**Студент: Епанешников В.С.**
**Группа: М80-206Б-19**
**Номер по списку: 9**

# «СИСТЕМЫ ПРОГРАММИРОВАНИЯ»
## Курсовая работа 2021.
## Часть 1.

**Перечень документов в отчете.**
**Вариант грамматики:n09**

**Контрольная задача №1 – zeller.**
**Полный скриншот трансляции  без трассировки**

**Распечатка файла zeller.cpp .**
**>**

```
Input gramma name>n09
Gramma:n09.txt
Source>zeller
Source:zeller.ss
   1|;zeller.ss
   2|(define (day-of-week)
   3| (zeller dd
   4|       (cond((< mm 3)(+ mm 10))(else (- mm 2)))
   5|       (remainder (cond((< mm 3)(- yyyy 1)) (else yyyy)) 100)
   6|       (quotient (cond((< mm 3)(- yyyy 1)) (else 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|       (else d)
 22| )
 23|)
 24|
 25|
 26|
 27|(define (birthday dw)
 28|;                    ^{0,...,6}
 29| (display "Your were born on ")
 30|    (display
 31|     (cond((= dw 1)"Monday ")
 32|           ((= dw 2)"Tuesday ")
 33|           ((= dw 3)"Wednesday ")
 34|           ((= dw 4)"Thursday ")
 35|           ((= dw 5)"Friday ")
 36|           ((= dw 6)"Saturday ")
 37|           (else "Sunday ") ))
 38|  (display dd)(display ".")
 39|  (display mm)(display ".")
 40| yyyy
 41|)
 42|(define dd 24)
 43|(define mm 7)
 44|(define yyyy 2001)
 45|(birthday (day-of-week))
 46|
```

```
_________________
Code:
/*  EVS   */
#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/*27*/ (double dw);
        extern double dd/*42*/ ;
        extern double mm/*43*/ ;
        extern double yyyy/*44*/ ;
        //_________________
double day__of__week/*2*/ (){
 return
 zeller(dd
        , ((mm < 3.)
       ? (mm + 10.)
       : ((mm - 2.)))
        , remainder(((mm < 3.)
```

```
        ? (yyyy - 1.)
        : (yyyy))
         , 100.)

         , quotient(((mm < 3.)
        ? (yyyy - 1.)
        : (yyyy))
         , 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.)
        : (d));
         }

double birthday/*27*/ (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/*42*/ = 24.;

double mm/*43*/ = 7.;

double yyyy/*44*/ = 2001.;
          int main(){
 display("Calculations!");
          newline();
          display(birthday(day__of__week()));
          newline();
          std::cin.get();
          return 0;
          }

Code is saved to file zeller.cpp !
```

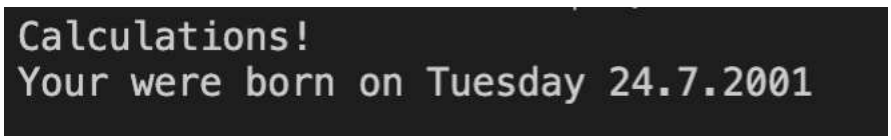
**Распечатка файла zeller.cpp**

```
>
/*  EVS   */
#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/*27*/ (double dw);
    extern double dd/*42*/ ;
    extern double mm/*43*/ ;
    extern double yyyy/*44*/ ;
    //____________________
double day__of__week/*2*/ (){
 return
 zeller(dd
    , ((mm < 3.)
   ? (mm + 10.)
   : ((mm - 2.)))
```

```
    , remainder(((mm < 3.)
   ? (yyyy - 1.)
   : (yyyy))
    , 100.)

    , quotient(((mm < 3.)
   ? (yyyy - 1.)
   : (yyyy))
    , 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.)
   : (d));
    }

double birthday/*27*/ (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/*42*/ = 24.;

double mm/*43*/ = 7.;

double yyyy/*44*/ = 2001.;
    int main(){
 display("Calculations!");
    newline();
    display(birthday(day__of__week()));
    newline();
    std::cin.get();
    return 0;
    }
```

