**** для компіляції коду рекомендується використати http://cpp.sh***

Домашнє завдання №2

Розробити програму, яка за допомогою **`boost::spirit`** реалізовує заданий відповідно до варіанту синтаксис для запису виразу.

Вибір варіанту

(Nж + Nг + 6) % 10 + 1

де: Nж – порядковий номер студента в групі, а Nг – номер групи(1,2,3,4,5,6 або 7)

Варіанти завдання

№	Синтаксис *(заданий РБНФ)*
1	<expression> = <term_a>, { "+", <term_a> }*. <term_a> = <term_m>, { ("*" | "%"), <term_m> }*. <term_m> = <value> | "+" <term_m> | "-" <term_m> | "(", <expression>, ")".
2	<expression> = <term_a>, { "-", <term_a> }*. <term_a> = <term_m>, { ("*" | "%"), <term_m> }*. <term_m> = <value> | "+" <term_m> | "-" <term_m> | "(", <expression>, ")".
3	<expression> = <term_a>, { "+"), <term_a> }*. <term_a> = <term_m>, { ("/" | "%"), <term_m> }*. <term_m> = <value> | "+" <term_m> | "-" <term_m> | "(", <expression>, ")".
4	<expression> = <term_a>, { "-", <term_a> }*. <term_a> = <term_m>, { ("/" | "%"), <term_m> }*. <term_m> = <value> | "+" <term_m> | "-" <term_m> | "(", <expression>, ")".
5	<expression> = <term_a>, { "+", <term_a> }*. <term_a> = <term_m>, { ("*" | "&"), <term_m> }*. <term_m> = <value> | "+" <term_m> | "-" <term_m> | "(", <expression>, ")".
6	<expression> = <term_a>, { "-" , <term_a> }*. <term_a> = <term_m>, { ("*" | "&"), <term_m> }*. <term_m> = <value> | "+" <term_m> | "-" <term_m> | "(", <expression>, ")".
7	<expression> = <term_a>, { "+", <term_a> }*. <term_a> = <term_m>, { ("/" | "&"), <term_m> }*. <term_m> = <value> | "+" <term_m> | "-" <term_m> | "(", <expression>, ")".
8	<expression> = <term_a>, { "-", <term_a> }*. <term_a> = <term_m>, { ("/" | "&"), <term_m> }*. <term_m> = <value> | "+" <term_m> | "-" <term_m> | "(", <expression>, ")".
9	<expression> = <term_a>, { "+", <term_a> }*. <term_a> = <term_m>, { ("*" | "/"), <term_m> }*. <term_m> = <value> | "+" <term_m> | "-" <term_m> | "(", <expression>, ")".
10	<expression> = <term_a>, { "-", <term_a> }*. <term_a> = <term_m>, { ("*" | "/"), <term_m> }*. <term_m> = <value> | "+" <term_m> | "-" <term_m> | "(", <expression>, ")".

Приклад коду

Наведена програма *(лістинг 2)* перевіряє коректність виразу, синтаксис якого заданий РБНФ *(лістинг 1)*:

Лістинг 1

```
<expression> = <term_a>, { ("+" | "-"), <term_a> }*.
<term_a>     = <term_m>, { ("*" | "/"), <term_m> }*.
<term_m>  = <value> | "+" <term_m> | "-" <term_m> | "(", <expression>, ")".
```

Лістинг 2

```
#include <iostream>
#include <string>
#include <boost/spirit/include/qi.hpp>

namespace qi = boost::spirit::qi;

template <typename Iterator>
struct calculator_simple : qi::grammar<Iterator>{
	calculator_simple() : calculator_simple::base_type(expression){
		expression = term   >> *( '+' >> term   | '-' >> term );
		term       = factor >> *( '*' >> factor | '/' >> factor );

		factor     =
			qi::uint_
			|  '(' >> expression >> ')'
			| '+' >> factor
			| '-' >> factor;
	}

	qi::rule<Iterator> expression, term, factor;
};


int main(){
	std::cout << "Welcome to the expression parser!\n\n";
	std::cout << "Type an expression or [q or Q] to quit\n\n";

	typedef std::string     str_t;
	typedef str_t::iterator str_t_it;

	str_t expression;

	calculator_simple<str_t_it> calc;

	while(true){
		std::getline(std::cin, expression);
		if(expression == "q" || expression == "Q") break;
		str_t_it begin = expression.begin(), end = expression.end();

		bool success = qi::parse(begin, end, calc);

		std::cout << "---------------------\n";
		if(success && begin == end)
			std::cout << "Parsing succeeded\n";
		else
			std::cout << "Parsing failed\nstopped at: \""
			<< str_t(begin, end) << "\"\n";
		std::cout << "---------------------\n";
	}
}
```

