Белорусский государственный университет информатики и радиоэлектроники

Факультет информационных технологий и управления

Кафедра интеллектуальных информационных технологий

Логические основы интеллектуальных систем

Отчёт по лабораторной работе №1

Студенты:	Макаренко А. И.
	Корсакова С. К.
Группа:	121701
Проверил:	Ивашенко В. П.

Минск 2023

Тема: Представление и синтаксическая обработка формул сокращённого языка логики высказываний.

Цель: Приобрести навыки программирования алгоритмов синтаксического разбора формул сокращенного языка логики высказываний.

Вариант (Н): Определить количество подформул заданного уровня в формуле сокращенного языка логики высказываний.

Теоретические сведения:

Лексическая грамматика сокращённого языка логики высказываний:

<логическая константа> ::= 1|0
<латинская заглавная буква>::=A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z
<отрицание> ::= !
<конъюкция> ::= v
<дизъюнкция> ::= &
<импликация> ::= ->
<эквиваленция> ::= ~
<открывающаяся скобка> ::= (
<закрывающаяся скобка> ::=)
<бинарная связка> ::= <эквиваленция>|<конъюкция>|<дизъюнкция>|<импликация>

Грамматика сокращённого языка логики высказываний:

Ф – формула
<атомарная Ф> ::= <латинская заглавная буква>
<унарная сложная Ф>::=<откр. скобка><отрицание><Ф><закрыв. скобка>
<бинарная сложная Ф> ::= <откр. скобка><Ф><бин. связка><Ф><закрыв. скобка>
<сложная Ф> ::= <унарная сложная ф-ла>|<бинарная сложная ф-ла>
<Ф> ::= <логическая константа>|<атомарная Ф>| <сложная Ф>

Подформула – подстрока формулы, которая тоже является формулой.

Принцип работы программы:

1. Пользователь вводит корректную формулу сокращённого языка логики высказываний. Формула сохраняется в виде строки.

2. Создаётся вспомогательный вектор *helping*, заполненный нулями; длина вектора равна длине строки формулы.

3. Далее функция *added-zero-level* находит все атомарные подформулы из строки формулы и сохраняет их индексы в строке формулы в отдельном векторе *data*. Вместе с этим сохраняет в *data* и уровень атомарных формул, а именно уровень 0.

4. Функция *get-number-of-brackets* подсчитывает количество открывающихся и закрывающихся скобок в строке формулы, чтобы определить необходимое количество итераций для поиска всех подформул. Данное количество сохраняется в переменную *number-of-brackets*.

5. Вызывается функция *fulfill-data*, в которой с помощью цикла while происходит поиск подформул. Цикл повторяется *number-of-brackets*-раз

   (a) В цикле происходит поиск такого символа строки формулы, чтобы он был равен открывающейся скобке и в соответсвующей ячейке вспомогательного вектора *helping* был равен 0 (означает, что скобка еще не была найдена). Если такой символ найден, то во вспомогательном векторе *helping* ячейка с индексом символа помечается как 1.

   (b) Далее происходит поиск такого символа строки формулы, чтобы он был равен закрывающейся скобке и во вспомогательном векторе *helping* был равен 0 (означает, что скобка еще не была найдена). Если такой символ найден, то во вспомогательном векторе *helping* ячейка с индексом символа помечается как 1.

   (c) Происходит проверка на наличие строки подформулы в векторе *data*, чтобы предотвратить ненужное дублирование одинаковых подформул.

   (d) Вызывается функция *define-level*, которая определяет уровень строки подформулы.

      i. Если подформула еще не была найдена, то индексы открывающейся и закрывающейся скобок сохраняются в *data* вместе с уровнем строки подформулы.

6. Пользователь вводит необходимый уровень подформул.

7. Вызывается функция *get-number-of-subformulas*. Её результат и есть количество подформул с заданным уровнем.