**Скриншот запуска задачи на C++.**
**>**




## Контрольная задача №2 – golden21.

**Полный скриншот трансляции  без трассировки**
**>**

```
Input gramma name>n09
Gramma:n09.txt
Source>golden21
Source:golden21.ss
   1|; golden21
   2|; Епанешников М80-206Б-19
   3|; [5, 7] 5,712
   4|; e^(-z) +sin(z)
   5|(define a 5)(define b 7)
   6|(define (fun x)
   7| (set! x (- x (/ 109 110)))
   8| (+(exp(- x))(sin x))
   9|)
  10|(define (golden-section-search a bz)
  11| (let(
  12|       (xmin(cond((< a b)(golden-start a b)) (else (golden-start b a ))))
  13|      )
  14|      (newline)
  15|      xmin
  16| )
  17|)
  18|(define (golden-start a b)
  19| (set! total-iterations 0)
  20| (let(
  21|       (xa (+ a (* mphi(- b a))))
  22|       (xb (+ b (-(* mphi(- b a)))))
  23|      )
  24|      (try a b xa (fun xa) xb (fun xb))
  25| )
  26|)
  27|(define mphi (* (- 3(sqrt 5))(/ 2.0e0)))
  28|(define (try a b xa ya xb yb)
  29| (cond((close-enough? a b)
  30|       (* (+ a b)0.5e0))
  31|       (else (let() (display "+")
  32|              (set! total-iterations (+ total-iterations 1))
  33|              (cond((< ya yb)(let() (set! b xb)
  34|                           (set! xb xa)
  35|                           (set! yb ya)
  36|                           (set! xa (+ a (* mphi(- b a))))
  37|                           (try a b xa (fun xa) xb yb))
  38|                    )
  39|                    (else   (let() (set! a xa)
  40|                           (set! xa xb)
  41|                           (set! ya yb)
  42|                           (set! xb (- b (* mphi(- b a))))
  43|                           (try a b xa ya xb (fun xb)))
  44|                     )
```

```
  45|             );cond...
  46|       );let...
  47| ));cond...
  48|)
  49|(define (close-enough? x y)
  50|  (<(abs (- x y))tolerance))
  51|(define tolerance 0.001e0)
  52|(define total-iterations 0)
  53|(define xmin 0)
  54|(set! xmin(golden-section-search a b))
  55|  (display"Interval=\t[")
  56|  (display a)
  57|  (display" , ")
  58|  (display b)
  59|  (display"]\n")
  60|  (display"Total number of iterations=")
  61|total-iterations
  62|  (display"xmin=\t\t")
  63|xmin
  64|  (display"f(xmin)=\t")
  65|(fun xmin)
  66|
_________________
Code:
/*  EVS   */
#include "mlisp.h"
extern double a/*5*/ ;
         extern double b/*5*/ ;
         double fun/*6*/ (double x);
         double golden__section__search/*10*/ (double a, double bz);
         double golden__start/*18*/ (double a, double b);
         extern double mphi/*27*/ ;
         double __EVS__try/*28*/ (double a, double b
         , double xa, double ya
         , double xb, double yb);
         bool close__enough_Q/*49*/ (double x, double y);
         extern double tolerance/*51*/ ;
         extern double total__iterations/*52*/ ;
         extern double xmin/*53*/ ;
         //_________________
double a/*5*/ = 5.;

double b/*5*/ = 7.;

double fun/*6*/ (double x){
 x = (x - (109. / 110.));
         return
 (exp((- x)) + sin(x));
```

```
         }

double golden__section__search/*10*/ (double a, double bz){
 {
 double xmin(((a < b)
        ? golden__start(a
         , b)

        : (golden__start(b
         , a)
         )));
         newline();
         return
 xmin;
         }
}

double golden__start/*18*/ (double a, double b){
 total__iterations = 0.;
         {
 double xa((a + (mphi * (b - a)))),
         xb((b + (- (mphi * (b - a)))));
         return
 __EVS__try(a
         , b
         , xa
         , fun(xa)
         , xb
         , fun(xb))
         ;
         }
}

double mphi/*27*/ = ((3. - sqrt(5.)) * (1. / 2.0e0));

double __EVS__try/*28*/ (double a, double b
         , double xa, double ya
         , double xb, double yb){
 return
 (close__enough_Q(a, b)
        ? ((a + b) * 0.5e0)
        : (display("+"),
         total__iterations = (total__iterations + 1.),
         ((ya < yb)
        ? b = xb,
         xb = xa,
         yb = ya,
         xa = (a + (mphi * (b - a))),
```

```
          __EVS__try(a
          , b
          , xa
          , fun(xa)
          , xb
          , yb)

         : (a = xa,
          xa = xb,
          ya = yb,
          xb = (b - (mphi * (b - a))),
          __EVS__try(a
          , b
          , xa
          , ya
          , xb
          , fun(xb))
          ))));
          }

bool close__enough_Q/*49*/ (double x, double y){
 return (abs((x - y)) < tolerance);
          }

double tolerance/*51*/ = 0.001e0;

double total__iterations/*52*/ = 0.;

double xmin/*53*/ = 0.;
          int main(){
 display("Calculations!");
          newline();
          xmin = golden__section__search(a
          , b)
          ;
          display("Interval=\t[");
          display(a);
          display(" , ");
          display(b);
          display("]\n");
          display("Total number of iterations=");
          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 .**
**>**

