

Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра интеллектуальных информационных технологий

**Отчет по лабораторной работе №1
по курсу «ЯПИС»**

на тему: «Языковые процессоры интеллектуальных систем и
интеллектуализация CASE-технологий»

Выполнил студент группы 921703:

Кравцов Михаил Сергеевич

Проверил:

Баснина Анастасия Павловна

МИНСК 2022

Содержание:

1. Вариант задания
2. Примеры на придуманном языке
3. Файл грамматики
4. Описание дополнительно разработанных классов
5. Перечень генерируемых ошибок
6. Пример генерации промежуточного кода
7. Примеры работы компилятора

Вариант задания 19:

1. Целевой код - исполняемый файл (.exe), формат промежуточного кода – язык.

C++, генерация целевого кода стандартным компилятором.

2. Язык - язык для работы с множествами. Встроенные типы: element, set.

Операции: переопределить +, -, *, \ и т.д. для встроенных типов.

3. Объявление переменных – неявное.
4. Преобразование типов – явное, например, a = (int) b.
5. Оператор присваивания – Одноцелевой, например, a = b.
6. Структуры, ограничивающие область видимости – подпрограммы.
7. Маркер блочного оператора - неявные, например как в python.
8. Условные операторы - двух вариантный оператор if-then-else.
9. Перегрузка подпрограмм – отсутствует.
10. Передача параметров в подпрограмму - только по значению и возвращаемому значению.
11. Допустимое место объявления подпрограмм - в начале программы.

Примеры на придуманном языке:

1. Первый пример демонстрирует работу с функциями и операции над множествами, а также работу с циклами и вывод в консоль.

```
def void showTwoSets(set A, set B
    for(elementA : A
        print(elementA);
        print(" "):
    print("\n");
    for(elementB : B
        print(elementB);
        print(" "):
    print("\n-----\n");
    return;

def void showOneSet(set someSet
    for(el : someSet
        print(el);
        print(" "):
```

```

print("\n");
return;

def void main ()
    set setA = {1,2,3,4,5};
    set setB = {1,2,3,4,5, 78, 555};
    print("set A and set B -> \n");
    showTwoSets(setA, setB);
    set setC = setA - setB;
    print("difference of A and B -> ");
    showOneSet(setC);

    element elementA = 45;
    element elementB = 99;
    element elementC = 120;

    setC.add(elementA);
    setC.add(elementB);
    setC.add(elementC);
    print("set C -> ");
    showOneSet(setC);

    setC = setC + setA;
    print("C after union with A -> ");
    showOneSet(setC);

    set setD = setC ^ setB;
    print("symmetric difference of C and B -> ");
    showOneSet(setD);
    return;

```

2. Во втором примере видна работа с операторами сравнения и циклами. Также продемонстрирована вложенность операторов друг в друга.

```

def element getNextElementOfSet(set someSet, element previousElement)
    var resultElement = -1;
    var condition = 0;
    for(el : someSet)
        if(el == previousElement)
            condition = 1;
        else
            if(condition > 0)
                resultElement = el;
                condition = 0;
    return resultElement;

def void main()
    element elementA = 5;
    element elementB = 785;
    set setA = {1,2,elementA,7,elementB};
    for(el : setA)
        print(el);
        print(" "):
    print("\n");

```

```

element nonElement = -1;
element afterA = getNextElementOfSet(setA, elementA);
element afterB = getNextElementOfSet(setA, elementB);

while(afterA != nonElement)
    print(elementA);
    print(" next -> ");
    print(afterA);
    print("\n");
    elementA = afterA;
    afterA = getNextElementOfSet(setA, elementA);

if(afterB == nonElement)
    print("done");
else
    print("error");
return;

```

3. В третьем примере продемонстрированы базовые операции с множествами.

```

def void main()
    element elementA = 7;
    set setA = {1, 2, 3, 4, 5, 6, elementA, 8, 9, 10};
    set setB = {};

    for(el: setA)
        if(el <= elementA)
            element sizeOfSetB = size(setB);
            if(sizeOfSetB < 4)
                setB.add(el);
            else
                clear(setB);

    setB.remove(elementA);
    setB.add(110);

    for(el : setB)
        print(el);
        print(" ");
    return;

```

Файл грамматики:

```
grammar PollaM;
```

```

//программа на разработанном языке состоит из объявленных пользователем
функций (от 0 и более), а также обязательной функции MAIN
program: (functionDefReturn | functionDefNonReturn)* mainDef;

```

```

//встроенные типы языка
type: SET
    | ELEMENT;

