**Задание:**

Требуется выполнить программную реализацию автомата, вычисляющего значения арифметических выражений. Автомат выполняется в виде класса, содержащего метод, принимающий на вход выражение, а на выходе выдающий булевское значение, показывающее, является ли выражение корректным. Другим методом реализуемого класса является метод получения результата вычисления.

Выражение в класс передается как строка. Строка может содержать пробелы между операндами, операциями и скобками.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Тип операндов** | **Минимум** | **Максимум** | **Модель автомата** |
| 3 | Целый | -128 | 127 | Мили |

**Исходный код программы:**

#include <iostream>

#include <string>

#include <sstream>

#include <stack>

#include <vector>

using namespace std;

class mat {

public:

string s;

string errors;

void check(string str) {

int bracketcounter = 0;

string temp;

for (int i = 0; i < str.length(); i++) {

if (str[i]=='.') {

errors += "Number isn`t integer \n";

}

if (str[i] == '(')

bracketcounter++;

if (str[i] == ')')

bracketcounter--;

if (str[i] != '+' && str[i] != '-' && str[i] != '/' && str[i] != '\*' && str[i] != '('&&str[i] != ')'&&str[i] != ' ') {

temp.push\_back(str[i]);

}

else {

if (!temp.empty()) {

int num = atoi(temp.c\_str());

if (num > 127 || num < -128) {

errors += "Out of range \n";

}

}

temp.clear();

}

}

if (bracketcounter != 0) {

errors += "Incorrect brackets sequence";

}

if (errors.empty())

cout << "all right \n";

else {

cout << errors;

exit(0);

}

}

void RPN(string str) {

s = str;

stack<pair<char, int>> p;

string res;

int priority = 0;

for (int i = 0; i < s.length(); i++) {

if (s[i] == ' ')

continue;

if (isdigit(s[i])) {

do {

res = res + s[i];

i++;

} while (isdigit(s[i]));

res = res + ' ';

i--;

}

else if (s[i] == '\*' || s[i] == '/')

priority = 3;

else if (s[i] == '+' || s[i] == '-')

priority = 2;

else if (s[i] == '(' || s[i] == ')')

priority = 1;

if (s[i] == '(') {

p.push(make\_pair(s[i], priority));

priority = 0;

}

if (s[i] == ')') {

while (p.top().first != '(') {

res = res + p.top().first + ' ';

p.pop();

}

p.pop();

priority = 0;

}

if (priority > 0 && (p.empty() || p.top().second < priority)) {

p.push(make\_pair(s[i], priority));

priority = 0;

}

if (priority > 0 && p.top().second >= priority) {

while (p.top().second >= priority) {

res = res + p.top().first + ' ';

p.pop();

if (p.empty())//???

break;

}

if (priority > 0 && (p.empty() || p.top().second < priority)) {

p.push(make\_pair(s[i], priority));

priority = 0;

}

}

}

while (!p.empty()) {

res = res + p.top().first + ' ';

p.pop();

}

s = res;

}

void RP() {

stack<int> a;

double p1, p2;

istringstream is;

is.str(s);

while (is >> s) {

if (s == "+") {

p1 = a.top();

a.pop();

p2 = a.top();

a.pop();

a.push(p1 + p2);

}

else if (s == "\*") {

p1 = a.top();

a.pop();

p2 = a.top();

a.pop();

a.push(p1 \* p2);

}

else if (s == "-") {

p1 = a.top();

a.pop();

if (a.empty())

a.push(-p1);

else {

p2 = a.top();

a.pop();

a.push(p2 - p1);

}

}

else if (s == "/") {

p1 = a.top();

a.pop();

p2 = a.top();

a.pop();

a.push(p2 / p1);

}

else a.push(stod(s));

}

cout << a.top();

}

};

int main() {

string s;

getline(cin, s);

mat d;

d.check(s);

d.RPN(s);

cout << d.s;

d.RP();

}

**Блок схема и граф**







