А. А. ПЕТРОСЯН

# ОСНОВНЫЕ БАЗИСНЫЕ ФУНКЦИИ ДЛЯ АВТОМАТИЗИРОВАННОГО СИНТЕЗА ПРОГРАММ

## ПОСТАНОВКА ЗАДАЧИ

В современном программировании большое место занимает задача синтеза программ. Для синтеза программ входной информацией служат условия задачи, а выход — это готовая программа. Так что при успешном решении проблемы синтеза программ отпадает не только необходимость писать программу, но и отлаживать ее, если условия задачи заданы правильно.

В этой статье рассматривается один из подходов к решению задачи синтеза программ.

В нашем подходе используем «Теорему неподвижной точки» из теории рекурсивных функций.

Пусть даны функции $\varphi_i(x_1, x_2, \dots x_n)$, где аргументы $x_1$, $x_2$—«сверхчисла», а $x_3, \dots x_n$—обыкновенные натуральные целые числа меньше $10^{15}$. «Сверхчисло» — это натуральное целое число меньше $10^{700}$. Оно организуется как массив длины 50, где каждый элемент определяется как десятичное целое число меньше $10^{15}$, а они определяются как 15-разрядные числа (но 1 разряд используется для переполнения). $\{\varphi_i\}$— есть «своеобразная» нумерация. Если $i \leqslant 30$, то $\varphi_i$ это базисная функция, а остальные функции получаются из базисных при помощи разных комбинаций.

Наша задача — найти решение уравнения, заданного в неявной форме, то есть найти решение уравнения вида

$$\alpha(x_1 x_2 \dots x_n) = F(x_1, x_2, \alpha(x_1, x_2, g_{11} \dots g_{1n}), \dots \alpha(x_1 x_2 g_{n1} \dots g_{nn})), \quad (1)$$

где $F(x_1 x_2 \dots x_n)$, $g_{ij}(x_1 x_2 x_3 \dots x_n)$ — базисные функции, а $\alpha(x_1 x_2 x_3 \dots \dots x_n)$ — неизвестная функция.

Среди базисных функций есть такие, которые не меняются, но есть и такие, как функции $F$, $g_{ij}$, которые в зависимости от задачи меняются.

Пусть имеем какую-нибудь допустимую нумерацию рекурсивных функций — $\{\varphi_i\}$. Тогда по определению универсальной функции $U$, любую функцию $\gamma$ можно представить как

$$\gamma(x_1 x_2 \dots x_n) = U(i_1, x_1 x_2 \dots x_n) = \varphi_{i_1}(x_1 x_2 \dots x_n).$$

Следовательно, уравнение (1) можно написать:

$$U(i,\ x_1x_2\ldots x_n) = F(x_1,\ x_2,\ U(i,\ x_1x_2g_{11}\ldots g_{1n}),\ \ldots U(i,\ x_1x_2g_{n_1}\ldots g_{nn}))$$

или

$$\varphi_i(x_1x_2\ldots x_n) = F(x_1,\ x_2,\ \varphi_i(x_1x_2g_{11}\ldots g_{1n}),\ \ldots \varphi_i(x_1x_2,\ g_{n1}\ldots g_{nn})).$$

Следовательно, наша задача — найти такое $i_0$, чтобы

$$\varphi_{i_0}(x_1x_2\ldots x_n) = F(x_1,\ x_2,\ \varphi_{i_0}(x_1x_2g_{11}\ldots g_{1n})\ldots \varphi_{i_0}(x_1x_2g_{nn}\ldots g_{nn})). \quad (2)$$

То есть найти «неподвижную точку» для уравнения вида (1).

Пусть $U^*$ есть новая функция, которая определяется как —

$$U^*(z,\ x_2x_3\ldots x_n) = U^*(\langle x_0,\ x_1\rangle,\ x_2x_3\ldots x_n) = \varphi_{x_0}(x_1,\ x_2,\ \ldots x_n),$$

где $z = \langle x_0, x_1\rangle$ есть Канторовское перечисление номеров пар. Из теории рекурсивных функций знаем, что существует универсальная функция $U_0$ такая, что

$$U_0(x_0,\ x_1x_2\ldots x_n) = U^*(\langle x_0,\ x_1\rangle,\ x_2,\ x_3\ldots x_n).$$

Определим новую функцию

$$U^*(\langle U^*(\langle q,\ q\rangle,\ x_2x_3\ldots x_n),\ x_1\rangle,\ x_2x_3\ldots x_n).$$

Ясно, что для любого $q$ новая функция дает свой номер — $h(q)$.

