НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

# Кафедра системного програмування та спеціалізованих комп’ютерних систем

**Лабораторна робота №2**

з дисципліни

«Основи проектування трансляторів»

Тема: **«Розробка генератора коду»**

Виконав: студент 3 курсу ФПМ

групи КВ-02

Костюков С. В.

Перевірив: Коваленко О. П.

Київ – 2023

**Постановка задачі**

Розробити програму генератора коду (ГК) для підмножини мови програмування SIGNAL.

Програма має забезпечувати наступні дії:

* читання дерева розбору та таблиць, створених синтаксичним аналізатором, що було розроблено в розрахунково-графічній роботі;
* виявлення семантичних помилок;
* генерацію коду та/або побудову внутрішніх таблиць для генерації коду;
* формування лістингу вхідної програми з повідомленнями про лексичні, синтаксичні та семантичні помилки.

***Варіант 11***

**<signal-program> --> <program>  
<program> --> PROGRAM <procedure-identifier> ; <block> .  
<block> --> <declarations> BEGIN <statements-list> END  
<declarations> --> <label-declarations>  
<label-declarations> --> LABEL <unsigned-integer> <labels-list>; |  
 <empty>  
<labels-list> --> , <unsigned-integer> <labels-list> |  
 <empty>  
<statements-list> --> <statement> <statements-list> |  
 <empty>  
<statement> --> <unsigned-integer> : <statement> |  
 GOTO <unsigned-integer> ; |  
 LINK <variable-identifier> , <unsigned-integer> ; |  
 IN <unsigned-integer> ; |  
 OUT <unsigned-integer> ;  
<variable-identifier> --> <identifier>  
<procedure-identifier> --> <identifier>  
<identifier> --> <letter><string>  
<string> --> <letter><string> |  
 <digit><string> |  
 <empty>  
<unsigned-integer> --> <digit><digits-string>  
<digits-string> --> <digit><digit-string> |  
 <empty>  
<digit> --> 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9  
<letter> --> A | B | C | D | ... | Z**

**Лістинг коду програми ГК:**

***Main.cpp***

#include "lexical\_analyzer.h"

#include "syntactic\_analyzer.h"

#include "data.h"

/\*

\* Variant 11

<signal-program> --> <program>

<program> --> PROGRAM <procedure-identifier> ; <block> .

<block> --> <declarations> BEGIN <statements-list> END

<declarations> --> <label-declarations>

<label-declarations> --> LABEL <unsigned-integer> <labels-list>; |

<empty>

<labels-list> --> , <unsigned-integer> <labels-list> |

<empty>

<statements-list> --> <statement> <statements-list> |

<empty>

<statement> --> <unsigned-integer> : <statement> |

GOTO <unsigned-integer> ; |

LINK <variable-identifier> , <unsigned-integer> ; |

IN <unsigned-integer> ; |

OUT <unsigned-integer> ;

<variable-identifier> --> <identifier>

<procedure-identifier> --> <identifier>

<identifier> --> <letter><string>

<string> --> <letter><string> |

<digit><string> |

<empty>

<unsigned-integer> --> <digit><digits-string>

<digits-string> --> <digit><digit-string> |

<empty>

<digit> --> 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<letter> --> A | B | C | D | ... | Z

\*/

/\*

',', ':', ';', '.'

44 58 59 46

\*/

**int** main() {

Data data;

data.root = **new** Tree();

data.curr = **new** Tree();

//if (!file\_parsing("FalseTest1.txt", data)) return 0; // No used label  
 //if(!file\_parsing("FalseTest2.txt", data)) return 0; // Overused label  
 //if(!file\_parsing("FalseTest3.txt", data)) return 0; // Value don`t link  
 if(!file\_parsing("FalseTest4.txt", data)) return 0; // Value in other state

if (!signal\_program(data)) return 0;  
  
 code\_generator(data);

**return** 0;

}

***generator.h***

#ifndef MAIN\_CPP\_GENERATOR\_H

#define MAIN\_CPP\_GENERATOR\_H

#include "data.h"

**void** code\_generator(Data &);

#endif //MAIN\_CPP\_GENERATOR\_H

***generator.cpp***

#include "generator.h"

**enum** **class** Generator\_Errors {

ERROR\_NO\_USED\_LABEL, // мітка не використовувалась перед операторами

ERROR\_OVER\_USED\_LABEL, // мітка використовувалась перед операторами більше одного разу

ERROR\_VALUE\_DONT\_LINK, // порт не був LINK-ований

ERROR\_VALUE\_IN\_OTHER\_STATE, // порт вже використовується для IN/OUT

};

**void** Error\_message(Generator\_Errors, **const** std::string &);

**bool** iterations(Tree &curr, std::string &result, Data &data) {

**if** (curr.code == 301) { // PROGRAM

result += ";" + data.program\_identifier + "\n\ndata SEGMENT\n";

data.curr\_keyword = 301;

} **else** **if** (curr.code == 302) { // BEGIN

result += "\ndata ENDS \n\ncode SEGMENT\nbegin:\n";

data.curr\_keyword = 302;

} **else** **if** (curr.code == 303) { // END

result += "\n\tret 0\ncode ENDS\n\tend begin\n";

} **else** **if** (curr.code == 304) { // GOTO

data.curr\_keyword = 304;

} **else** **if** (curr.code == 305) { // LINK

data.curr\_keyword = 305;

} **else** **if** (curr.code == 306) { // IN

data.curr\_keyword = 306;

} **else** **if** (curr.code == 307) { // OUT

data.curr\_keyword = 307;

}

**if** (curr.parent && curr.parent->name == "<unsigned-integer>") {

**if** (data.curr\_keyword == 304) { // GOTO

**if** (data.labels[curr.name].count\_uses == 1) {

result += "\tJMP " + curr.name + "\n";