```

```
typeIdPart: type ID;
```

```

typeVarPart: type | VAR;

//правило объявления функции MAIN (начало программы)
mainDef: DEF VOID MAIN OPEN_BRACKET CLOSE_BRACKET
functionBodyWithoutReturn;

//правило объявления функции с возвращаемым значением
functionDefReturn: DEF typeIdPart OPEN_BRACKET functionDefParameters?
CLOSE_BRACKET functionBodyWithReturn;

//правило объявления функции, которая ничего не возвращает. Отличается от
предыдущей наличием параметров функции и другим телом функции (без return)
functionDefNonReturn: DEF VOID ID OPEN_BRACKET functionDefParameters?
CLOSE_BRACKET functionBodyWithoutReturn;

//тело функции без возвращаемого значения состоит из N-ого количества строк
кода
functionBodyWithoutReturn: ((contentLine SEMICOLON) | operators)* emptRet;
emptRet: RETURN SEMICOLON;

//отличие функции с возвращаемом значением состоит в наличии обязательной
строки RETURN значение
functionBodyWithReturn: ((contentLine SEMICOLON) | operators)* RETURN ID
SEMICOLON;

//способ объявления параметров функции -> тип имя, тип имя, ...
functionDefParameters: (typeIdPart COMMA)* typeIdPart;

//строка кода может быть следующих видов:
// -операции с множествами (set)
// -логические операции
// -операции изменения множества (add, remove, clear, и др)
// -декларация переменных, их присвоение, вызов функции
// -циклы
// -вызов встроенных функций (print, clear, size)
contentLine: operationsWithSets
| booleanOperations
| changeSetOperation
| variableDeclaration
| valueAssignment
| variableDeclarationWithAssignment
| typeConversion
| functionCall
| printCall
| sizeCall
| clearCall;

operators: ifBlock
| forBlock
| whileBlock;

//объявление переменных (также можно объявить неявно через ключевое слово
var)
variableDeclaration: typeVarPart ID;

//объявление переменных вместе с присвоением значения
variableDeclarationWithAssignment: typeVarPart valueAssignment;

```

```

//присвоение значений переменной. Можно присвоить следующие значения:
// -число
// -значение другой переменной
// -возвращаемое из функции значения
valueAssignment: ID ASSIGN (NUMBER | typeConversion | ID | sizeCall |
operationsWithSets | functionCall
                                | (OPEN_CURLY_BRACKET (((NUMBER | ID)
COMMA)* (NUMBER | ID))? CLOSE_CURLY_BRACKET));

//приведение типов
typeConversion: OPEN_BRACKET type CLOSE_BRACKET ID;

//вызов функции
functionCall: ID OPEN_BRACKET ((ID COMMA)* ID)? CLOSE_BRACKET;

//операции с множествами включают в себя:
// -операцию объединения  $A \cup B = \{x \mid (x \in A) \vee (x \in B)\}$ 
// -операцию разности  $A \setminus B = \{x \mid (x \in A) \wedge (x \notin B)\}$ 
// -операцию пересечения  $A \cap B = \{x \mid (x \in A) \wedge (x \in B)\}$ 
// -операцию симметричной разности  $A \Delta B = (A \setminus B) \cup (B \setminus A)$ 
operationsWithSets: unionOperation
                        | differenceOperation
                        | intersectionOperation
                        | symmetricDifferenceOperation;

//операции изменения множества (добавление|удаление element)
changeSetOperation: ID DOT (ADD | REMOVE) OPEN_BRACKET (ID | NUMBER)
CLOSE_BRACKET;

//объединение множества
unionOperation: ID UNION ID;

//разность двух множеств
differenceOperation: ID DIFFERENCE ID;

//пересечение множеств
intersectionOperation: ID INTERSECTION ID;

//симметрическая разность
symmetricDifferenceOperation: ID SYMMETRIC_DIFFERENCE ID;

//стандартные логические операции
booleanOperations: (ID|NUMBER) EQUALS (ID|NUMBER)
                    | (ID|NUMBER) GT_EQ (ID|NUMBER)
                    | (ID|NUMBER) LT_EQ (ID|NUMBER)
                    | (ID|NUMBER) NOT_EQUALS (ID|NUMBER)
                    | (ID|NUMBER) GT (ID|NUMBER)
                    | (ID|NUMBER) LT (ID|NUMBER)
                    | ID;

//объявление if блока. Он состоит из: «if (логическое выражение) {ноль или
более строк кода} else(опционально) {ноль или более строк кода}»
ifBlock: IF OPEN_BRACKET booleanOperations CLOSE_BRACKET
operatorBody
(ELSE operatorBody)?;