```
/*  EVS   */
#include "mlisp.h"
extern double a/*5*/ ;
    extern double b/*5*/ ;
    double fun/*6*/ (double x);
    double golden__section__search/*10*/ (double a,
double bz);
    double golden__start/*18*/ (double a, double b);
    extern double mphi/*27*/ ;
    double __EVS__try/*28*/ (double a, double b
    , double xa, double ya
    , double xb, double yb);
    bool close__enough_Q/*49*/ (double x, double y);
    extern double tolerance/*51*/ ;
    extern double total__iterations/*52*/ ;
    extern double xmin/*53*/ ;
    //____________________
double a/*5*/ = 5.;

double b/*5*/ = 7.;

double fun/*6*/ (double x){
 x = (x - (109. / 110.));
    return
 (exp((- x)) + sin(x));
    }

double golden__section__search/*10*/ (double a, double
bz){
 {
 double xmin(((a < b)
```

```
   ? golden__start(a
    , b)

   : (golden__start(b
    , a)
    )));
    newline();
    return
 xmin;
    }
}

double golden__start/*18*/ (double a, double b){
 total__iterations = 0.;
    {
 double xa((a + (mphi * (b - a)))),
    xb((b + (- (mphi * (b - a)))));
    return
 __EVS__try(a
    , b
    , xa
    , fun(xa)
    , xb
    , fun(xb))
    ;
    }
}

double mphi/*27*/ = ((3. - sqrt(5.)) * (1. / 2.0e0));

double __EVS__try/*28*/ (double a, double b
    , double xa, double ya
    , double xb, double yb){
 return
 (close__enough_Q(a, b)
   ? ((a + b) * 0.5e0)
   : (display("+"),
    total__iterations = (total__iterations + 1.),
    ((ya < yb)
   ? b = xb,
    xb = xa,
    yb = ya,
```

```
    xa = (a + (mphi * (b - a))),
    __EVS__try(a
    , b
    , xa
    , fun(xa)
    , xb
    , yb)

   : (a = xa,
    xa = xb,
    ya = yb,
    xb = (b - (mphi * (b - a))),
    __EVS__try(a
    , b
    , xa
    , ya
    , xb
    , fun(xb))
    ))));
    }

bool close__enough_Q/*49*/ (double x, double y){
 return (abs((x - y)) < tolerance);
    }

double tolerance/*51*/ = 0.001e0;

double total__iterations/*52*/ = 0.;

double xmin/*53*/ = 0.;
    int main(){
 display("Calculations!");
    newline();
    xmin = golden__section__search(a
    , b)
    ;
    display("Interval=\t[");
    display(a);
    display(" , ");
    display(b);
    display("]\n");
    display("Total number of iterations=");
```

```
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++.**
**>**

```
Calculations!
+++++++++++++++
Interval=       [5 , 7]
Total number of iterations=16
xmin=           5.712322723532441
f(xmin)=        -0.9910566934375074
```

