3I) В левой части подстановки все вхождения символа i_7 заменяются пустым символом. Вспомогательные символы i_3 , i_5 ,

z_7 заменяются пустым символом. В собственной области действия оператора типа подстановки символами z_3 ограничивается очередной открытый отрезок, если он есть. Затем выполняются пункты A.I. ÷ A.3, после чего выполняется пункт Г. Если очередного открытого отрезка нет, то выполняется Е. Это означает, что оператор подстановки не применим.

Д. Подстановка правой части подстановки

I) В области действия оператора подстановки слово, стоящее между символами z_5 и z_7, заменяется пустым словом.

2) Слово, стоящее в правой части подстановки между символами z_7, вставляется между символами z_5 и z_7.

3) Все вхождения переменных символов и слов заменяются на их значения.

Е. Все вспомогательные символы z_4, z_3, z_2, z_1, z_5, z_7 заменяются пустым символом.

ПРИМЕЧАНИЕ. Оператор определения вхождения эквивалентен по действию оператору подстановки, у которого левая и правая части подстановки совпадают.

5.3. Условный оператор

5.3.I. Синтаксис

<Условный оператор> ::= если <оператор подстановки без переменных> применима то <слово условного оператора> | если <оператор подстановки без переменных> не применима то <слово условного оператора> | если <оператор определения вхождения без переменных> есть то <слово условного оператора> | если <оператор определения вхождения без переменных> нет то <слово условного оператора> | если <оператор подстановки с переменными и постоянной границей> применима то <слово условного оператора> | если <оператор подстановки с переменными и постоянной границей> не применима то <слово условного оператора> | если <оператор определения вхождения с переменными и постоянной границей> есть то <слово условного оператора> | если <оператор определения вхождения с переменными и постоянной границей> нет то <слово условного оператора>

<слово условного оператора> ::= <слово, не содержащее символа ; и не начинающееся с символов | , разделитель> | <слово, ограниченное разделителем>

5.3.2. Правильная структура относительно условного оператора

5.3.2.I. Синтаксис

Правильная структура относительно условного оператора для операторов, стоящих на I ÷ 4 местах в определении условного оператора в пункте 5.3.I., совпадает с правильной структурой относительно оператора типа подстановки с переменными и постоянной границей; для операторов, стоящих на 9 ÷ I2 местах, - с правильной структурой относительно оператора типа подстановки с переменными.

5.3.2.2. Семантика

Пусть программа представлена в виде слова ABC, где B - условный оператор.

I) Оператор подстановки (определения вхождения) выполняется.

2) Если он применим и после него в слове стоит символ применима (есть), то чтение программы продолжается с первого символа вхождения в B слова условного оператора оператора B; если не применим (вхождения нет) и после него в B стоит символ применима (есть), то чтение программы продолжается с первого символа слова C

Другой вариант пункта 2.

2) Если оператор подстановки (определения вхождения) применим и после него в слове B стоит символ не применима (нет), то чтение программы продолжается с первого символа слова C ; если оператор не применим и после него в слове B стоит символ не применима (нет), то чтение программы продолжается с первого символа вхождения в B слова условного оператора оператора B.

5.4. Оператор чтения

5.4.1. Синтаксис

< оператор чтения > ::= < оператор двустороннего чтения > | < оператор одностороннего чтения >

< оператор двустороннего чтения > ::= < оператор двустороннего чтения с постоянной границей > | < оператор двустороннего чтения с переменной границей >

< оператор двустороннего чтения с постоянной границей > ::= в < имя границы > читать < имя метки >

< оператор двустороннего чтения с переменной границей > ::= в < переменное имя границы > читать < имя метки >

< оператор одностороннего чтения > ::= читать справа < имя метки > | читать слева < имя метки >

< метка > ::= < имя метки > :

< имя метки > ::= < символ алфавита >

5.4.2. Правильная структура относительно оператора двустороннего чтения

Пусть α , β , γ - любые символы алфавита, неизменные в определениях пункта 5.4.2.1.

