

Студент: Моисеенков И. П.
Группа: М8О-208Б-19
Номер по списку: 21

«СИСТЕМЫ ПРОГРАММИРОВАНИЯ»
Курсовая работа 2021.
Часть 1.

Для заданного в Лабораторной №8 диалекта языка МИКРОЛИСП разработайте синтаксически управляемый транслятор на язык C++, применяя методику из Лабораторной №9, Правила TranslationRules21.rtf и TextLayout.txt .

Работоспособность транслятора проверьте на трех контрольных задачах из Лабораторной №8.

Шаблон файла code-gen.cpp создайте с помощью приложения Make-code-gen.cpp .

Вариант грамматики: j21

Контрольная задача №1 – zeller.

Полный скриншот трансляции без трассировки.

```
Source>zeller
Source:zeller.ss
1|;zeller.ss
2|(define (day-of-week)
3| (zeller dd
4|      (cond((< mm 3)(+ mm 10))(#t(- mm 2)))
5|      (remainder (cond((< mm 3)(- yyyy 1))(#t yyyy))100)
6|      (quotient (cond((< mm 3)(- yyyy 1))(#t yyyy))100)
7| )
8|)
9|(define (zeller d m y c)
10| (neg-to-pos (remainder (+ d y
11|                        (quotient (-(* 26 m)2) 10)
12|                        (quotient y 4)
13|                        (quotient c 4)
14|                        (* 2(- c))
15|                        )
16|      7)
17| )
18|)
```

```

19| (define (neg-to-pos d)
20|   (cond((< d 0)(+ d 7))
21|         (#t d)
22|   )
23| )
24| (define (birthday dw)
25|   ^{0,...,6};
26|   (display "Your were born on ")
27|   (display
28|     (if(= dw 1)"Monday "
29|         (if (= dw 2)"Tuesday "
30|             (if (= dw 3)"Wednesday "
31|                 (if (= dw 4)"Thursday "
32|                     (if (= dw 5)"Friday "
33|                         (if (= dw 6)"Saturday "
34|                             "Sunday" ))))))))
35|   (display dd)(display ".")
36|   (display mm)(display ".")
37|   yyyy
38| )
39| (define dd 31)
40| (define mm 03)
41| (define yyyy 2001)
42| (birthday (day-of-week))
43|

```

Code:

```

/* MIP */
#include "mlisp.h"
double day__of__week/*2*/ ( ) ;
double zeller/*9*/ (double d, double m
, double y, double c) ;
double neg__to__pos/*19*/ (double d) ;
double birthday/*24*/ (double dw) ;
extern double dd/*39*/;
extern double mm/*40*/;
extern double yyyy/*41*/;
//
double day__of__week/*2*/ ( ) {
return
zeller(dd
, ((mm < 3.)
? (mm + 10.)
: true
? (mm - 2.)
: _infinity)
, remainder(((mm < 3.)
? (yyyy - 1.)
: true
? yyyy
: _infinity)
, 100.)

, quotient(((mm < 3.)
? (yyyy - 1.)
: true
? yyyy
: _infinity)
, 100.)

```

```

    )
    ;
}
double zeller/*9*/ (double d, double m
    , double y, double c) {
    return
    neg__to__pos(remainder((d + y + quotient(((26. * m) - 2.)
        , 10.)
        + quotient(y
        , 4.)
        + quotient(c
        , 4.)
        + (2. * (- c)))
        , 7.)
    );
}
double neg__to__pos/*19*/ (double d) {
    return
    ((d < 0.)
        ? (d + 7.)
        : true
        ? d
        : _infinity);
}
double birthday/*24*/ (double dw) {
    display("Your were born on ");
    display(((dw == 1.)
        ? "Monday "
        : ((dw == 2.)
        ? "Tuesday "
        : ((dw == 3.)
        ? "Wednesday "
        : ((dw == 4.)
        ? "Thursday "
        : ((dw == 5.)
        ? "Friday "
        : ((dw == 6.)
        ? "Saturday "
        : "Sunday"))))))));
    display(dd);
    display(".");
    display(mm);
    display(".");
    return
    yyyy;
}
double dd/*39*/ = 31.;
double mm/*40*/ = 03.;
double yyyy/*41*/ = 2001.;
int main(){
    display(birthday(day__of__week()));
    newline();
    std::cin.get();
    return
    0;
}

```

Code is saved to file zeller.cpp !

Распечатка файла zeller.cpp .

```
/* MIP */
#include "mlisp.h"
double day__of__week/*2*/ ( ) ;
    double zeller/*9*/ (double d, double m
    , double y, double c) ;
    double neg__to__pos/*19*/ (double d) ;
    double birthday/*24*/ (double dw) ;
    extern double dd/*39*/;
    extern double mm/*40*/;
    extern double yyyy/*41*/;
    //_____
double day__of__week/*2*/ ( ) {
    return
    zeller(dd
        , ((mm < 3.)
        ? (mm + 10.)
        : true
        ? (mm - 2.)
        : _infinity)
        , remainder(((mm < 3.)
        ? (yyyy - 1.)
        : true
        ? yyyy
        : _infinity)
        , 100.)

        , quotient(((mm < 3.)
        ? (yyyy - 1.)
        : true
        ? yyyy
        : _infinity)
        , 100.)
        )
    ;
}
double zeller/*9*/ (double d, double m
    , double y, double c) {
    return
    neg__to__pos(remainder((d + y + quotient(((26. * m) - 2.)
        , 10.)
        + quotient(y
```

```

    , 4.)
    + quotient(c
    , 4.)
    + (2. * (- c)))
    , 7.)
    );
}
double neg__to__pos/*19*/ (double d) {
return
((d < 0.)
? (d + 7.)
: true
? d
: _infinity);
}
double birthday/*24*/ (double dw) {
display("Your were born on ");
display(((dw == 1.)
? "Monday "
: ((dw == 2.)
? "Tuesday "
: ((dw == 3.)
? "Wednesday "
: ((dw == 4.)
? "Thursday "
: ((dw == 5.)
? "Friday "
: ((dw == 6.)
? "Saturday "
: "Sunday"))))));
display(dd);
display(".");
display(mm);
display(".");
return
yyyy;
}
double dd/*39*/ = 31.;
double mm/*40*/ = 03.;
double yyyy/*41*/ = 2001.;
int main(){
display(birthday(day__of__week()));

```

```

newline();
std::cin.get();
return
0;
}

```

Скриншот запуска задачи на C++.

```

mosik@LAPTOP-69S778GL:~/sp_cw1$ ./zeller
Your were born on Saturday 31.3.2001

```

Контрольная задача №2 – golden21.