## Контрольная задача №3 – coin21.

**Полный скриншот трансляции  без трассировки (крупный белый шрифт на ярком черном фоне).**
**>**

```
Source>coin21
Source:coin21.ss
   1|; coin21
   2|; Епанешников М80-206Б-19
   3|
   4|(define VARIANT 9)
   5|(define LAST-DIGIT-OF-GROUP-NUMBER 6)
   6|(define KINDS-OF-COINS 7)
   7|
   8|(define (first-denomination kinds-of-coins)
   9|  (cond((= kinds-of-coins 1) 1)
```

```
      (else (cond((= kinds-of-coins 2) 2)
      (else (cond((= kinds-of-coins 3) 3)
      (else (cond((= kinds-of-coins 4) 5)
      (else (cond((= kinds-of-coins 5) 10)
      (else (cond((= kinds-of-coins 6) 15)
      (else (cond((= kinds-of-coins 7) 20)
      (else 0))))))))))))
)


(define (AND3? x? y? z?)
     ( = 1 (cond(x? (cond(y? (cond(z? 1) (else 0))) (else 0))) (else 0)))
)
(define (AND2? x? y?)
  ( = 1 (cond(x? (cond(y? 1) (else 0))) (else 0)))
)
(define (count-change amount)
  (display "______")
  (newline)
  (display " amount: ")
  (newline)
  (display "KINDS-OF-COINS: ")
  (display KINDS-OF-COINS)
  (newline)
  (let(
       (largest-coin (first-denomination KINDS-OF-COINS))
      )
      (display "largest-coin: ")
      (display largest-coin)
      (newline)
      (cond((AND3? (< 0 amount) (< 0 KINDS-OF-COINS) (< 0 largest-coin))
         (let()
           (display "List of coin denominations: ")
           (denomination-list KINDS-OF-COINS)
           (display "count-change= ")
           (cc amount KINDS-OF-COINS)
         ))
         (else (let()
           (display "Improrer parametr value!")
           (newline)
           (display "count-change =") -1))
      )
   )
)

(define (pier? x? y?)
```

```
  (not (OR? x? y?))
)


(define (OR? x? y?)
    (not(AND2? (not x?) (not y?)))
)


(define (cc amount kinds-of-coins)
  (cond( (= amount 0) 1)
       (else (cond((pier? (< amount 0) (= kinds-of-coins 0))
        (+ (cc amount (- kinds-of-coins 1))
           (cc (- amount (first-denomination kinds-of-coins)) kinds-of-coins)))
        (else 0))))
)

(define (denomination-list kinds-of-coins)
  (cond((= kinds-of-coins 0) (let() (newline) 0))
     (else (let()
       (display (first-denomination kinds-of-coins))
       (display " ")
       (denomination-list (- kinds-of-coins 1))
     )))
)


(define (GR-AMOUNT)
  (remainder (+ (* 100 LAST-DIGIT-OF-GROUP-NUMBER) VARIANT) 231)
)

(display "Variant ")
(display VARIANT)
(newline)
(newline)
(display (count-change 100))
(newline)
(display (count-change (GR-AMOUNT)))
(newline)
(set! KINDS-OF-COINS 13)
(display (count-change 100))
(newline)
(display "(c) Epaneshnikov V.S. 2021")
(newline)

```

Code:

```
/*  EVS   */
```

```
#include "mlisp.h"
extern double VARIANT/*4*/ ;
         extern double LAST__DIGIT__OF__GROUP__NUMBER/*5*/ ;
         extern double KINDS__OF__COINS/*6*/ ;
         double first__denomination/*8*/ (double kinds__of__coins);
         bool AND3_Q/*20*/ (double x_Q, double y_Q
         , double z_Q);
         bool AND2_Q/*24*/ (double x_Q, double y_Q);
         double count__change/*28*/ (double amount);
         bool pier_Q/*57*/ (double x_Q, double y_Q);
         bool OR_Q/*62*/ (double x_Q, double y_Q);
         double cc/*67*/ (double amount, double kinds__of__coins);
         double denomination__list/*75*/ (double kinds__of__coins);
         double GR__AMOUNT/*85*/ ();
         //__________________
double VARIANT/*4*/ = 9.;

double LAST__DIGIT__OF__GROUP__NUMBER/*5*/ = 6.;

double KINDS__OF__COINS/*6*/ = 7.;

double first__denomination/*8*/ (double kinds__of__coins){
 return
 ((kinds__of__coins == 1.)
        ? 1.
        : (((kinds__of__coins == 2.)
        ? 2.
        : (((kinds__of__coins == 3.)
        ? 3.
        : (((kinds__of__coins == 4.)
        ? 5.
        : (((kinds__of__coins == 5.)
        ? 10.
        : (((kinds__of__coins == 6.)
        ? 15.
        : (((kinds__of__coins == 7.)
        ? 20.
        : (0.))))))))))));
         }

bool AND3_Q/*20*/ (double x_Q, double y_Q
         , double z_Q){
 return (1. == (x_Q
        ? (y_Q
        ? (z_Q
        ? 1.
        : (0.))
        : (0.))
```

```
        : (0.)));
         }

bool AND2_Q/*24*/ (double x_Q, double y_Q){
 return (1. == (x_Q
        ? (y_Q
        ? 1.
        : (0.))
        : (0.)));
         }

double count__change/*28*/ (double amount){
 display("______");
         newline();
         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
 (AND3_Q((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)

        : (display("Improrer parametr value!"),
         newline(),
         display("count-change ="),
         -1.));
         }
}

bool pier_Q/*57*/ (double x_Q, double y_Q){
 return (!OR_Q(x_Q, y_Q));
         }

bool OR_Q/*62*/ (double x_Q, double y_Q){
 return (!AND2_Q((!x_Q), (!y_Q)));
         }
```

```
double cc/*67*/ (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)
         )
        : (0.))));
         }

double denomination__list/*75*/ (double kinds__of__coins){
 return
 ((kinds__of__coins == 0.)
        ? newline(),
         0.
        : (display(first__denomination(kinds__of__coins)),
         display(" "),
         denomination__list((kinds__of__coins - 1.))));
         }

double GR__AMOUNT/*85*/ (){
 return
 remainder(((100. * LAST__DIGIT__OF__GROUP__NUMBER) + VARIANT)
         , 231.)
         ;
         }
int main(){
 display("Calculations!");
         newline();
         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) Epaneshnikov V.S. 2021");
         newline();
         std::cin.get();
         return 0;
```

```
          }

  Code is saved to file coin21.cpp !
```