```

```

//объявление цикл for, для перебора всех элементов
forBlock: FOR OPEN_BRACKET ID FROM ID CLOSE_BRACKET operatorBody;

//объявление блока while
whileBlock: WHILE OPEN_BRACKET booleanOperations CLOSE_BRACKET
operatorBody;

//тело операторов for, while, if-then-else
operatorBody: ((contentLine SEMICOLON) | operators)* ((contentLine COLON) |
operators);

//вызов функции PRINT
printCall: PRINT OPEN_BRACKET (ID | STRING | NUMBER) CLOSE_BRACKET;

//вызов функции SIZE
sizeCall: SIZE OPEN_BRACKET ID CLOSE_BRACKET;

//вызов функции CLEAR для очистки множества
clearCall: CLEAR OPEN_BRACKET ID CLOSE_BRACKET;

MAIN: 'main';
DEF: 'def';
VOID: 'void';
IF: 'if';
ELSE: 'else';
WHILE: 'while';
FOR: 'for';
RETURN: 'return';
FROM: 'from';

SET: 'set';
ELEMENT: 'element';
VAR: 'var';

PRINT: 'print';

SIZE: 'size';
CLEAR: 'clear';
ADD: 'add';
REMOVE: 'remove';

ID : [a-zA-Z_]+;
STRING: '"'(.)+?'"';
NUMBER: '0' | '-'?[1-9][0-9]*;

COLON: ':';
SEMICOLON: ';';
DOT: '.';
COMMA: ',';
ASSIGN: '=';
UNION: '+';
DIFFERENCE: '-';
INTERSECTION: '&';
SYMMETRIC_DIFFERENCE: '^';
EQUALS: '==';
NOT_EQUALS: '!=';
GT_EQ: '>=';

```

```

LT_EQ: '<=';
GT: '>';
LT: '<';

OPEN_BRACKET : '(';
CLOSE_BRACKET : ')';
OPEN_CURLY_BRACKET : '{';
CLOSE_CURLY_BRACKET : '}';

WS: [ \t\r\n]+ -> skip;

```

Описание дополнительно разработанных классов:

1. CustomVisitor – отвечает за составление пода на промежуточном языке (C++)
2. CustomErrorHandler – отвечает за работу с ошибками во время компиляции программы
3. AntlrCompiler – класс который является главным и отвечает за чтение текстов из файлов, из парсинг и компиляцию.
4. PMTokes и CTokens – классы описывающие токены языков (C++ и PollaM)
5. PollaMParser – парсер
6. VariableContext – помогает при отслеживании ошибок типо двойного объявления переменной.

Генерируемые ошибки:

1. SyntaxError – при неверно написанной программе
2. CompilerError – ошибка компиляции, где указывается неверные выражения.

Пример работы компилятора:

Работа компилятора будет продемонстрирована с использованием первого примера кода.

После запуска получим результат в консоли:

```

01:21:04 INFO - example files -> [example_1, example_2, example_3]
01:21:04 INFO - input from file example_1:

```

```

def void showTwoSets(set A, set B)
    for(elementA from A)
        print(elementA);
        print(" ");
    print("\n");
    for(elementB from B)
        print(elementB);
        print(" ");
    print("\n-----\n");

```



```

return;

def void showOneSet(set someSet)
    for(el from someSet)
        print(el);
        print(" ");
    print("\n");
    return;

```

```

def void main ()
    set setA = {1,2,3,4,5};
    set setB = {1,2,3,4,5, 78, 555};
    print("set A and set B -> \n");
    showTwoSets(setA, setB);
    set setC = setA - setB;
    print("difference of A and B -> ");
    showOneSet(setC);

```

```

element elementA = 45;
element elementB = 99;
element elementC = 120;

```

```

setC.add(elementA);
setC.add(elementB);
setC.add(elementC);
print("set C -> ");
showOneSet(setC);

```

```

setC = setC + setA;
print("C after union with A -> ");
showOneSet(setC);

```

```

set setD = setC ^ setB;
print("symmetric difference of C and B -> ");
showOneSet(setD);
return;

```

01:21:04 INFO - output:

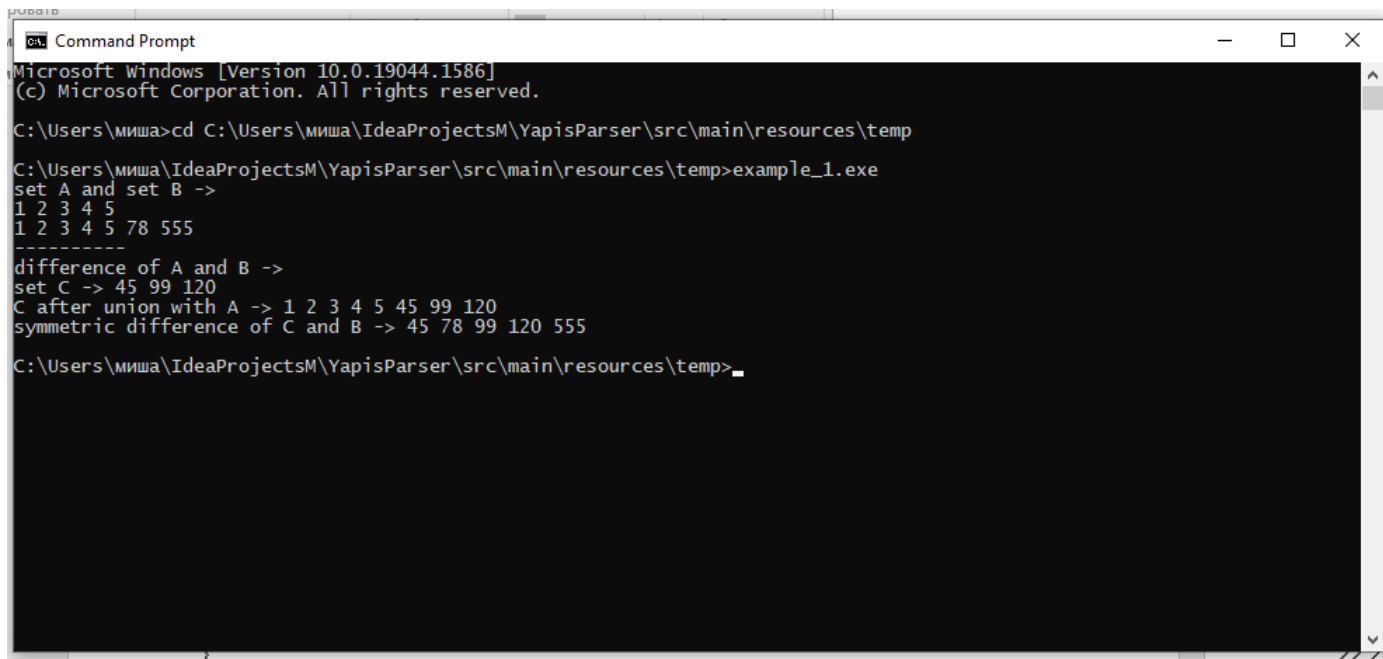
```

#include <iostream>
#include <set>
#include <algorithm>
using namespace std;
void showTwoSets(set<int> A,set<int> B){for(auto&
elementA:A){std::cout<<elementA;std::cout<<" ";}std::cout<<"\n";for(auto&
elementB:B){std::cout<<elementB;std::cout<<" ";}std::cout<<"\n-----\n";}void
showOneSet(set<int> someSet){for(auto& el:someSet){std::cout<<el;std::cout<<"
";}std::cout<<"\n";}int main(){set<int> setA={1,2,3,4,5};set<int>
setB={1,2,3,4,5,78,555};std::cout<<"set A and set B ->
\n";showTwoSets(setA,setB);set<int> setC;set_difference(setA.begin(), setA.end(),
setB.begin(), setB.end() , inserter(setC, setC.begin()));std::cout<<"difference of A

```

```
and B -> ";showOneSet(setC);int elementA=45;int elementB=99;int
elementC=120;setC.insert(elementA);setC.insert(elementB);setC.insert(elementC);st
d::cout<<"set C -> ";showOneSet(setC);setC;set_union(setC.begin(), setC.end(),
setA.begin(), setA.end() , inserter(setC, setC.begin()));std::cout<<"C after union with
A -> ";showOneSet(setC);set<int> setD;set_symmetric_difference(setC.begin(),
setC.end(), setB.begin(), setB.end() , inserter(setD,
setD.begin()));std::cout<<"symmetric difference of C and B -> ";showOneSet(setD);}
01:21:05 INFO - example_1 file was compiled
```

После запуска файла *example 1.exe* в консоли увидим результат:



```
Command Prompt
Microsoft Windows [Version 10.0.19044.1586]
(c) Microsoft Corporation. All rights reserved.

C:\Users\миша>cd C:\Users\миша\IdeaProjectsM\YapisParser\src\main\resources\temp

C:\Users\миша\IdeaProjectsM\YapisParser\src\main\resources\temp>example_1.exe
set A and set B ->
1 2 3 4 5
1 2 3 4 5 78 555
-----
difference of A and B ->
set C -> 45 99 120
C after union with A -> 1 2 3 4 5 45 99 120
symmetric difference of C and B -> 45 78 99 120 555

C:\Users\миша\IdeaProjectsM\YapisParser\src\main\resources\temp>_
```