Полный скриншот трансляции без трассировки.

```

Source>golden21
Source:golden21.ss
1|;golden21
2|(define a 2)(define b 6)
3|(define (fun x)
4|  (set! x (- x (/ 21 22)))
5|  (- (expt(- x 3) 4) (expt(atan x) 3) 2)
6|)
7|(define (golden-section-search a b)
8|  (let(
9|    (xmin(cond((< a b)(golden-start a b))(#t(golden-start b a ))))
10|  )
11|    (newline)
12|    xmin
13|  )
14|)
15|(define (golden-start a b)
16|  (set! total-iterations 0)
17|  (let(
18|    (xa (+ a (* mphi(- b a))))
19|    (xb (+ b (-(* mphi(- b a)))))
20|  )
21|    (try a b xa (fun xa) xb (fun xb))
22|  )
23|)
24|(define mphi (* (- 3(sqrt 5))(/ 2e+0)))
25|(define (try a b xa ya xb yb)
26|  (cond((close-enough? a b)
27|    (* (+ a b)5e-1))
28|    (#t(display "+")
29|      (set! total-iterations (+ total-iterations 1))
30|      (cond((< ya yb)(set! b xb)
31|        (set! xb xa)
32|        (set! yb ya)
33|        (set! xa (+ a (* mphi(- b a))))
34|        (try a b xa (fun xa) xb yb)
35|      )

```

```

36|          (#t (set! a xa)
37|             (set! xa xb)
38|             (set! ya yb)
39|             (set! xb (- b (* mphi(- b a))))
40|             (try a b xa ya xb (fun xb))
41|             )
42|          );cond...
43|      )
44|  )
45|)
46|(define (close-enough? x y)
47|  (<(abs (- x y))tolerance))
48|(define tolerance 1e-3)
49|(define total-iterations 0)
50|(define xmin 0)
51|(set! xmin(golden-section-search a b))
52|  (display"Interval=\t[")
53|  (display a)
54|  (display" , ")
55|  (display b)
56|  (display"]\n")
57|  (display"Total number of iteranions=")
58|total-iterations
59|  (display"xmin=\t\t")
60|xmin
61|  (display"f(xmin)=\t")
62|(fun xmin)
63|

```

Code:

```

/* MIP */
#include "mlisp.h"
extern double a/*2*/;
extern double b/*2*/;
double fun/*3*/ (double x) ;
double golden__section__search/*7*/ (double a, double b) ;
double golden__start/*15*/ (double a, double b) ;
extern double mphi/*24*/;
double __MIP__try/*25*/ (double a, double b
, double xa, double ya
, double xb, double yb) ;
bool close__enough_Q/*46*/ (double x, double y);
extern double tolerance/*48*/;
extern double total__iterations/*49*/;
extern double xmin/*50*/;
//
double a/*2*/ = 2.;
double b/*2*/ = 6.;
double fun/*3*/ (double x) {
x = (x - (21. / 22.));
return
(expt((x - 3.)
, 4.)
- expt(atan(x)
, 3.)
- 2.);
}

```

```
double golden__section__search/*7*/ (double a, double b) {
{
double xmin( ((a < b)
? golden__start(a
, b)

: true
? golden__start(b
, a)

: _infinity) );
newline();
return xmin;
}
}
```

```
double golden__start/*15*/ (double a, double b) {
total__iterations = 0.;
{
double xa( (a + (mphi * (b - a))) ),
xb( (b + (- (mphi * (b - a)))) );
return __MIP__try(a
, b
, xa
, fun(xa)
, xb
, fun(xb))
;
}
}
```

```
double mphi/*24*/ = ((3. - sqrt(5.)) * (1. / 2e+0));
double __MIP__try/*25*/ (double a, double b
, double xa, double ya
, double xb, double yb) {
return
(close__enough_Q(a, b)

? ((a + b) * 5e-1)
: true
? display("+"),
total__iterations = (total__iterations + 1.),
((ya < yb)
? b = xb,
xb = xa,
yb = ya,
xa = (a + (mphi * (b - a))),
__MIP__try(a
, b
, xa
, fun(xa)
, xb
, yb)

: true
? a = xa,
xa = xb,
```



```

        ya = yb,
        xb = (b - (mphi * (b - a))),
        __MIP__try(a
        , b
        , xa
        , ya
        , xb
        , fun(xb))

        : _infinity)
        : _infinity);
    }
bool close_enough_Q/*46*/ (double x, double y){
    return
    (abs((x - y)) < tolerance);
}
double tolerance/*48*/ = 1e-3;
double total__iterations/*49*/ = 0.;
double xmin/*50*/ = 0.;
int main(){
xmin = golden__section__search(a
    , b)
    ;
    display("Interval=\t[");
    display(a);
    display(" , ");
    display(b);
    display("]\n");
    display("Total number of iteranions=");
    display(total__iterations);
    newline();
    display("xmin=\t\t");
    display(xmin);
    newline();
    display("f(xmin)=\t");
    display(fun(xmin));
    newline();
    std::cin.get();
    return
0;
}

```

Code is saved to file golden21.cpp !