$$U^*(\langle U^*(\langle q,\ q\rangle,\ x_2x_3\ldots x_n),\ x_1\rangle,\ x_2x_3\ldots x_n) = \varphi_{h(q)}(x_1x_2\ldots x_n), \quad (3)$$

где

$$h(q) = K(0,\ 0,\ K(0,\ 5,\ K(3,\ \langle q,\ q\rangle,\ 0))).$$

Здесь $K$ это функция кодирования, определяющаяся по формуле

$$K(d,\ \alpha_1\alpha_2\ldots\alpha_n) = a + d + b\cdot C^{n+1}(\alpha_1\alpha_2\ldots\alpha_n,\ 0),$$

где $a$ — число базисных функций (у нас $a = 30$); $b$ — число базисных действий (у нас $b = 6$); $C^{n+1}(\alpha_1, \alpha_2, \ldots, \alpha_n 0)$ — Канторовская нумерация наборов; $d$ — код базисных действий ($d \leqslant 5$).

$$d = \begin{cases} 0 \text{ — суперпозиция} \\ 1 \text{ — }\Delta\text{-функция } (\varphi_a^{(a)}) \\ 2 \text{ — универсальная функция} \\ 3 \text{ — введение константы} \\ 4 \text{ — минимизация} \\ 5 \text{ — ветвление} \end{cases}$$

Более подробно эти действия будут обсуждены в дальнейшем.

Такое кодирование дает возможность легко и однозначно декодировать «сверхчисла».

С другой стороны, для любого $i$, уравнение (1) дает новую функцию с номером $f(i)$. То есть

$$F(x_1, x_2, \varphi_i(x_1 x_2 g_{11} \ldots g_{1n}), \ldots \varphi_i(x_1, x_2, g_{n1} \ldots g_{nn})) = \varphi_{f(i)}(x_1 x_2 \ldots x_n).$$

Ясно, что функция $f(i)$ зависит от $F$, $g_{11} \ldots g_{1n} \ldots g_{n1} \ldots g_{nn}$. Пусть $a_f = \ulcorner f \cdot h \urcorner$ (номер суперпозиции функций $f$ и $h$). Тогда

$$\varphi_{a_f}(x_1 x_2 \ldots x_n) \underset{def}{\equiv} U^*(\langle U^*(\langle a_f, a_f \rangle, x_2 x_3 \ldots x_n), x_1 \rangle, x_2 x_3 \ldots x_n) =$$

$$= U^*(\langle f \cdot h(a_f), x_1 \rangle, x_2 x_3 \ldots x_n) = \varphi_{f \cdot h(a_f)}(x_1 x_2 x_3 \ldots x_n).$$

То есть $i_o = h(a_f)$ есть неподвижная точка для уравнения вида (1), и эта неподвижная точка — константа. В зависимости от задачи меняются функции $F, g_{11} \ldots g_{1n} \ldots g_{n1} \ldots g_{nn}$, а также функция $f$, но их номера в базисе не меняются.

Для реализации функции $f$, $h$, а также кодирования и декодирования надо предварительно организовать такие элементарные базисные действия со «сверхчислами», как умножение, сложение «сверхчисел», корень из «сверхчисла» и другие.

## ОРГАНИЗАЦИЯ ЭЛЕМЕНТАРНЫХ БАЗИСНЫХ ДЕЙСТВИЙ СО «СВЕРХЧИСЛАМИ»

В этом параграфе обсудим те элементарные базисные действия, при помощи которых реализуются функции $f$, $h$, кодирование, декодирование. В конце будем давать также конкретную реализацию функции $h$, кодирование, декодирование. В этом параграфе описаны также все необходимые блок-схемы.

Перечисленные элементарные действия реализованы на ЭВМ ЕС-1045. Программы написаны на алгоритмическом языке ПЛ/1 ОС ЕС ЭВМ.

Полный список элементарных действий со «сверхчислами», описание формальных параметров процедур реализации этих действий, а также имена всех процедур представлены на рис. 1.

## СЛОЖЕНИЕ «СВЕРХЧИСЛА» И ЧИСЛА $K \leqslant 99$

Программа *PGSK* предназначена для сложения «сверхчисла» $X$ и числа $K \leqslant 99$.

Вход — $X$ и $K$. Выход — $X = X + K$.

Вызов — CALL PGSK ($X$, $K$).

Сложение осуществляется по следующему алгоритму. «Сверхчисло» $X$ представляется как $X_1 X_2, \ldots, X_{50}$, где $0 \leqslant X_i \leqslant 10^{15} - 1$ для любого $1 \leqslant i \leqslant 50$.