**Распечатка файла coin21.cpp .**
**>**

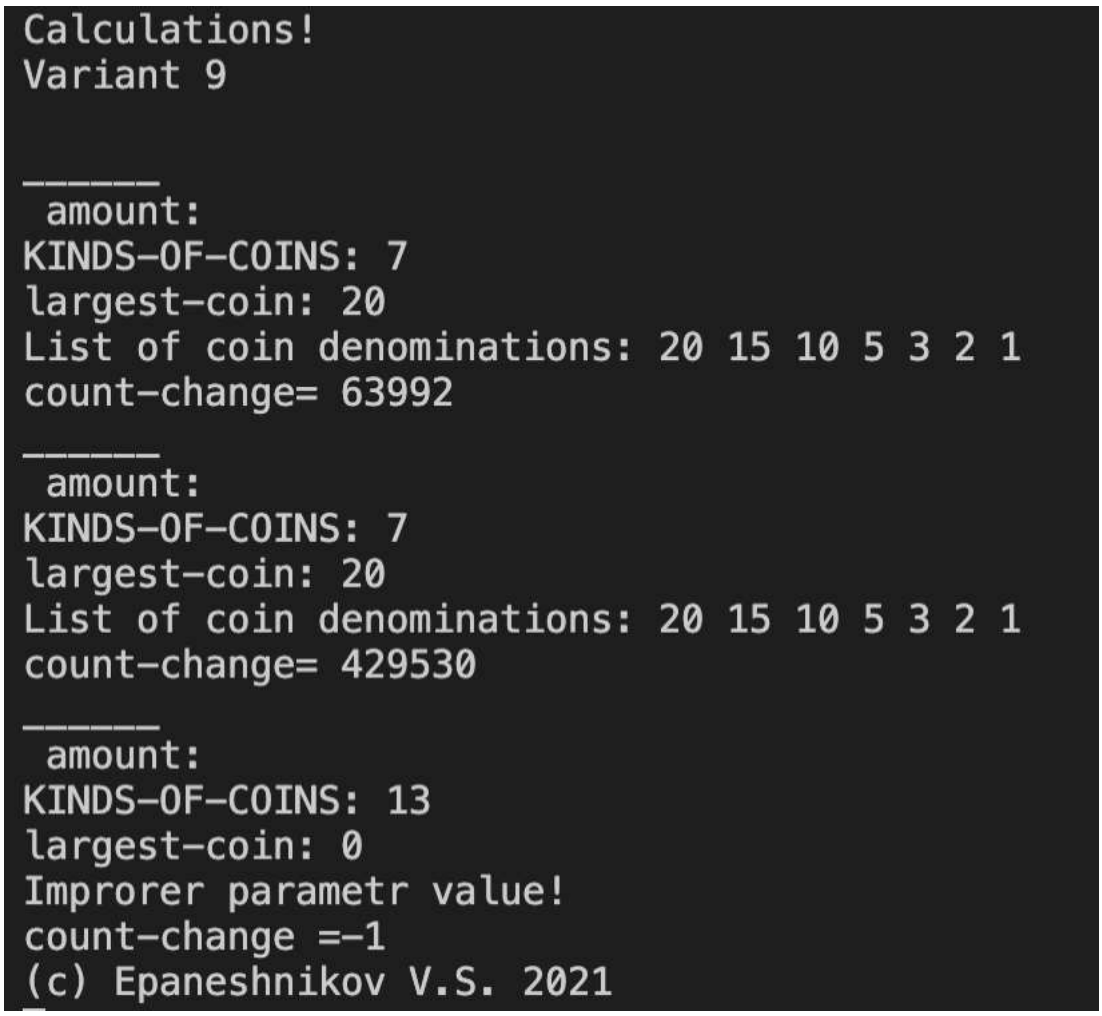
```
/*  EVS   */
#include "mlisp.h"
extern double VARIANT/*4*/ ;
    extern double
LAST__DIGIT__OF__GROUP__NUMBER/*5*/ ;
    extern double KINDS__OF__COINS/*6*/ ;
    double first__denomination/*8*/ (double
kinds__of__coins);
    bool AND3_Q/*20*/ (double x_Q, double y_Q
    , double z_Q);
    bool AND2_Q/*24*/ (double x_Q, double y_Q);
    double count__change/*28*/ (double amount);
    bool pier_Q/*57*/ (double x_Q, double y_Q);
    bool OR_Q/*62*/ (double x_Q, double y_Q);
    double cc/*67*/ (double amount, double
kinds__of__coins);
    double denomination__list/*75*/ (double
kinds__of__coins);
    double GR__AMOUNT/*85*/ ();
    //____________________
double VARIANT/*4*/ = 9.;

double LAST__DIGIT__OF__GROUP__NUMBER/*5*/ = 6.;

double KINDS__OF__COINS/*6*/ = 7.;

double first__denomination/*8*/ (double
kinds__of__coins){
 return
 ((kinds__of__coins == 1.)
   ? 1.
   : (((kinds__of__coins == 2.)
   ? 2.
   : (((kinds__of__coins == 3.)
```

```
   ? 3.
   : (((kinds__of__coins == 4.)
   ? 5.
   : (((kinds__of__coins == 5.)
   ? 10.
   : (((kinds__of__coins == 6.)
   ? 15.
   : (((kinds__of__coins == 7.)
   ? 20.
   : (0.))))))))))));
    }

bool AND3_Q/*20*/ (double x_Q, double y_Q
    , double z_Q){
 return (1. == (x_Q