Распечатка файла golden21.cpp .

```

/* MIP */
#include "mlisp.h"
extern double a/*2*/;
extern double b/*2*/;
double fun/*3*/ (double x) ;
double golden__section__search/*7*/ (double a, double
b) ;
double golden__start/*15*/ (double a, double b) ;
extern double mphi/*24*/;
double __MIP__try/*25*/ (double a, double b
, double xa, double ya

```

```

, double xb, double yb) ;
bool close__enough_Q/*46*/ (double x, double y);
extern double tolerance/*48*/;
extern double total__iterations/*49*/;
extern double xmin/*50*/;
//_____
double a/*2*/ = 2.;
double b/*2*/ = 6.;
double fun/*3*/ (double x) {
x = (x - (21. / 22.));
return
(expt((x - 3.)
, 4.)
- expt(atan(x)
, 3.)
- 2.);
}
double golden__section__search/*7*/ (double a, double b)
{
{
double xmin( ((a < b)
? golden__start(a
, b)

: true
? golden__start(b
, a)

: _infinity) );
newline();
return xmin;
}
}
double golden__start/*15*/ (double a, double b) {
total__iterations = 0.;
{
double xa( (a + (mphi * (b - a))) ),
xb( (b + (- (mphi * (b - a)))) );
return __MIP__try(a
, b
, xa
, fun(xa)

```

```

    , xb
    , fun(xb))
    ;
}
}
double mphi/*24*/ = ((3. - sqrt(5.)) * (1. / 2e+0));
double __MIP__try/*25*/ (double a, double b
    , double xa, double ya
    , double xb, double yb) {
return
(close__enough_Q(a, b)

? ((a + b) * 5e-1)
: true
? display("+"),
total__iterations = (total__iterations + 1.),
((ya < yb)
? b = xb,
xb = xa,
yb = ya,
xa = (a + (mphi * (b - a))),
__MIP__try(a
, b
, xa
, fun(xa)
, xb
, yb)

: true
? a = xa,
xa = xb,
ya = yb,
xb = (b - (mphi * (b - a))),
__MIP__try(a
, b
, xa
, ya
, xb
, fun(xb))

: _infinity)
: _infinity);

```

```

    }
    bool close__enough_Q/*46*/ (double x, double y){
        return
        (abs((x - y)) < tolerance);
    }
    double tolerance/*48*/ = 1e-3;
    double total__iterations/*49*/ = 0.;
    double xmin/*50*/ = 0.;
    int main(){
        xmin = golden__section__search(a
        , b)
        ;
        display("Interval=\t[");
        display(a);
        display(" , ");
        display(b);
        display("]\n");
        display("Total number of iteranions=");
        display(total__iterations);
        newline();
        display("xmin=\t\t");
        display(xmin);
        newline();
        display("f(xmin)=\t");
        display(fun(xmin));
        newline();
        std::cin.get();
        return
        0;
    }

```

Скриншот запуска задачи на C++.

```

mosik@LAPTOP-69S778GL:~/sp_cw1$ ./golden21
+++++
Interval=      [2 , 6]
Total number of iteranions=18
xmin=          4.412874391742154
f(xmin)=       -4.099152034991492

```

Контрольная задача №3 – coin21.

Полный скриншот трансляции без трассировки.

```

Source>coin21
Source:coin21.ss
1|;coin21.ss
2|(define VARIANT 21)
3|(define LAST-DIGIT-OF-GROUP-NUMBER 8)
4|(define KINDS-OF-COINS 5)
5|
6|(define (first-denomination kinds-of-coins)
7|  (cond((= kinds-of-coins 1) 1)
8|        ((= kinds-of-coins 2) 3)
9|        ((= kinds-of-coins 3) 10)
10|       ((= kinds-of-coins 4) 20)
11|       ((= kinds-of-coins 5) 50)
12|       (#t 0)
13|  )
14|)
15|
16|(define (count-change amount)
17|  (display "_____\n amount: ")
18|  (display amount)
19|  (newline)
20|  (display "KINDS-OF-COINS: ")
21|  (display KINDS-OF-COINS)
22|  (newline)
23|  (let(
24|    (largest-coin (first-denomination KINDS-OF-COINS))
25|    )
26|    (display "largest-coin: ")
27|    (display largest-coin)
28|    (newline)
29|    (cond((and (< 0 amount)(< 0 KINDS-OF-COINS)(< 0 largest-coin))
30|          (display "List of coin denominations: ")
31|            (denomination-list KINDS-OF-COINS)
32|            (display "count-change= ")
33|            (cc amount KINDS-OF-COINS)
34|            )
35|          (#t (display "Improper parameter value!\ncount-change= ") -1)
36|    )
37|  )
38|)
39|
40|(define(NOT? x?)(= 0(cond(x? 1)(#t 0))))
41|
42|(define (pier? x? y?)
43|  (NOT? (or x? y?))
44|)
45|
46|(define (cc amount kinds-of-coins)
47|  (cond((= amount 0) 1)
48|        ((pier? (< amount 0)(= kinds-of-coins 0))
49|          (+ (cc amount (- kinds-of-coins 1))
50|            (cc (- amount (first-denomination kinds-of-coins)) kinds-of-coins))
51|        )
52|        (#t 0)
53|  )
54|)
55|

```

```

56| (define (denomination-list kinds-of-coins)
57|   (cond(= kinds-of-coins 0) (newline) 0)
58|     (#t (display (first-denomination kinds-of-coins))
59|           (display " ")
60|           (denomination-list (- kinds-of-coins 1))
61|         )
62|   )
63| )
64|
65| (define (GR-AMOUNT)
66|   (remainder (+ (* 100 LAST-DIGIT-OF-GROUP-NUMBER) VARIANT) 231)
67| )
68|
69| (display "Variant ")
70| (display VARIANT)
71| (newline)
72| (newline)
73| (display (count-change 100)) (newline)
74| (display (count-change(GR-AMOUNT))) (newline)
75| (set! KINDS-OF-COINS 13)
76| (display (count-change 100)) (newline)
77| (display "(c) Moiseenkov I.P. 2021\n")
78|
79|
80|
81|

```

Code:

```

/* MIP */
#include "mlisp.h"
extern double VARIANT/*2*/;
    extern double LAST_DIGIT_OF_GROUP_NUMBER/*3*/;
    extern double KINDS_OF_COINS/*4*/;
    double first_denomination/*6*/ (double kinds_of_coins) ;
    double count_change/*16*/ (double amount) ;
    bool NOT_Q/*40*/ (double x_Q);
    bool pier_Q/*42*/ (double x_Q, double y_Q);
    double cc/*46*/ (double amount, double kinds_of_coins) ;
    double denomination_list/*56*/ (double kinds_of_coins) ;
    double GR_AMOUNT/*65*/ () ;
    //
double VARIANT/*2*/ = 21.;
    double LAST_DIGIT_OF_GROUP_NUMBER/*3*/ = 8.;
    double KINDS_OF_COINS/*4*/ = 5.;
    double first_denomination/*6*/ (double kinds_of_coins) {
return
((kinds_of_coins == 1.)
? 1.
: (kinds_of_coins == 2.)
? 3.
: (kinds_of_coins == 3.)
? 10.
: (kinds_of_coins == 4.)
? 20.
: (kinds_of_coins == 5.)
? 50.
: true
? 0.
: _infinity);
}