Нотація Бекуса-Наура

Нота́ція Бе́куса—Нау́ра (англ. Backus-Naur form, BNF) — це спосіб запису правил контекстно-вільної граматики, тобто форма опису формальної мови.

БНФ визначає скінченну кількість символів (нетерміналів). Крім того, вона визначає правила заміни символу на якусь послідовність букв (терміналів) і символів. Процес отримання ланцюжка букв можна визначити поетапно: спочатку є один символ (символи зазвичай знаходяться у кутових дужках, а їх назва не несе жодної інформації). Потім цей символ замінюється на деяку послідовність букв і символів, відповідно до одного з правил. Потім процес повторюється (на кожному кроці один із символів замінюється на послідовність, згідно з правилом). Зрештою , виходить ланцюжок , що складається з букв і не містить символів. Це означає , що отриманий ланцюжок може бути виведений з початкового символу.

Нотація БНФ є набором «продукцій», кожна з яких відповідає зразку:

<символ> ::= <вираз, що містить символи>

<вираз, що містить символи> це послідовність символів або послідовності символів, розділених вертикальною рискою |, що повністю перелічують можливий вибір символ з лівої частини формули.

Наступні чотири символи є символами мета-мови, вони не визначені у мові, котру описують:

< — лівий обмежувач виразу

> — правий обмежувач виразу

::= — визначене як

| — або

Інші символи належать до «абетки» описуваної мови.

Приклад 1. БНФ для поштової адреси:

Лістинг 3

```
<поштова-адреса> ::= <поштове-відділення> <вулична-адреса> <особа>
<поштове-відділення> ::= <індекс> ", " <місце> <EOL>
<місце> ::= <село> | <місто>
<вулична-адреса> ::= <вулиця> "," <будинок> <EOL>
<особа> ::= <прізвище> <ім'я> <EOL> | <прізвище> <ім'я> <по батькові> <EOL>
```

Приклад 2. Один зі способів означення натуральних чисел за допомогою БНФ:

Лістинг 4

```
<нуль> ::= 0
<ненульова цифра> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<цифра> ::= <нуль> | <ненульова цифра>
<послідовність цифр> ::= <нуль> | <ненульова цифра> | <цифра><послідовність цифр>
<натуральне число> ::= <цифра> | <ненульова цифра><послідовність цифр>
```

Розширена нотація Бекуса-Наура

Розши́рена фо́рма Бе́куса — На́ура (англ. extended Backus–Naur form, EBNF) була розроблена Ніклавсом Віртом, яка сьогодні існує в багатьох варіантах, перед усім — ISO-14977. РБНФ відрізняється від БНФ більш «ємкими» конструкціями, що дозволяють при тій же виразності спростити і скоротити в обсязі опис.

При використанні розширеної форми Бекуса-Наура (EBNF):

- не термінальні символи записуються як окремі слова
- термінальні символи записуються в лапках
- вертикальна риска |, як і в БНФ, використовується для визначення альтернатив
- круглі дужки використовуються для групування
- квадратні дужки використовуються для визначення можливого входження символу або групи символів
- фігурні дужки використовуються для визначення можливого повторення символу або групи символів
- символ рівності використовується замість символу :: =
- конкатенація позначається комою
- символ крапки використовується для позначення кінця правила
- коментарі записуються між символами (* ... *)

Приклад 4. Один зі способів означення цілих чисел за допомогою РБНФ:

Лістинг 5

```
Integer = Sign, UnsignedInteger.
UnsignedInteger = digit, {digit}.
Sign = [ "+" | "-"].
digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9".
```