   ? (y_Q
   ? (z_Q
   ? 1.
   : (0.))
   : (0.))
   : (0.)));
    }

bool AND2_Q/*24*/ (double x_Q, double y_Q){
 return (1. == (x_Q
   ? (y_Q
   ? 1.
   : (0.))
   : (0.)));
    }

double count__change/*28*/ (double amount){
 display("_______");
    newline();
    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
 (AND3_Q((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)

   : (display("Improrer parametr value!"),
    newline(),
    display("count-change ="),
    -1.));
    }
}

bool pier_Q/*57*/ (double x_Q, double y_Q){
 return (!OR_Q(x_Q, y_Q));
    }

bool OR_Q/*62*/ (double x_Q, double y_Q){
 return (!AND2_Q((!x_Q), (!y_Q)));
    }

double cc/*67*/ (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)
    )
   : (0.))));
    }
```

```
double denomination__list/*75*/ (double
kinds__of__coins){
 return
 ((kinds__of__coins == 0.)
   ? newline(),
    0.
   : (display(first__denomination(kinds__of__coins)),
    display(" "),
    denomination__list((kinds__of__coins - 1.))));
    }

double GR__AMOUNT/*85*/ (){
 return
 remainder(((100. *
LAST__DIGIT__OF__GROUP__NUMBER) + VARIANT)
    , 231.)
    ;
    }
int main(){
 display("Calculations!");
    newline();
    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) Epaneshnikov V.S. 2021");
    newline();
    std::cin.get();
    return 0;
    }
```