```

```

double count_change/*16*/ (double amount) {
    display("_____\\n amount: ");
    display(amount);
    newline();
    display("KINDS-OF-COINS: ");
    display(KINDS__OF__COINS);
    newline();
    {
double largest_coin( first_denomination(KINDS__OF__COINS) );
    display("largest-coin: ");
    display(largest_coin);
    newline();
    return (((0. < amount) && (0. < KINDS__OF__COINS) && (0. < largest_coin))
? display("List of coin denominations: "),
    denomination_list(KINDS__OF__COINS),
    display("count-change= "),
    cc(amount
    , KINDS__OF__COINS)

    : true
    ? display("Improper parameter value!\\ncount-change= "),
    -1.
    : _infinity);
    }
}

bool NOT_Q/*40*/ (double x_Q){
    return
    (0. == (x_Q
    ? 1.
    : true
    ? 0.
    : _infinity));
}

bool pier_Q/*42*/ (double x_Q, double y_Q){
    return
    NOT_Q((x_Q || y_Q));
}

double cc/*46*/ (double amount, double kinds_of_coins) {
    return
    ((amount == 0.)
    ? 1.
    : pier_Q((amount < 0.), (kinds_of_coins == 0.))

    ? (cc(amount
    , (kinds_of_coins - 1.))
    + cc((amount - first_denomination(kinds_of_coins))
    , kinds_of_coins)
    )
    : true
    ? 0.
    : _infinity);
}

double denomination_list/*56*/ (double kinds_of_coins) {
    return
    ((kinds_of_coins == 0.)
    ? newline(),
    0.
    : true
    ? display(first_denomination(kinds_of_coins)),
    display(" "),
    denomination_list((kinds_of_coins - 1.))
    : _infinity);
}

```

```
double GR__AMOUNT/*65*/ () {
    return
    remainder(((100. * LAST__DIGIT__OF__GROUP__NUMBER) + VARIANT)
    , 231.)
    ;
}
int main(){
    display("Variant ");
    display(VARIANT);
    newline();
    newline();
    display(count__change(100.));
    newline();
    display(count__change(GR__AMOUNT()));
    newline();
    KINDS__OF__COINS = 13.;
    display(count__change(100.));
    newline();
    display("(c) Moiseenkov I.P. 2021\n");
    std::cin.get();
    return
    0;
}
```

Code is saved to file coin21.cpp !

Распечатка файла coin21.cpp .

```
/* MIP */
#include "mlisp.h"
extern double VARIANT/*2*/;
    extern double LAST__DIGIT__OF__GROUP__NUMBER/*3
*/;
    extern double KINDS__OF__COINS/*4*/;
    double first__denomination/*6*/ (double kinds__of__co
ins) ;
    double count__change/*16*/ (double amount) ;
    bool NOT_Q/*40*/ (double x_Q);
    bool pier_Q/*42*/ (double x_Q, double y_Q);
    double cc/*46*/ (double amount, double kinds__of__coi
ns) ;
    double denomination__list/*56*/ (double kinds__of__co
ins) ;
    double GR__AMOUNT/*65*/ () ;
    //_____
double VARIANT/*2*/ = 21.;
    double LAST__DIGIT__OF__GROUP__NUMBER/*3*/ = 8.
;
    double KINDS__OF__COINS/*4*/ = 5.;
    double first__denomination/*6*/ (double kinds__of__co
ins) {
    return
```



```

((kinds__of__coins == 1.)
 ? 1.
 : (kinds__of__coins == 2.)
 ? 3.
 : (kinds__of__coins == 3.)
 ? 10.
 : (kinds__of__coins == 4.)
 ? 20.
 : (kinds__of__coins == 5.)
 ? 50.
 : true
 ? 0.
 : _infinity);
}
double count__change/*16*/ (double amount) {
display("_____\n amount: ");
display(amount);
newline();
display("KINDS-OF-COINS: ");
display(KINDS__OF__COINS);
newline();
{
double largest__coin( first__denomination(KINDS__OF__C
OINS) );
display("largest-coin: ");
display(largest__coin);
newline();
return (((0. < amount) && (0. < KINDS__OF__COINS) &
& (0. < largest__coin))
 ? display("List of coin denominations: "),
denomination__list(KINDS__OF__COINS),
display("count-change= "),
cc(amount
, KINDS__OF__COINS)

: true
 ? display("Improper parameter value!\ncount-
change= "),
-1.
: _infinity);
}
}