5.4.2.1. Синтаксис

< слово E > ::= < слово области действия оператора двустороннего чтения, не содержащее в словах, открытых для оператора двустороннего чтения, слов $\Gamma\alpha$ и граница α >

< слово F > ::= < слово области действия оператора двустороннего чтения, не содержащее в словах, открытых для оператора двустороннего чтения, слов $\Gamma\alpha$, граница α и γ : >

< слово G > ::= < слово области действия оператора двустороннего чтения, не содержащее в словах, открытых для оператора двустороннего чтения, слов $\Gamma\alpha$, граница α , γ : и символа значения переменных >

< слово H > ::= < слово области действия оператора двустороннего чтения, не содержащее в словах, открытых для оператора двустороннего чтения, слов $\Gamma\alpha$, граница α и символа значения переменных >

< слово области действия оператора двустороннего чтения > ::= < слово, открытое для оператора двустороннего чтения > | < слово, закрытое для оператора двустороннего чтения > | < слово области действия оператора двустороннего чтения > < слово, открытое для оператора двустороннего чтения > | < слово области действия оператора двустороннего чтения > < слово, закрытое для оператора двустороннего чтения >

< слово, открытое для оператора двустороннего чтения > ::= < слово, не содержащее слов $\leq\leq$, $\geq\geq$ >

< слово, закрытое для оператора двустороннего чтения > ::= $\leq\leq$ < слово, не содержащее слов $\geq\geq$ > $\geq\geq$

< правильная структура относительно оператора двустороннего чтения с постоянной границей > ::= начало программы < слово > граница α < слово E > γ : < слово F > в α читать γ ; < слово F > граница α < слово > конец программы | начало программы < слово > граница α < слово F > в α читать γ ; < слово F > γ : < слово E > граница α < слово > конец программы

< правильная структура относительно оператора двустороннего чтения с переменной границей > ::= начало программы < слово > граница α < слово E > γ : < слово F > в символ β читать γ ; < слово G > < описание значений переменных символов и слов, содержащее слово символ $\beta * \alpha *$ >; < слово F > граница α < слово > конец программы | начало программы < слово > граница α < слово F > в символ β читать γ ; < слово G > γ : < слово H > < описание значений переменных символов и слов, содержащее слово символ $\beta * \alpha *$ >; < слово E > граница α < слово > конец программы | начало программы < слово > граница α < слово E > γ : < слово F > в символ β читать γ ; < слово G > граница α < слово, не содержащее символа значения переменных > < описание зна-

чений переменных символов и слов, содержащее слово символ $\beta *\alpha *>$; <слово> конец программы | начало программы <слово> граница α <слово F> в символ β читать γ : <слово G> γ : <слово H> граница α <слово, не содержащее символа значения переменных> <описание значений переменных символов и слов, содержащее слово символ $\beta*\alpha*>$; <слово> конец программы

5.4.2.2. Семантика

Представим программу в виде слова ABC, где B -оператор чтения.

Для облегчения описания правил выполнения этого оператора введем символы i_1, i_2, не являющиеся символами языка, и символы x, y, z, обозначающие произвольные символы алфавита языка.

Закрытые отрезки в том или ином подслове определяются аналогично тому, как это делалась в пункте 5.2.2.2., только символы (,) заменяются символами $\leq$, $\geq$, соответственно. Открытые отрезки определяются полностью аналогично тому, как это делалась в пункте 5.2.2.2.

5.4.2.2.I. Пусть B -оператор двустороннего чтения с постоянной границей x и именем метки y.

I) В слове A в заключительном (если его нет, то в начальном) отрезке, открытом для оператора B, определяется последнее вхождение слова граница x (Γx). Если его там нет, то процедура продолжается для $n-1$-го открытого отрезка и т.д. до тех пор, пока вхождение слова граница x (Γx) не будет найдено.

2) Это вхождение граница x (Γx) заменяется символом граница $x\, i_1$ ($\Gamma x\, i_1$).

3) В слове C в начальном отрезке, открытом для оператора B, определяется первое вхождение слова граница x (Γx). Если его там нет, то процедура продолжается для первого отрезка и т.д. до тех пор, пока вхождение слова граница x (Γx) не будет найдено.

4) Это вхождение слова граница x (Γx) заменяется словом i_1 граница x ($i_1 \Gamma x$).

5) Представим слово, заключенное между символами i_1, в виде слова DBY. Если в слове D есть открытые для оператора B отрезки, то в последнем отрезке, открытом для оператора B, определяется последнее вхождение слова $y:$, если его там нет, то эта процедура продолжается для предпоследнего открытого отрезка и т.д. Если вхождения слова $y:$ в открытых отрезках слова D нет или это слово не имеет отрезков, открытых для оператора B, то в слове Y в начальном отрезке, открытом для оператора B, определяется первое вхождение слова $y:$, если его там нет, то эта процедура продолжается для первого открытого отрезка слова Y и т.д. Таким способом может быть обнаружено одно и только одно вхождение слова $y:$.