**Скриншот запуска задачи на C++.**
**>**




**Распечатка файла code-gen.cpp.**
**>**

```
/* $n09 */
#include "code-gen.h"
using namespace std;

void tCG::init() {
   declarations.clear();
   Authentication = "EVS";
}

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 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 0;\n\t }\n";
   return 0;
}

int tCG::p04() { //   PROG -> DEFS CALCS
   S1->obj += "int main(){\n "
           "display(\"Calculations!\");\n\t newline();\n\t ";
   S1->obj += S2->obj;
   S1->obj += "std::cin.get();\n\t return 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 -> EASYLET
   return 0;
}
int tCG::p11() { //       E -> CPROC
   return 0;
}

int tCG::p12() { //     AREX -> HAREX E )
   if (S1->count == 0 && S1->name == "/")
      S1->obj = "(1. " + S1->obj + " " + S2->obj + ")";
   else
      S1->obj = "(" + S1->obj + " " + S2->obj + ")";
   return 0;
}

int tCG::p13() { //    HAREX -> ( AROP
   S1->obj = S2->obj;
   S1->name = S2->name;
   return 0;
}

int tCG::p14() { //    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::p15() { //    AROP -> +
   S1->obj = S1->name;
   return 0;
}
```

```
int tCG::p16() { //    AROP -> -
   S1->obj = S1->name;
   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() { //  EASYLET -> HEASYL E )
   if (S1->count != 0)
      S1->obj += S2->obj + ";\n\t";
   S1->obj += S2->obj;
   ++(S1->count);
   return 0;
}

int tCG::p20()
{ //   HEASYL -> ( let ( )
   return 0;
}

int tCG::p21() { //   HEASYL -> HEASYL INTER
   S1->obj += S2->obj + ",\n\t ";
   return 0;
}

int tCG::p22() { //    CPROC -> HCPROC )
   if (S1->count <= 1)
      S1->obj = S1->obj + ")";
   else
      S1->obj = S1->obj + ")\n\t ";
   return 0;
}

int tCG::p23() { //   HCPROC -> ( $id
```

```
   S1->obj = decor(S2->name) + "(";
   return 0;
}

int tCG::p24() { //   HCPROC -> HCPROC E
   if (S1->count)
      S1->obj += "\n\t , ";
   S1->obj += S2->obj;
   ++(S1->count);
   return 0;
}

int tCG::p25() { //    COND -> ( cond BRANCHES )
   S1->obj = "(" + S3->obj + ")";
   return 0;
}

int tCG::p26() { // BRANCHES -> CLAUS ELSE
   S1->obj += S2->obj;
   return 0;
}

int tCG::p27() { //    CLAUS -> ( BOOL E )
   S1->obj = "" + S2->obj + "\n\t? " + S3->obj + "\n\t: ";
   return 0;
}

int tCG::p28() { //    ELSE -> ( else E )
   S1->obj = "(" + S3->obj + ")";
   return 0;
}

int tCG::p29() { //    STR -> $str
   S1->obj = S1->name;
   return 0;
}

int tCG::p30() { //    STR -> SCOND
   return 0;
}

int tCG::p31() { //   SCOND -> ( cond SBRANCHES )
```

```
   S1->obj = "(" + S3->obj + ")";
   return 0;
}

int tCG::p32() { // SBRANCHES -> SELSE
   return 0;
}

int tCG::p33()
{ // SBRANCHES -> SCLAUS SBRANCHES
   S1->obj = S1->obj + S2->obj;
   return 0;
}

int tCG::p34() { //   SCLAUS -> ( BOOL STR )
   S1->obj = S2->obj + "\n\t? " + S3->obj + "\n\t: ";
   return 0;
}

int tCG::p35() { //    SELSE -> ( else STR )
   S1->obj = "(" + S3->obj + ")";
   return 0;
}

int tCG::p36() { //     BOOL -> $bool
   S1->obj += (S1->name == "#t" ? "true" : "false");
   return 0;
}

int tCG::p37() { //     BOOL -> $idq
   S1->obj = decor(S1->name);
   return 0;
}

int tCG::p38() { //     BOOL -> REL
   return 0;
}

int tCG::p39() { //     BOOL -> ( not BOOL )
   S1->obj = "(!" + S3->obj + ")";
   return 0;
}
```

```
int tCG::p40() { //    BOOL -> CPRED
   return 0;
}

int tCG::p41() { //   CPRED -> HCPRED )
   S1->obj += ")";
   return 0;
}

int tCG::p42() { //   HCPRED -> ( $idq
   S1->obj = decor(S2->name) + "(";
   return 0;
}

int tCG::p43() { //   HCPRED -> HCPRED ARG
   if (S1->count)
      S1->obj += S1->count % 2 ? ", " : "\n\t , ";
   S1->obj += S2->obj;
   ++(S1->count);
   return 0;
}

int tCG::p44() { //    ARG -> E
   return 0;
}

int tCG::p45() { //    ARG -> BOOL
   return 0;
}

int tCG::p46() { //    REL -> ( = E E )
   S1->obj = "(" + S3->obj + " == " + S4->obj + ")";