```

```

bool NOT_Q/*40*/ (double x_Q){
    return
    (0. == (x_Q
        ? 1.
        : true
        ? 0.
        : _infinity));
}
bool pier_Q/*42*/ (double x_Q, double y_Q){
    return
    NOT_Q((x_Q || y_Q));
}
double cc/*46*/ (double amount, double kinds__of__coins)
{
    return
    ((amount == 0.)
        ? 1.
        : pier_Q((amount < 0.), (kinds__of__coins == 0.))

        ? (cc(amount
            , (kinds__of__coins - 1.))
            + cc((amount - first__denomination(kinds__of__coins))
            , kinds__of__coins)
            )
        : true
        ? 0.
        : _infinity);
}
double denomination__list/*56*/ (double kinds__of__coins
) {
    return
    ((kinds__of__coins == 0.)
        ? newline(),
        0.
        : true
        ? display(first__denomination(kinds__of__coins)),
        display(" "),
        denomination__list((kinds__of__coins - 1.))
        : _infinity);
}
double GR__AMOUNT/*65*/ ( ) {
    return

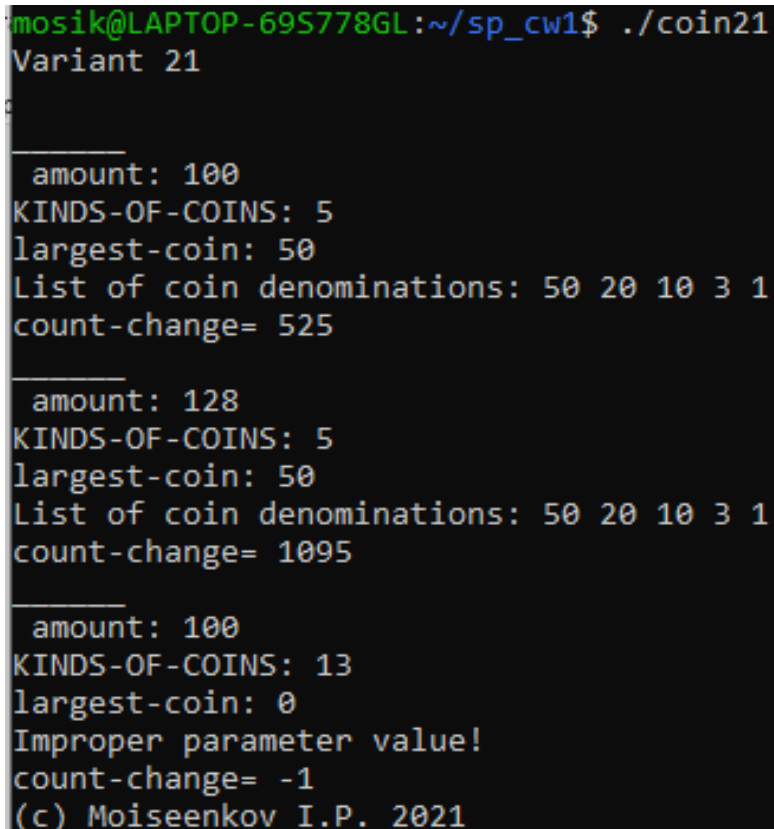
```

```

    remainder(((100. * LAST__DIGIT__OF__GROUP__NUMBER
) + VARIANT)
    , 231.)
    ;
}
int main(){
    display("Variant ");
    display(VARIANT);
    newline();
    newline();
    display(count__change(100.));
    newline();
    display(count__change(GR__AMOUNT()));
    newline();
    KINDS__OF__COINS = 13.;
    display(count__change(100.));
    newline();
    display("(c) Moiseenkov I.P. 2021\n");
    std::cin.get();
    return
0;
}

```

Скриншот запуска задачи на C++.



```

mosik@LAPTOP-69S778GL:~/sp_cw1$ ./coin21
Variant 21

amount: 100
KINDS-OF-COINS: 5
largest-coin: 50
List of coin denominations: 50 20 10 3 1
count-change= 525

amount: 128
KINDS-OF-COINS: 5
largest-coin: 50
List of coin denominations: 50 20 10 3 1
count-change= 1095

amount: 100
KINDS-OF-COINS: 13
largest-coin: 0
Improper parameter value!
count-change= -1
(c) Moiseenkov I.P. 2021

```

Распечатка файла code-gen.cpp.

```
/* $j21 */
#include "code-gen.h"
using namespace std;
void tCG::init(){declarations.clear();
    Authentication = "MIP";
}
int tCG::p01(){ // S -> PROG
    string header = "/* " + Authentication + " */\n";
    header += "#include \"mlisp.h\"\n";
    header += declarations;
    header += "//_____ \n";
    S1->obj = header + S1->obj;
    return 0;
}
int tCG::p02(){ // PROG -> CALCS
    S1->obj = "int main(){\n " + S1->obj +
"std::cin.get();\n\t return\n 0;\n\t }\n";
    return 0;
}
int tCG::p03(){ // PROG -> DEFS
    S1->obj += "int main(){\n "
        "display(\"No calculations!\");\n\t newline();\n\t "
        "std::cin.get();\n\t return\n 0;\n\t }\n";
    return 0;
}
int tCG::p04(){ // PROG -> DEFS CALCS
    S1->obj += "int main(){\n " + S2->obj +
"std::cin.get();\n\t return\n 0;\n\t }\n";
    return 0;
}
int tCG::p05(){ // E -> $id
    S1->obj = decor(S1->name);
    return 0;
}
int tCG::p06(){ // E -> $int
    S1->obj = S1->name + ".";
    return 0;
}
int tCG::p07(){ // E -> $dec
```

```

    S1->obj = S1->name;
    return 0;
}
int tCG::p08(){ // E -> AREX
    return 0;
}
int tCG::p09(){ // E -> COND
    return 0;
}
int tCG::p10(){ // E -> CPROC
    return 0;
}
int tCG::p11(){ // CPROC -> HCPROC )
    S1->obj += ")";
    if (S1->count >= 2) {
        S1->obj += "\n\t ";
    }
    return 0;
}
int tCG::p12(){ // HCPROC -> ( $id
    S1->obj = decor(S2->name) + "(";
    return 0;
}
int tCG::p13(){ // HCPROC -> HCPROC E
    if (S1->count) {
        S1->obj += "\n\t , ";
    }
    S1->obj += S2->obj;
    ++S1->count;
    return 0;
}
int tCG::p14(){ // AREX -> HAREX E )
    if (S1->name == "/" && S1->count == 0) {
        S1->obj = "(1. " + S1->obj + " " + S2->obj + ")";
    }
    else {
        S1->obj = "(" + S1->obj + " " + S2->obj + ")";
    }
    return 0;
}
int tCG::p15(){ // HAREX -> ( AROP
    S1->obj = S2->obj;