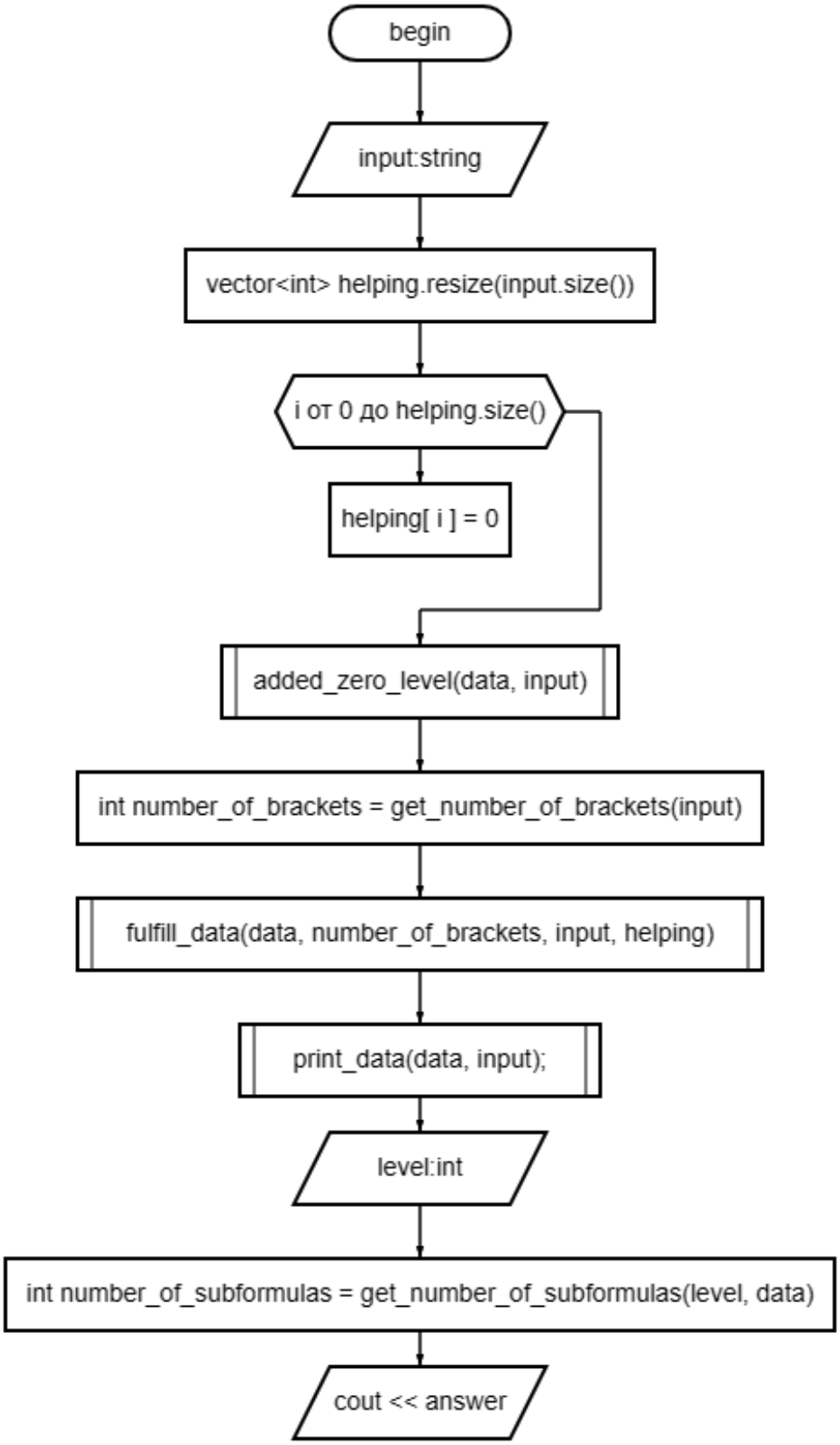
8. Вывод ответа в консоль.

9. Конец программы.

Пример выполнения программы:

```
Input the formula
((((A&B)~(!C))&((C&D)~(A&B)))&((F&A)~(!C)))
Level: 0   A
Level: 0   B
Level: 0   C
Level: 0   D
Level: 0   F
Level: 1   (A&B)
Level: 1   (!C)
Level: 1   (C&D)
Level: 1   (F&A)
Level: 2   ((A&B)~(!C))
Level: 2   ((C&D)~(A&B))
Level: 2   ((F&A)~(!C))
Level: 3   (((A&B)~(!C))&((C&D)~(A&B)))
Level: 4   ((((A&B)~(!C))&((C&D)~(A&B)))&((F&A)~(!C)))
Input the nessesary level of subformula:
2
Number of subformulas with the 2 level is 3
```

Вывод: Выполнение задачи требовало понимания грамматики сокращенного языка логики высказываний и подформулы формулы. Разработанный алгоритм позволяет отыскать все подформулы формулы сокращённого языка логики высказываний и определить их уровень.


begin
input:string
vector<int> helping.resize(input.size())
i от 0 до helping.size()
helping[i] = 0
added_zero_level(data, input)
int number_of_brackets = get_number_of_brackets(input)
fulfill_data(data, number_of_brackets, input, helping)
print_data(data, input);
level:int
int number_of_subformulas = get_number_of_subformulas(level, data)
cout << answer

Блок-схема