6) Это вхождение слова $y:$ заменяется словом $y: i_2$.

7) Все символы i_1 заменяются пустым словом.

8) Дальнейшее чтение программы должно быть продолжено с символа i_2.

9) Символ i_2 заменяется пустым словом.

5.4.2.2.2. Пусть B -оператор двустороннего чтения с переменной границей символ x и именем метки y.

В подслове C в первом вхождении описания значений переменных символов и слов определяется первое вхождение слова символ x, символу z из слова $*z*$, следующего за вхождением этого подслова, придается смысл постоянной границы и относительно этой постоянной границы повторяются все пункты 5.4.2.2.I.

5.4.3. Правильная структура относительно оператора одностороннего чтения

Пусть γ - любой символ алфавита, неизменный в определениях пункта 5.4.3.I.

5.4.3.I. Синтаксис

<правильная структура относительно оператора левосторон-

него чтения >::= начало программы < слово > γ : < слово, не содержащее слово γ: > читать слева γ ; < слово > конец программы

< правильная структура относительно оператора правостороннего чтения >::= начало программы < слово > читать справа γ; < слово, не содержащее слово γ: > γ : < слово > конец программы .

5.4.3.2. Семантика

Представим программу в виде слова ABC , где B - оператор одностороннего чтения.

Для облегчения описания правил выполнения этого оператора введем символ i_1 и символ γ , обозначающий произвольный символ алфавита языка.

5.4.3.2.I. Пусть B -оператор левостороннего чтения с именем метки γ .

I) В слове A отыскивается последнее вхождение слова γ: и заменяется словом $\gamma : i_1$.

2) Дальнейшее чтение программы должно быть продолжено с символа i_1 .

3) Символ i_1 заменяется пустым словом.

5.4.3.2.2. Пусть B -оператор правостороннего чтения с именем метки γ .

I) В слове C отыскивается первое вхождение слова γ: и заменяется словом $\gamma : i_1$.

2) Далее 2) и 3) из 5.4.3.2.I.

5.5. Оператор переписи

5.5.I. Синтаксис

< оператор переписи >::= < оператор переписи переменных > | < оператор переписи слов >

< оператор переписи переменных >::= переписать < переменный символ > к < символ переписи > | переписать < переменное слово > к < символ переписи >

< оператор переписи слов >::= переписать < левая граница переписи > < правая граница переписи > к < символ переписи >

< левая граница переписи >::= < символ алфавита >

< правая граница переписи >::= < символ алфавита >

< символ переписи >::= < символ алфавита >

5.5.2. Правильная структура относительно оператора переписи

Пусть α , β , γ , δ , μ , ν - любые символы алфавита, неизменные в определениях пункта 5.5.2.I.

5.5.2.I. Синтаксис

< слово M >::= < слово, не содержащее символ γ >

< слово N >::= < слово, не содержащее символы значения переменных и γ >

< слово P >::= < описание значений переменных символов и слов, содержащее слово символ α и не содержащее символ γ >

< слово Q >::= < слово, не содержащее символ значения переменных >

< слово R >::= < описание значений переменных символов и слов, содержащее слово символ α >

< слово S >::= < описание значений переменных символов и слов, содержащее слово слово β и не содержащее символ γ >

< слово T >::= < описание значений переменных символов и слов, содержащее слово слово β >

< слово U >::= < слово, не содержащее символ ν >

< слово W >::= < слово, не содержащее символы δ, μ, ν >

< слово U >::= < слово, не содержащее символы δ, μ >

< правильная структура относительно оператора переписи переменных >::= начало программы < слово > γ < слово M > переписать символ α к γ ; < слово N > < слово P > < слово M > конец программы | начало программы < слово M > переписать символ α к γ ; < слово N > < слово P > < слово M > γ < слово > ко-