```

```

    S1->name = S2->name;
    return 0;
}
int tCG::p16(){ // HAREX -> HAREX E
    if (S1->count == 0)
        S1->obj = S2->obj + " " + S1->name;
    else
        S1->obj = S1->obj + " " + S2->obj + " " + S1->name;
    ++S1->count;
    return 0;
}
int tCG::p17(){ // AROP -> +
    S1->obj = S1->name;
    return 0;
}
int tCG::p18(){ // AROP -> -
    S1->obj = S1->name;
    return 0;
}
int tCG::p19(){ // AROP -> *
    S1->obj = S1->name;
    return 0;
}
int tCG::p20(){ // AROP -> /
    S1->obj = S1->name;
    return 0;
}
int tCG::p21(){ // COND -> ( cond BRANCHES )
    S1->obj = "(" + S3->obj + "_infinity)";
    return 0;
}
int tCG::p22(){ // BRANCHES -> CLAUS
    return 0;
}
int tCG::p23(){ // BRANCHES -> CLAUS BRANCHES
    S1->obj += S2->obj;
    return 0;
}
int tCG::p24(){ // CLAUS -> ( BOOL CLAUSB )
    S1->obj += S2->obj + "\n\t? " + S3->obj + "\n\t: ";
    return 0;
}
}

```

```

int tCG::p25(){ // CLAUSB -> E
    return 0;
}
int tCG::p26(){ // CLAUSB -> INTER CLAUSB
    S1->obj += ",\n\t " + S2->obj;
    ++S1->count;
    return 0;
}
int tCG::p27(){ // STR -> $str
    S1->obj = S1->name;
    return 0;
}
int tCG::p28(){ // STR -> SIF
    return 0;
}
int tCG::p29(){ // SIF -> ( if BOOL STR STR )
    S1->obj = "(" + S3->obj + "\n\t? " + S4->obj + "\n\t: "
+ S5->obj + ")";
    return 0;
}
int tCG::p30(){ // BOOL -> $bool
    S1->obj = (S1->name == "#t" ? "true" : "false");
    return 0;
}
int tCG::p31(){ // BOOL -> $idq
    S1->obj = decor(S1->name);
    return 0;
}
int tCG::p32(){ // BOOL -> REL
    return 0;
}
int tCG::p33(){ // BOOL -> OR
    return 0;
}
int tCG::p34(){ // BOOL -> AND
    return 0;
}
int tCG::p35(){ // BOOL -> CPRED
    return 0;
}
int tCG::p36(){ // CPRED -> HCPRED )
    S1->obj += ")";
}

```

```

    if (S1->count >= 2) {
        S1->obj += "\n\t ";
    }
    return 0;
}
int tCG::p37(){ // HCPRED -> ( $idq
    S1->obj = decor(S2->name) + "(";
    return 0;
}
int tCG::p38(){ // HCPRED -> HCPRED ARG
    if (S1->count) {
        S1->obj += S1->count % 2 ? ", " : "\n\t , ";
    }
    S1->obj += S2->obj;
    ++S1->count;
    return 0;
}
int tCG::p39(){ // ARG -> E
    return 0;
}
int tCG::p40(){ // ARG -> BOOL
    return 0;
}
int tCG::p41(){ // REL -> ( = E E )
    S1->obj = "(" + S3->obj + " == " + S4->obj + ")";
    return 0;
}
int tCG::p42(){ // REL -> ( < E E )
    S1->obj = "(" + S3->obj + " < " + S4->obj + ")";
    return 0;
}
int tCG::p43(){ // OR -> HOR BOOL )
    S1->obj = "(" + S1->obj + S2->obj + ")";
    return 0;
}
int tCG::p44(){ // HOR -> ( or
    return 0;
}
int tCG::p45(){ // HOR -> HOR BOOL
    S1->obj += S2->obj + " || ";
    return 0;
}
}

```



```

int tCG::p46(){ // AND -> HAND BOOL )
    S1->obj = "(" + S1->obj + S2->obj + ")";
    return 0;
}
int tCG::p47(){ // HAND -> ( and
    return 0;
}
int tCG::p48(){ // HAND -> HAND BOOL
    S1->obj += S2->obj + " && ";
    return 0;
}
int tCG::p49(){ // SET -> HSET E )
    S1->obj += S2->obj;
    return 0;
}
int tCG::p50(){ // HSET -> ( set! $id
    S1->obj = decor(S3->name) + " = ";
    return 0;
}
int tCG::p51(){ // DISPSET -> ( display E )
    S1->obj = "display(" + S3->obj + ")";
    return 0;
}
int tCG::p52(){ // DISPSET -> ( display BOOL )
    S1->obj = "display(" + S3->obj + ")";
    return 0;
}
int tCG::p53(){ // DISPSET -> ( display STR )
    S1->obj = "display(" + S3->obj + ")";
    return 0;
}
int tCG::p54(){ // DISPSET -> ( newline )
    S1->obj = "newline()";
    return 0;
}
int tCG::p55(){ // DISPSET -> SET
    return 0;
}
int tCG::p56(){ // INTER -> DISPSET
    return 0;
}
int tCG::p57(){ // INTER -> E

```

```

    return 0;
}
int tCG::p58(){ // CALCS -> CALC
    return 0;
}
int tCG::p59(){ // CALCS -> CALCS CALC
    S1->obj += S2->obj;
    return 0;
}
int tCG::p60(){ // CALC -> E
    S1->obj = "display(" + S1->obj + ");\n\t newline();\n\t";
    return 0;
}
int tCG::p61(){ // CALC -> BOOL
    S1->obj = "display(" + S1->obj + ");\n\t newline();\n\t";
    return 0;
}
int tCG::p62(){ // CALC -> STR
    S1->obj = "display(" + S1->obj + ");\n\t newline();\n\t";
    return 0;
}
int tCG::p63(){ // CALC -> DISPSET
    S1->obj += ";\n\t ";
    return 0;
}
int tCG::p64(){ // DEFS -> DEF
    return 0;
}
int tCG::p65(){ // DEFS -> DEFS DEF
    S1->obj += S2->obj;
    return 0;
}
int tCG::p66(){ // DEF -> PRED
    return 0;
}
int tCG::p67(){ // DEF -> VAR
    return 0;
}
int tCG::p68(){ // DEF -> PROC

