Практикум на ЭВМ. Интерпретатор. Полиз

Баев А.Ж.

Казахстанский филиал МГУ

08 февраля 2019

План на семестр

1. Интерпретатор
2. Web сервер
3. Параллельное программирование

Интерпретатор

1. Арифметические операторы
2. Оператор присваивания
3. Логические операторы
4. Оператор перехода (goto)
5. Условный оператор
6. Цикл while
7. Массивы
8. Функции
9. Рекурсия (стек для вызова функций)

Интерпретатор (полиз)

Пример:

$$1 + 2$$

Польская инверсная запись:

1	2	+

Стек для вычислений:

1

1		

2

1	2	

+

3		

Интерпретатор (полиз)

Пример:

$$1 + 2 * 4$$

Польская инверсная запись:

1	2	4	*	+

Стек для вычислений:

1	1		
2	1	2	
4	1	2	4
*	1	8	
+	9		

Интерпретатор (полиз)

Пример:

$$(1 + 2) * 4$$

Польская инверсная запись:

1	2	+	4	*

Стек для вычислений:

1	1		
2	1	2	
+	3		
4	3	4	
*	12		

Интерпретатор (полиз)

Пример:

$$1 + 2 * 3 * (9 - 4)$$

Польская инверсная запись:

1	2	3	$*$	9	4	$-$	$*$	$+$

Стек для вычислений:

1	1				
2	1	2			
3	1	2	3		
$*$	1	6			
9	1	6	9		
4	1	6	9	4	
$-$	1	6	5		
$*$	1	30			
$+$	31				

Алгоритм построения полиза «сортировочная станция»

Пример:

$$1 + 2$$

символ входной строки	стек операторов					выходная строка				
1						1				
+	+					1				
2	+					1	2			
конец строки						1	2	+		

Алгоритм построения полиза «сортировочная станция»

Пример:

$$1 + 2 * 4$$

символ входной строки	стек операторов					выходная строка				
1						1				
+	+					1				
2	+					1	2			
*	+	*				1	2			
4	+	*				1	2	4		
конец строки						1	2	4	*	+

Алгоритм построения полиза «сортировочная станция»

Пример:

$$(1 + 2) * 4$$

символ входной строки	стек операторов					выходная строка				
(	(					1				
1	(					1				
+	(	+				1				
2	(	+				1	2			
)						1	2	+		
*	*					1	2	+		
4	*					1	2	+	4	
конец строки						1	2	+	4	*

Алгоритм построения полиза «сортировочная станция»

Пример:

$$1 + 2 * 3 * (9 - 4)$$

Алгоритм построения полиза «сортировочная станция»

1. Читаем очередной символ.
2. Если символ является числом, добавляем его к выходной строке.
3. Если символ является открывающей скобкой, помещаем его в стек.
4. Если символ является закрывающей скобкой, то выталкиваем все элементы из стека в выходную строку, пока верхним элементом стека не станет открывающая скобка.
5. Если символ является бинарной операцией «oper», то пока на вершине стека операция на вершине стека имеет приоритет больше (или равен для левоассоциативной операции), чем «oper», то выталкиваем верхний элемент стека в выходную строку, в конце помещаем «oper» в стек.
6. Когда входная строка закончилась, выталкиваем все символы из стека (операторы) в выходную строку.

Реализация

```
class Lexem {
public:
    Lexem();
};

class Number: public Lexem {
    int value;
public:
    Number();
    int getValue();
};
```

Реализация

```
enum OPERATOR {
    PLUS, MINUS, MULTIPLY, LBRACKET, RBRACKET
};
int PRIORITY[] = {
    0, 0, 1, 2, 2
};

class Oper: public Lexem {
    OPERATOR opertype;
public:
    Oper();
    OPERATOR getType();
    int getValue(const Number& left,
                 const Number& right);
};
```

Реализация

```
#include <string>
#include <vector>

std::vector<Lexem> parseLexem(std::string input);
std::vector<Lexem> buildPoliz(std::vector<Lexem> infix);
int evaluatePoliz(std::vector<Lexem> poliz);
```