нец программы | начало программы <слово M> переписать символ α к γ ; <слово N>γ<слово Q> <слово R> <слово> конец программы | начало программы <слово> γ <слово M> переписать слово β к γ ; <слово N> <слово S><слово M> конец программы | начало программы <слово M> переписать слово β к γ ; <слово N><слово S> <слово M>γ <слово> конец программы | начало программы <слово M> переписать слово β к γ ; <слово $N\gamma$ <слово Q> <слово T> <слово> конец программы

<правильная структура относительно оператора переписи слов> ::= начало программы <слово v>δ<слово w>μ<слово w> переписать $\delta\mu$ к v ; <слово w>ν<слово u> конец программы | начало программы <слово>δ<слово u>μ <слово u>ν<слово w> переписать $\delta\mu$ к ν ; <слово w> конец программы | начало программы <слово>δ<слово u> ν <слово w>μ <слово w> переписать $\delta\mu$ к ν ; <слово w> конец программы | начало программы <слово>δ<слово u>ν<слово w> переписать $\delta\mu$ к ν ; <слово w> μ <слово v> конец программы | начало программы <слово v>δ<слово w> переписать $\delta\mu$ к ν ; <слово w>μ<слово v>ν<слово> конец программы | начало программы <слово v>δ<слово w> переписать $\delta\mu$ к ν ; <слово w>ν <слово u>μ <слово> конец программы | начало программы <слово>ν <слово v>δ<слово w> переписать $\delta\mu$ к ν ; <слово w>μ<слово v> конец программы | начало программы <слово>ν <слово v>δ <слово w>μ <слово w> переписать $\delta\mu$ к ν ; <слово w> конец программы | начало программы <слово u>ν<слово w> переписать $\delta\mu$ к ν ; <слово w>δ<слово w>μ <слово v> конец программы | начало программы <слово w> переписать $\delta\mu$ к ν ; <слово w>δ<слово w>μ <слово v>ν <слово>. конец программы | начало программы <слово w> переписать $\delta\mu$ к ν ; <слово w>δ<слово w>ν<слово u>μ<слово> конец программы | начало программы <слово w> переписать $\delta\mu$ к ν ; <слово w>ν<слово u>δ<слово u>μ<слово> конец программы

5.5.2.2. Семантика

Представим программу в виде слова ABC, где B – оператор переписи.

Для облегчения описания правил выполнения оператора введем символы i_1, i_2, i_3, не являющиеся символами языка, и символы x, y, z, обозначающие произвольные символы алфавита языка.

5.5.2.2.I. Пусть B – оператор переписи переменного символа, например, переменного символа символ x. Этот оператор выполняется так.

I) Отыскивается последнее вхождение символа переписи y оператора B в подслово A или, если его там нет, первое вхождение символа переписи y в подслово C и заменяется словом $y i_1$.

2) В первом вхождении описания переменных символов и слов в подслово C определяется вхождение слова символ x и символу z из слова $*z*$, следующего за этим вхождением, придается смысл значения переменного символа оператора B.

3) Подслово $y i_1$ заменяется подсловом yz.

5.5.2.2.2. Пусть B – оператор переписи переменного слова, например, переменного слова слова x.

Выполнение этого оператора состоит в следующем.

I) Выполняется пункт I) из 5.5.2.2.I.

2) В первом вхождении описания переменных символов и слов в подслово C определяется первое вхождение слова слово x.

3) Слову, ограниченному символами $*$ и следующему за этим вхождением, придается смысл значения переменного слова оператора B. Обозначим это слово буквой F.

4) Подслово $y i_1$ заменяется словом yF.

5.5.2.2.3. Пусть B – оператор переписи слов, z – левая граница переписи этого оператора, x – правая.

Этот оператор выполняется так.

I) Выполняется пункт I) из 5.5.2.2.I.

2) Отыскивается последнее вхождение левой границы переписи z в подслово A или, если его там нет, первое вхождение z в слово C. Найденное вхождение заменяется словом $i_2 z$.

3) Правая граница переписи x в операторе B заменяет-

ся символом i_3 .

4) В подслове, расположенном правее символа i_2 , отыскивается первое вхождение правой границы переписи x оператора B Слово x заменяется словом $x i_2$.

5) Слово, расположенное между символами i_2 , обозначим буквой D . Подслово $y i_1$ заменяется словом $y D$.

6) Все вхождения символов i_1 , i_2 заменяются пустым словом.

7) Все символы i_3 заменяются символом x.

Оператор переписи не меняет характера чтения программы.