```

```

    return 0;
}
int tCG::p69(){ // PRED -> HPRED BOOL )
    S1->obj += "return\n " + S2->obj + ";\n\t }\n";
    return 0;
}
int tCG::p70(){ // HPRED -> PDPAR )
    S1->obj += ")\n";
    declarations += S1->obj + ";\n\t ";
    S1->obj += "{\n ";
    return 0;
}
int tCG::p71(){ // PDPAR -> ( define ( $idq
    S1->obj = "bool " + decor(S4->name) + "/*" + S4->line
+ "*/ (";
    S1->count = 0;
    return 0;
}
int tCG::p72(){ // PDPAR -> PDPAR $idq
    if (S1->count) {
        S1->obj += S1->count % 2 ? ", " : "\n\t , ";
    }
    S1->obj += "double " + decor(S2->name);
    ++S1->count;
    return 0;
}
int tCG::p73(){ // PDPAR -> PDPAR $id
    if (S1->count) {
        S1->obj += S1->count % 2 ? ", " : "\n\t , ";
    }
    S1->obj += "double " + decor(S2->name);
    ++S1->count;
    return 0;
}
int tCG::p74(){ // VAR -> VARDCL E )
    declarations += "extern double " + S1->obj + "/*" + S1-
>line + "*/;\n\t ";
    S1->obj = "double " + S1->obj + "/*" + S1->line + "*/ =
" + S2->obj + ";\n\t ";
    return 0;
}
int tCG::p75(){ // VARDCL -> ( define $id

```

```

    S1->obj = decor(S3->name);
    return 0;
}
int tCG::p76(){ // PROC -> HPROC BLOCK )
    S1->obj += S2->obj + "}\n";
    return 0;
}
int tCG::p77(){ // PROC -> HPROC E )
    S1->obj += "return\n " + S2->obj + ";\n\t }\n";
    return 0;
}
int tCG::p78(){ // HPROC -> PCPAR )
    S1->obj += ") ";
    declarations += S1->obj + ";\n\t ";
    S1->obj += "{\n ";
    return 0;
}
int tCG::p79(){ // HPROC -> HPROC INTER
    S1->obj += S2->obj + ";\n\t ";
    return 0;
}
int tCG::p80(){ // PCPAR -> ( define ( $id
    S1->obj = "double " + decor(S4->name) + "/*" + S4-
>line + "*/ (";
    S1->count = 0;
    S1->name = S4->name;
    return 0;
}
int tCG::p81(){ // PCPAR -> PCPAR $id
    if (S1->count) {
        S1->obj += S1->count % 2 ? ", " : "\n\t , ";
    }
    S1->obj += "double " + decor(S2->name);
    ++S1->count;
    return 0;
}
int tCG::p82(){ // BLOCK -> HBLOCK E )
    S1->obj += "return " + S2->obj + ";\n\t }\n";
    return 0;
}
int tCG::p83(){ // HBLOCK -> BLVAR )
    S1->obj += ";\n\t ";

```

```

    return 0;
}
int tCG::p84(){ // HBLOCK -> HBLOCK INTER
    S1->obj += S2->obj + ";\n\t ";
    return 0;
}
int tCG::p85(){ // BLVAR -> ( let ( LOCDEF
    S1->obj += "{\n double " + S4->obj;
    return 0;
}
int tCG::p86(){ // BLVAR -> BLVAR LOCDEF
    S1->obj += ",\n\t " + S2->obj;
    return 0;
}
int tCG::p87(){ // LOCDEF -> ( $id E )
    S1->obj += decor(S2->name) + "( " + S3->obj + " )";
    return 0;
}
//_____
int tCG::p88(){return 0;} int tCG::p89(){return 0;}
int tCG::p90(){return 0;} int tCG::p91(){return 0;}
int tCG::p92(){return 0;} int tCG::p93(){return 0;}
int tCG::p94(){return 0;} int tCG::p95(){return 0;}
int tCG::p96(){return 0;} int tCG::p97(){return 0;}
int tCG::p98(){return 0;} int tCG::p99(){return 0;}
int tCG::p100(){return 0;} int tCG::p101(){return 0;}
int tCG::p102(){return 0;} int tCG::p103(){return 0;}
int tCG::p104(){return 0;} int tCG::p105(){return 0;}
int tCG::p106(){return 0;} int tCG::p107(){return 0;}
int tCG::p108(){return 0;} int tCG::p109(){return 0;}
int tCG::p110(){return 0;}

```

ДОПОЛНИТЕЛЬНОЕ ЗАДАНИЕ

После определения БУЛЕВСКОЙ функции поместить комментарий с именем ее прототипа в Микролиспе. Например, `bool pier_Q/*33*/(...){`

```

...
} // pier?

```

Продукции могут получать имена ТОЛЬКО из атрибута `name` и НЕ могут извлекать их из кода на C++.

Продемонстрировать на контрольной задаче `coin21`.

Скриншот трансляции coin21

```
Input gramma name>j21
Gramma:j21.txt
Source>coin21
Source:coin21.ss
1|;coin21.ss
2|(define VARIANT 21)
3|(define LAST-DIGIT-OF-GROUP-NUMBER 8)
4|(define KINDS-OF-COINS 5)
5|
6|(define (first-denomination kinds-of-coins)
7|  (cond((= kinds-of-coins 1) 1)
8|        ((= kinds-of-coins 2) 3)
9|        ((= kinds-of-coins 3) 10)
10|       ((= kinds-of-coins 4) 20)
11|       ((= kinds-of-coins 5) 50)
12|       (#t 0)
13|  )
14|)
15|
16|(define (count-change amount)
17|  (display "_____\n amount: ")
18|  (display amount)
19|  (newline)
20|  (display "KINDS-OF-COINS: ")
21|  (display KINDS-OF-COINS)
22|  (newline)
23|  (let(
24|    (largest-coin (first-denomination KINDS-OF-COINS))
25|  )
26|    (display "largest-coin: ")
27|    (display largest-coin)
28|    (newline)
29|    (cond((and (< 0 amount)(< 0 KINDS-OF-COINS)(< 0 largest-coin))
30|          (display "List of coin denominations: ")
31|          (denomination-list KINDS-OF-COINS)
32|          (display "count-change= ")
33|          (cc amount KINDS-OF-COINS)
34|          )
35|          (#t (display "Improper parameter value!\ncount-change= ") -1)
36|    )
37|  )
38|)
39|
40|(define(NOT? x?)(= 0(cond(x? 1)(#t 0))))
41|
42|(define (pier? x? y?)
43|  (NOT? (or x? y?))
44|)
45|
46|(define (cc amount kinds-of-coins)
47|  (cond((= amount 0) 1)
48|        ((pier? (< amount 0)(= kinds-of-coins 0))
49|          (+ (cc amount (- kinds-of-coins 1))
50|            (cc (- amount (first-denomination kinds-of-coins)) kinds-of-coins))
51|        )
52|        (#t 0)
53|  )