   return 0;
}

int tCG::p47() { //    REL -> ( < E E )
   S1->obj = "(" + S3->obj + " < " + S4->obj + ")";
   return 0;
}

int tCG::p48() { //    SET -> HSET E )
```

```
   S1->obj += S2->obj;
   return 0;
}

int tCG::p49() { //     HSET -> ( set! $id
   S1->obj = decor(S3->name) + " = ";
   return 0;
}

int tCG::p50() { //  DISPSET -> ( display E )
   S1->obj = "display(" + S3->obj + ")";
   return 0;
}

int tCG::p51() { //  DISPSET -> ( display BOOL )
   S1->obj = "display(" + S3->obj + ")";
   return 0;
}

int tCG::p52() { //  DISPSET -> ( display STR )
   S1->obj = "display(" + S3->obj + ")";
   return 0;
}

int tCG::p53() { //  DISPSET -> ( newline )
   S1->obj = "newline()";
   return 0;
}

int tCG::p54() { //  DISPSET -> SET
   return 0;
}

int tCG::p55() { //    INTER -> DISPSET
   return 0;
}

int tCG::p56() { //    INTER -> E
   return 0;
}

int tCG::p57() { //    CALCS -> CALC
```

```
   return 0;
}

int tCG::p58() { //    CALCS -> CALCS CALC
   S1->obj += S2->obj;
   return 0;
}

int tCG::p59() { //    CALC -> E
   S1->obj = "display(" + S1->obj + ");\n\t newline();\n\t
";
   return 0;
}

int tCG::p60() { //    CALC -> BOOL
   S1->obj = "display(" + S1->obj + ");\n\t newline();\n\t
";
   return 0;
}

int tCG::p61() { //    CALC -> STR
   S1->obj = "display(" + S1->obj + ");\n\t newline();\n\t
";
   return 0;
}

int tCG::p62() { //    CALC -> DISPSET
   S1->obj = S1->obj + ";\n\t ";
   return 0;
}

int tCG::p63() { //    DEFS -> DEF
   return 0;
}

int tCG::p64() { //    DEFS -> DEFS DEF
   S1->obj = S1->obj + "\n" + S2->obj;
   return 0;
}

int tCG::p65() { //     DEF -> PRED
   return 0;
```

```
}

int tCG::p66() { //     DEF -> VAR
   return 0;
}

int tCG::p67() { //     DEF -> PROC
   return 0;
}

int tCG::p68() { //    PRED -> HPRED BOOL )
   S1->obj += S2->obj + ";\n\t }\n";
   return 0;
}

int tCG::p69() { //    HPRED -> PDPAR )
   S1->obj += ")";
   declarations += S1->obj + ";\n\t ";
   S1->obj += "{\n return ";
   S1->count = 0;
   return 0;
}

int tCG::p70() { //    PDPAR -> ( define ( $idq
   S1->obj = "bool " + decor(S4->name) + "/*" + S4->line
+ "*/ (";
   S1->count = 0;
   return 0;
}

int tCG::p71() { //    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::p72() { //    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::p73() { //     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::p74() { //   VARDCL -> ( define $id
   S1->obj = decor(S3->name);
   return 0;
}

int tCG::p75() { //     PROC -> HPROC BLOCK )
   S1->obj += S2->obj + "}\n";
   return 0;
}

int tCG::p76() { //     PROC -> HPROC E )
   S1->obj += "return\n " + S2->obj + ";\n\t }\n";
   return 0;
}

int tCG::p77() { //    HPROC -> PCPAR )
   S1->obj += ")";
   declarations += S1->obj + ";\n\t ";
   S1->obj += "{\n ";
   return 0;
}

int tCG::p78() { //    HPROC -> HPROC INTER
   S1->obj += S2->obj + ";\n\t ";
   return 0;
}

int tCG::p79() { //    PCPAR -> ( define ( $id
   S1->obj = "double " + decor(S4->name) + "/*" + S4-
>line + "*/ (";
```

```
   S1->count = 0;
   S1->name = S4->name;
   return 0;
}

int tCG::p80() { //    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::p81() { //    BLOCK -> HBLOCK E )
   S1->obj = S1->obj + "return\n " + S2->obj + ";\n\t
}\n";
   return 0;
}

int tCG::p82() { //   HBLOCK -> BLVAR )
   S1->obj = S1->obj + ";\n\t ";
   return 0;
}

int tCG::p83() { //   HBLOCK -> HBLOCK INTER
   S1->obj += S2->obj + ";\n\t ";
   return 0;
}

int tCG::p84() { //  BLVAR -> ( let ( LOCDEF
   S1->obj = "{\n double " + S4->obj;
   return 0;
}

int tCG::p85() { //  BLVAR -> BLVAR LOCDEF
   S1->obj += ",\n\t " + S2->obj;
   return 0;
}

int tCG::p86() { // LOCDEF -> ( $id E )
   S1->obj += decor(S2->name) + "(" + S3->obj + ")";
   return 0;
```

```
}
//_____________________________
int tCG::p87() { 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; }
```