5.6. Оператор чистки

5.6.1. Синтаксис

< оператор чистки > ::= < оператор чистки переменных > | < оператор чистки слов > | < левый оператор чистки слов > | < правый оператор чистки слов >

< оператор чистки переменных > ::= стереть < список переменных символов и слов >

< список переменных символов и слов > ::= < переменный символ > | < переменное слово > | < список переменных символов и слов > < переменный символ > | < список переменных символов и слов > < переменное слово >

< оператор чистки слов > ::= стереть < левая граница стирания > < правая граница стирания >

< левый оператор чистки слов > ::= стереть слева < левая граница стирания > < правая граница стирания >

< правый оператор чистки слов > ::= стереть справа < левая граница стирания > < правая граница стирания >

< правая граница стирания > ::= < символ алфавита >

< левая граница стирания > ::= < символ алфавита >

5.6.2. Правильная структура относительно оператора чистки

Пусть α, β - любые символы алфавита, неизменные в определениях, приведенных ниже.

5.6.2.1. Синтаксис

< правильная структура относительно оператора чистки переменных > ::= начало программы < слово > < оператор чистки переменных > ; < слово, не содержащее символ значение переменных > < описание значений переменных символов и слов > < слово > конец программы

< слово $\mathcal{Z}$ > ::= < слово, не содержащее символов α, β >

< слово $\mathcal{J}$ > ::= < слово, не содержащее символа α >

< правильная структура относительно оператора чистки слов > ::= начало программы < слово > α < слово $\mathcal{Z}$ > β < слово $\mathcal{Z}$ > стереть $\alpha\beta$; < слово $\mathcal{Z}$ > конец программы | начало программы < слово > α < слово $\mathcal{Z}$ > стереть $\alpha\beta$; < слово $\mathcal{Z}$ > β < слово $\mathcal{J}$ > конец программы | начало программы < слово $\mathcal{Z}$ > стереть $\alpha\beta$; < слово $\mathcal{Z}$ > α < слово $\mathcal{Z}$ > β < слово > конец программы

< правильная структура относительно левого оператора чистки слов > ::= начало программы < слово > α < слово $\mathcal{Z}$ > β < слово $\mathcal{Z}$ > стереть слева $\alpha\beta$; < слово > конец программы

< правильная структура относительно правого оператора чистки слов > ::= начало программы < слово > стереть справа $\alpha\beta$; < слово $\mathcal{Z}$ > α < слово $\mathcal{Z}$ > β < слово > конец программы

5.6.2.2. Семантика

Представим программу в виде слова ABC , где B - оператор чистки.

Для облегчения описания правил выполнения оператора введем символы i_1 , i_2 , не являющиеся символами языка, и символы x , y , , обозначающие произвольные символы алфавита языка.

5.6.2.2.1. Пусть B - оператор чистки переменных. Выполнение этого оператора состоит в том, что в первом вхождении описания значений переменных символов и слов в подслово C все вхождения элементов списка значений переменных из списка оператора заменяются пустым словом.

5.6.2.2.2. Пусть B - оператор чистки слов, x - левая граница стирания этого оператора, y - правая.

Выполнение этого оператора осуществляется так.

1) Отыскивается последнее вхождение левой границы стирания x в подслово A , или, если его нет, то первое вхождение этой левой границы в подслово C и заменяется словом $x i_1$.

2) Правая граница стирания y в операторе B заменяется сим-

волом i_2.

3) В подслове, расположенном правее символа i_1, отыскивается первое вхождение правой границы стирания y и заменяется словом $i_1 y$.

4) Подслово, началом которого служит слово $x i_1$, концом $i_1 y$, заменяется пустым словом.

5) Символ i_2 заменяется символом x.

5.6.2.2.3. Пусть B – левый (правый) оператор чистки слов, x – левая граница стирания этого оператора, y –правая.

Выполнение этого оператора осуществляется следующим образом.

I) Отыскивается последнее (первое) вхождение левой границы стирания x в подслове A (C) и заменяется словом $x i_1$.

2) В подслове слова A (C), расположенном правее символа i_1, отыскивается первое вхождение правой границы стирания y и заменяется словом $i_1 y$.

3) Подслово, началом которого служит слово $x i_1$, концом – $i_1 y$. заменяется пустым словом.

Оператор чистки не меняет характера чтения программы.