```

```

54|)
55|
56|(define (denomination-list kinds-of-coins)
57|  (cond((= kinds-of-coins 0) (newline) 0)
58|        (#t (display (first-denomination kinds-of-coins))
59|              (display " ")
60|              (denomination-list (- kinds-of-coins 1))
61|              )
62|  )
63|)
64|
65|(define (GR-AMOUNT)
66|  (remainder (+ (* 100 LAST-DIGIT-OF-GROUP-NUMBER) VARIANT) 231)
67|)
68|
69|(display "Variant ")
70|(display VARIANT)
71|(newline)
72|(newline)
73|(display (count-change 100)) (newline)
74|(display (count-change(GR-AMOUNT))) (newline)
75|(set! KINDS-OF-COINS 13)
76|(display (count-change 100)) (newline)
77|(display "(c) Moiseenkov I.P. 2021\n")
78|
79|
80|
81|

```

Code:

```

/* MIP */
#include "mlisp.h"
extern double VARIANT/*2*/;
extern double LAST__DIGIT__OF__GROUP__NUMBER/*3*/;
extern double KINDS__OF__COINS/*4*/;
double first__denomination/*6*/ (double kinds__of__coins) ;
double count__change/*16*/ (double amount) ;
bool NOT_Q/*40*/ (double x_Q);
bool pier_Q/*42*/ (double x_Q, double y_Q);
double cc/*46*/ (double amount, double kinds__of__coins) ;
double denomination__list/*56*/ (double kinds__of__coins) ;
double GR__AMOUNT/*65*/ () ;
//
double VARIANT/*2*/ = 21.;
double LAST__DIGIT__OF__GROUP__NUMBER/*3*/ = 8.;
double KINDS__OF__COINS/*4*/ = 5.;
double first__denomination/*6*/ (double kinds__of__coins) {
return
((kinds__of__coins == 1.)
? 1.
: (kinds__of__coins == 2.)
? 3.
: (kinds__of__coins == 3.)
? 10.
: (kinds__of__coins == 4.)
? 20.
: (kinds__of__coins == 5.)
? 50.

```

```

        : true
        ? 0.
        : _infinity);
    }
double count_change/*16*/ (double amount) {
    display("_____ \n amount: ");
    display(amount);
    newline();
    display("KINDS-OF-COINS: ");
    display(KINDS__OF__COINS);
    newline();
    {
        double largest_coin( first_denomination(KINDS__OF__COINS) );
        display("largest-coin: ");
        display(largest_coin);
        newline();
        return (((0. < amount) && (0. < KINDS__OF__COINS) && (0. < largest_coin))
            ? display("List of coin denominations: "),
            denomination_list(KINDS__OF__COINS),
            display("count-change= "),
            cc(amount
                , KINDS__OF__COINS)

            : true
            ? display("Improper parameter value!\ncount-change= "),
            -1.
            : _infinity);
    }
}
bool NOT_Q/*40*/ (double x_Q){
    return
        (0. == (x_Q
            ? 1.
            : true
            ? 0.
            : _infinity));
    } // NOT?
bool pier_Q/*42*/ (double x_Q, double y_Q){
    return
        NOT_Q((x_Q || y_Q));
    } // pier?
double cc/*46*/ (double amount, double kinds_of_coins) {
    return
        ((amount == 0.)
            ? 1.
            : pier_Q((amount < 0.), (kinds_of_coins == 0.))

            ? (cc(amount
                , (kinds_of_coins - 1.))
                + cc((amount - first_denomination(kinds_of_coins))
                , kinds_of_coins)
            )
            : true
            ? 0.
            : _infinity);
    }
double denomination_list/*56*/ (double kinds_of_coins) {
    return
        ((kinds_of_coins == 0.)
            ? newline(),
            0.

```



```

        : true
        ? display(first__denomination(kinds__of__coins)),
          display(" "),
          denomination__list((kinds__of__coins - 1.))
        : _infinity);
    }
double GR__AMOUNT/*65*/ ( ) {
    return
    remainder(((100. * LAST__DIGIT__OF__GROUP__NUMBER) + VARIANT)
              , 231.)
    ;
}
int main(){
    display("Variant ");
    display(VARIANT);
    newline();
    newline();
    display(count__change(100.));
    newline();
    display(count__change(GR__AMOUNT()));
    newline();
    KINDS__OF__COINS = 13.;
    display(count__change(100.));
    newline();
    display("(c) Moiseenkov I.P. 2021\n");
    std::cin.get();
    return
    0;
}

```

Code is saved to file coin21.cpp !

Скриншот запуска на C++

```

mosik@LAPTOP-69S778GL:~/sp_cw1$ ./coin21
Variant 21
_____
amount: 100
KINDS-OF-COINS: 5
largest-coin: 50
List of coin denominations: 50 20 10 3 1
count-change= 525
_____
amount: 128
KINDS-OF-COINS: 5
largest-coin: 50
List of coin denominations: 50 20 10 3 1
count-change= 1095
_____
amount: 100
KINDS-OF-COINS: 13
largest-coin: 0
Improper parameter value!
count-change= -1
(c) Moiseenkov I.P. 2021

```

Распечатка ИЗМЕНЕННЫХ продукций

```
int tCG::p69(){ // PRED -> HPRED BOOL )
    S1->obj += "return\n " + S2->obj + ";\n\t } // " + S1-
>name + "\n";
    return 0;
}
```

```
int tCG::p71(){ // PDPAR -> ( define ( $idq
    S1->obj = "bool " + decor(S4->name) + "/*" + S4->line
+ "*/ (";
    S1->count = 0;
    S1->name = S4->name;
    return 0;
}
```