} **else** **if** (data.labels[curr.name].count\_uses == 0) {

Error\_message(Generator\_Errors::ERROR\_NO\_USED\_LABEL, curr.name);

**return** **false**;

} **else** {

Error\_message(Generator\_Errors::ERROR\_OVER\_USED\_LABEL, curr.name);

**return** **false**;

}

data.curr\_keyword = 0;

} **else** **if** (data.curr\_keyword == 305) { // LINK

**if** (data.labels[curr.name].type == Label\_type::LABEL\_POINT) {

Error\_message(Generator\_Errors::ERROR\_VALUE\_IN\_OTHER\_STATE, curr.name + "(it is Point label)");

**return** **false**;

} **else** {

data.labels[curr.name].is\_link = **true**;

}

data.curr\_keyword = 0;

} **else** **if** (data.curr\_keyword == 306) { // IN

**if** (!data.labels[curr.name].is\_link) {

Error\_message(Generator\_Errors::ERROR\_VALUE\_DONT\_LINK, curr.name + " 1");

**return** **false**;

} **else** **if** (data.labels[curr.name].type == Label\_type::LABEL\_OUT) {

Error\_message(Generator\_Errors::ERROR\_VALUE\_IN\_OTHER\_STATE, curr.name + "(used in OUT state)");

**return** **false**;

} **else** {

data.labels[curr.name].type = Label\_type::LABEL\_IN;

}

data.curr\_keyword = 0;

} **else** **if** (data.curr\_keyword == 307) { // OUT

**if** (!data.labels[curr.name].is\_link) {

Error\_message(Generator\_Errors::ERROR\_VALUE\_DONT\_LINK, curr.name + " 2");

**return** **false**;

} **else** **if** (data.labels[curr.name].type == Label\_type::LABEL\_IN) {

Error\_message(Generator\_Errors::ERROR\_VALUE\_IN\_OTHER\_STATE, curr.name + "(used in IN state)");

**return** **false**;

} **else** {

data.labels[curr.name].type = Label\_type::LABEL\_OUT;

}

data.curr\_keyword = 0;

} **else** **if** (data.labels[curr.name].type == Label\_type::LABEL\_POINT && data.curr\_keyword != 301) {

result += "\t" + curr.name + ":\n";

} **else** **if** (data.curr\_keyword == 302) {

data.curr\_keyword = 0;

}

}

**for** (**auto** &i: curr.child) {

**if** (!iterations(\*i, result, data)) **return** **false**;

}

**return** **true**;

};

**void** code\_generator(Data &data) {

std::string result;

data.curr\_keyword = 0;

**if** (iterations(\*data.root, result, data)) {

std::cout << result << std::endl;

}

**if** (!data.warning.empty()) {

std::cout << "\n" << data.warning << std::endl;

}

}

**void** Error\_message(Generator\_Errors error, **const** std::string &description) {

std::cout << "Generator: Error: ";

**if** (error == Generator\_Errors::ERROR\_NO\_USED\_LABEL) {

std::cout << "No used label " << description << std::endl;

} **else** **if** (error == Generator\_Errors::ERROR\_OVER\_USED\_LABEL) {

std::cout << "Over used label " << description << std::endl;

} **else** **if** (error == Generator\_Errors::ERROR\_VALUE\_DONT\_LINK) {

std::cout << "Value don`t link " << description << std::endl;

} **else** **if** (error == Generator\_Errors::ERROR\_VALUE\_IN\_OTHER\_STATE) {

std::cout << "Value in other state " << description << std::endl;

} **else** {

std::cout << "Other Error" << std::endl;

}

}

**Тестування програми:**

*True-тестування:*

***Текст програми:***

PROGRAM PROG;  
LABEL 12, 13, 14, 15, 12;  
  
BEGIN  
12 : GOTO 13 ;  
 LINK VER1, 14 ;  
13 : IN 14 ;  
 LINK VER2, 15 ;  
 OUT 15 ;  
END.

***Отриманий результат:***

;PROG

data SEGMENT

data ENDS

code SEGMENT

begin:

12:

JMP 13

13:

ret 0

code ENDS

end begin

WARNING: Multiply declared of label 12

*False-тестування:*

1. ***Текст програми:***

PROGRAM PROG;  
LABEL 12, 13, 14, 15;  
  
BEGIN  
12 : GOTO 13 ;  
 LINK VER1, 14 ;  
12 : IN 14 ;  
 LINK VER2, 15 ;  
 OUT 15 ;  
END.

***Отриманий результат:***

Generator: Error: No used label 13

1. ***Текст програми:***

PROGRAM PROG;  
LABEL 12, 13, 14, 15;  
  
BEGIN  
13 : GOTO 13 ;  
 LINK VER1, 14 ;  
13 : IN 14 ;  
 LINK VER2, 15 ;  
 OUT 15 ;  
END.

***Отриманий результат:***

Generator: Error: Overused label 13

1. ***Текст програми:***

PROGRAM PROG;  
LABEL 12, 13, 14, 15;  
  
BEGIN  
12 : GOTO 13 ;  
 LINK VER1, 14 ;  
13 : IN 14 ;  
 LINK VER2, 14 ;  
 OUT 15 ;  
END.

***Отриманий результат:***

Generator: Error: Value don`t link 15

1. ***Текст програми:***

PROGRAM PROG;  
LABEL 12, 13, 14, 15;  
  
BEGIN  
12 : GOTO 13 ;  
 LINK VER1, 14 ;  
13 : IN 14 ;  
 LINK VER2, 15 ;  
 OUT 14 ;  
END.

***Отриманий результат:***

Generator: Error: Value in other state 14(used in IN state)