5.7. Оператор конца

5.7.I. Синтаксис

< оператор конца >::= стоп

5.7.2. Правильная структура относительно оператора конца.

< правильная структура относительно оператора конца >::= начало программы < слово > стоп < слово > конец программы

5.7.3. Семантика

Выполнение оператора конца состоит в прекращении чтения программы.

§6. Пример программы, написанной на языке JP-I

В качестве алфавита языка JP-I возьмем объединение букв русского, греческого и латинского алфавитов, цифр и символов +,

=, Σ, нуль, ,

Опишем программу сложения двух целых положительных чисел любой разрядности

Пусть первое слагаемое α ограничено слева буквой a, справа-буквой b, второе-$\mathcal{Z}$ соответственно буквами c и d. Сумму требуется поместить между символами ρ и Σ.

К символу R нужно приписать слово "слагаемое не определено", если хотя бы одно из слагаемых пусто. Сложение чисел α и $\mathcal{Z}$ выполняется с использованием таблицы сложения T.

Эта таблица устроена так:

начало таблицы сложения

0 + I = 0I,
0 + 2 = 02,
. . . . . .
0 + 9 = 09
I + 0 = 0I
. . . . . .
I + 9 = I0,
2 + 0 = 02

конец таблицы сложения.

Разряды чисел α и $\mathcal{Z}$ нумеруются справа налево. Сложение начинается с младших разрядов. Если в процессе сложения одно из слагаемых кончается, а другое нет, то в дальнейшем осуществляется сложение разрядов большого слагаемого с нулем.

Пусть наступил момент, когда нужно найти сумму i -х разрядов чисел α и $\mathcal{Z}$ и перенос в $i+1$ -й разряд, причем i-й разряд α равен x, $\mathcal{Z}$-y, перенос из $i-1$-го разряда- z.

Сперва находится сумма i -х разрядов w и перенос в $i+1$ -й разряд без учета переноса из $i-1$ -го разряда. Затем осуществляется корреляция чисел w и v : w складывается с z, полученная сумма и есть значение i -го разряда; перенос x, возникший при сложении этих чисел, складывается с v. Их сумма z – это значение переноса в $i+1$ - й разряд.

Заключим в область с границами β слова a и b, c и d, T, нуль 0 и символ R.

ПРИМЕЧАНИЕ. При задании соответствующей таблицы эта программа может быть использована для сложения чисел в любой другой системе исчисления, выполнения таких логических операций ,

как конъюнкция, дизъюнкция, сложение по модулю 2.

Программа сложения целых положительных чисел любой разрядности

```
начало программы
    граница α
    если в β вхождение ab есть то в α читать A ;
    если в β вхождение cd есть то в α читать A ;
    в α читать B ;
    A : вβзаменить R на r слагаемое не определено;
    в α читать K ;
    B : вβзаменить B на mB ;
    в β заменить d на nd;
    в β определить вхождение нуль символ z ;
    L : если в β вхождение am есть то в α читать C ;
    в β заменить символ xm на m символ x ;
    в α читать D ;
    C : в β определить вхождение нуль символ x ;
    D : если в β вхождение cn есть то в α читать E ;
    в β заменить символ yn на n символ y ;
    M : в β определить вхождение символ x+ символ y=
символ v символ w ;
    стереть символ x символ y ;
    в β определить вхождение символ w+ символ z= символ x
символ y ;
    стереть символ z символ w ;
    в β определить вхождение символ x+ символ v= символ
w символ z ;
    в β заменить ρ на ρ символ y ;
    стереть символ x символ y символ w символ v ;
    в α читать L ;
    E : если в β вхождение am есть то в α читать P ;
    в β определить вхождение нуль символ y ;
    в α читать M ;
    P : если в β вхождение нуль символ z нет то в β заме -
нить ρ на ρ символ z ;
```



```
    в β заменить m на  ;
    в β заменить n на  ;
    K : стоп;
    граница α
    значения переменных;
    граница β
    RTρΣ a 2 1 b c Bd нуль 0
    граница β
конец программы
```

Литература

I. An INTRODUCTION to COMIT Programming. The Research Lab. of Electronics and the Computation Center, M.I.T.,1961.

2. The SNOBOL 3 Programming Language. The Bell System Technical Journal, vol. XLV, N 6,1966.

Поступила в редакцию
IO.IУ.I97I