1.3. ПОСТФІКСНЕ ПРЕДСТАВЛЕННЯ АРИФМЕТИЧНИХ ВИРАЗІВ

В *інфіксній* формі запису оператор двомісної операції стоїть між операндами, наприклад, сума двох змінних записується так: `A+B`. Це найбільш прийнята форма запису, яка застосовується при математичних записах, у більшості мов програмування для представлення арифметичних записів.

В *префіксній* формі запису оператор двомісної операції стоїть перед операндами, наприклад, `+AB`. Ця форма запису використовується для виклику функцій, наприклад, `SUM(A,B)`.

У *постфіксній* формі запису оператор двохмісної операції стоїть після операндів, наприклад, `AB+`. Постфіксна форма є цікавою тим, що вона дозволяє представити будь-який вираз без застосування дужок, при цьому для підрахунку значення виразу використовується стек. Тому в компіляторах мов програмування, які оперують з арифметичними виразами, спочатку здійснюється перетворення виразу з інфіксної форми в постфіксну форму, яка реалізується з використанням стеку. Наведемо кілька простих прикладів перетворення виразів з інфіксної форми в постфіксну.

Приклад 1.4

`A+B*C` → `A+(B*C)` → `A+(BC*)` → `A(BC*)+` → `ABC*+`

Приклад 1.5

`(A+B)*C` → `(AB+)*C` → `(AB+)C*` → `AB+C*`

Обчислення виразу в постфіксній формі здійснюється за таким алгоритмом. Рядок сканується зліва направо доки не зустрінеться перший оператор, який здійснює операцію з двома операндами, що знаходяться ліворуч, результат операції заноситься в системний регістр (змінна, що виділена для збереження результату операції), який виступає як проміжний операнд. Далі сканування продовжується і найближчий справа оператор виконує операцію з двома новими операндами, що знаходяться ліворуч, при чому операндом може бути й системний регістр.

В табл. 1.1 наведено інші приклади запису еквівалентних виразів в інфіксній та постфіксній формах, де оператор «↑» відповідає операції «піднесення до ступеню».

Таблиця 1.1

Приклади запису в інфіксній та постфіксній формах

Інфіксна форма	Постфіксна форма
A+B	AB+
A+B-C	AB+C-
(A+B)*(C-D)	AB+CD-*
A↑B*C-D+E/F/(G+H)	AB↑C*D-EF/GH+/+
(A+B)*C-(D-E)↑(F+G)	AB+C*DE-FG+↑-
A-B/(C*D↑E)	ABCDE↑*/-

Алгоритм перетворення виразу з інфіксної форми в постфіксну в загальному вигляді виглядає так: спочатку здійснюється перетворення операції найвищого пріоритету (№ 1) одного рівня в операнди постфіксної форми, потім – наступного рівня і т. д. до найнижчого рівня. Наприклад, у виразі A+B*C послідовність операцій <+, *>, а порядок їх виконання <*, +>, тобто при інфіксній формі матиме вираз ABC*+. А якщо для завдання пріоритетів використовуються дужки, спочатку за вище наведеним правилом виконуються перетворення на самому значущому вищому рівні пріоритету (самі внутрішні дужки) і вираз в дужках має бути перетворений в операнд, потом наступний і т. д. до самої зовнішньої пари дужок. Наприклад, у виразі (A+B)*C і послідовність операцій <+, *>, і порядок їх виконання такий же <+, *>, тобто при інфіксній формі матиме вираз AB+C*. Цей алгоритм легко реалізується стеком: якщо дужка відкрилася та її внесено в стек, то наступна аналогічна дужка теж заноситься в стек і т. д., поки не дійдемо до першої закриваючої дужки, що буде означати найвищий пріоритет в межах самих зовнішніх дужок. Після виконання операцій в цих дужках і вилучення зі стеку обох останніх дужок (закриваючої і відкриваючої) будемо йти до наступної закриваючої дужки і т. д. В алгоритмі треба також передбачити врахування

пріоритетів арифметичних операцій. Для демонстрації алгоритму використаємо таблицю прикладу 1.5, в якій в колонці «Symbol» будемо розміщувати поточне значення операнда або оператора при його скануванні; в колонці «Postf» – поточне значення виразу в постфіксній формі, при цьому до нього заносяться операнди при кожній появі в колонці «Symbol», а оператори – при «розвантаженні» стеку (колонка Stack), куди вони заносяться в порядку появи при скануванні. Розвантажування стеку (перенесення символів операторів в кінець поточного значення колонки «Postf») здійснюється за такими умовами:

1) завершено сканування всього рядка, що являє собою вираз в інфіксній формі;

2) черговий оператор, що прочитаний сканером, має пріоритет нижчий, ніж у оператора, що знаходиться в стеку на вищій позиції (в таблиці – крайній справа). При цьому прочитаний оператор заноситься в стек;

3) зустрілася закриваюча дужка «)». При цьому розвантажується вміст стеку тільки до першої відкриваючої дужки. Самі дужки не переносяться, а відкриваюча дужка просто видаляється зі стеку.

Приклад 1.5

Перетворення виразу A+B*C з інфіксної форми в постфіксну

№	Symbol	Postf	Stack	Comments
1	A	A		
2	+	A	+	
3	B	AB	+	
4	*	AB	+*	
5	C	ABC	+*	Прочитано останній символ, треба розвантажувати стек
6		ABC*	+	Зчитано та додано до «Postf» верхній елемент стеку
7		ABC*+		Зчитано та додано до «Postf» наступний (він і останній) елемент стеку

Приклад 1.6

Перетворення виразу A*B+C з інфіксної форми в постфіксну

№	Symbol	Postf	Stack	Comments
1	A	A		
2	*	A	*	
3	B	AB	*	
4	*	AB	*+	Операція «+» має менший пріоритет у порівнянні з операцією «*», тому розвантажуємо стек – символ операції «*» переноситься до «Postf»
5		AB*	+	
5	C	AB*C	+	Прочитано останній символ, треба розвантажувати стек
6		AB*C+		Зчитано зі стеку та додано до «Postf» вміст стеку

Приклад 1.7

Перетворення виразу `(A+B)*C` з інфіксної форми в постфіксну

№	Symbol	Postf	Stack	Comments
	(		(	
1	A	A	(	
2	+	A	(+	
3	B	AB	(+	
4	)	AB	(+	зустрілася закриваюча дужка «)», тому розвантажуємо стек – символ операції «+» переноситься до «Postf», а дужка видаляється
5		AB+		
	*	AB+	*	
5	C	AB+C	*	Прочитано останній символ, треба розвантажувати стек
6		AB+C*		Зчитано зі стеку та додано до «Postf» вміст стеку

Приклад 1.8

Перетворення виразу `((A-(B+C))*D)↑(E+F)` (без коментарів)

№	Symbol	Postf	Stack
1	(		(
2	(		((
3	A	A	((
4	-	A	((-
5	(	A	((-(
6	B	AB	((-(
7	+	AB	((-(+
8	C	ABC	((-(+
9	)	ABC+	((-
10		ABC+-	(
11	*	ABC+-	(*
12	D	ABC+-D	(*
13	)	ABC+-D*	
14	↑	ABC+-D*	↑
15	(	ABC+-D*	↑(
16	E	ABC+-D*E	↑(
17	+	ABC+-D*E	↑(+
18	F	ABC+-D*EF	↑(+
19	)	ABC+-D*EF+	↑
20		ABC+-D*EB+↑	