Если $X_{50} + K \geqslant 10^{15}$ то $X'_{50} = X_{50} + K - 10^{15}$, $K = 1$ и вычисление продолжаем аналогичным образом уже для $X_{49}$.

Если же $X_{50} + K < 10^{15}$ то $X'_{50} = X_{50} + K$ и вычисление заканчивается. Таким образом получим $X' = X + K$.

1. PGSK $(X, K)$. $X = X + K$. $K$ FIXED (2). $K \leqslant 99$.
2. PGUK $(X, K)$. $X = X \cdot K$. $K$ FIXED (1). $K \leqslant 9$.
3. PGDK $(X, K)$. $X = [X/K]$. $K$ FIXED (1). $K \leqslant 9$.
4. PGV1 $(X)$. $X = X \dot{-} 1$.
5. PGVK $(X, K)$. $X = X \dot{-} K$. $K$ FIXED (2). $K \leqslant 99$.
6. PGU10 $(X)$. $X = X \cdot 10$.
7. PGU100 $(X)$. $X = X \cdot 100$.
8. PGSS $(X, Y, Z)$. $Z = X + Y$.
9. PGVS $(X, Y, Z)$. $Z = X \dot{-} Y$.
10. PGUS $(X, Y)$. $Z = X \cdot Y$.
11. PGKS $(X)$. $X = [\sqrt{X}]$.
12. PKPP $(X, Y, Z)$. $Z = C(X, Y) = \langle X, Y \rangle$.
13. PKPN $(A, Q)$. $A$ (20, 50) FIXED (15).

$$Q = C^{n+1}(A_1, A_2, \ldots, A_n, 0).$$

14. PGKN $(D, A, Z)$. $A$ (20, 50) FIXED (15), $D$ FIXED (2).
15. PGDN $(E, D, A)$. $A$ (20, 50) FIXED (15), $D$ FIXED (2).
16. PGVH $(X)$. $X = H(X)$.

Во всех программах $X$, $Y$, $Z$, $E$, $Q$, $A_i$ «сверхчисла».

Рис. 1. Список элементарных действий со „сверхчислами" и описание их формальных параметров

## УМНОЖЕНИЕ «СВЕРХЧИСЛА» НА $K < 9$

Программа PGUK предназначена для умножения «сверхчисла» на $K$, где $0 \leqslant K \leqslant 9$.

Вход — «сверхчисло» $X$ и число $K \leqslant 9$. Выход — $X = X \cdot K$.

Вызов — CALL PGUK $(X, K)$.

Определение — Натуральное число $K_x$ называется порядком «сверхчисла» $X$, если $10^{11(K_x - 1)} < X \leqslant 10^{11 \cdot K_x}$.

Алгоритм программы PGUK осуществляется как циклическое действие. Сначала вычисляется порядок «сверхчисла» — $K_x$. Цикл повторяется $K_x$ раз, на каждом шаге вычисляя значение $X'_i$ и $O'_i$ следующими формулами.

$$X'_i = \begin{cases} X_i \cdot K + O_{i-1}, & \text{если } X'_i \cdot K + O_{i-1} < 10^{15} \\ X_i \cdot K + O_{i-1} - 10^{15}, & \text{если } X_i \cdot K + O_{i-1} \geqslant 10^{15}, \end{cases}$$

где $1 \leqslant i \leqslant K_x$, а $O_{i-1}$ остаток от шага $(i-1)$. $O_0 = 0$.

$$O_i = \begin{cases} 0, & \text{если } X_i \cdot K + O_{i-1} < 10^{15} \\ X_i \cdot K + O_{i-1} - X'_i, & \text{если } X_i \cdot K + O_{i-1} \geqslant 10^{15}. \end{cases}$$

## ДЕЛЕНИЕ «СВЕРХЧИСЛА» НА $K < 9$

Программа PGDK предназначена для деления «сверхчисла» $X$ на натуральное число $K \leqslant 9$.

Вход — $X$ и $K$. Выход — $X = \left[\frac{X}{K}\right]$.

Вызов — CALL PGDK $(X, K)$.

Алгоритм этой программы следующий:

если $K = 1$ или 0, то результат соответственно будет $X$ или ситуация „ZERODIVIDE", если же $K > 1$, то вычисляется порядок «сверхчисла» $K_x$ и $K'_x = 51 - K_x$, «сверхчисло» $X$ представляется как $X_{K'_x}, X_{k'_x+1}, \ldots X_{50}$, потом циклически на $K$ делятся все $X_i$, где $K'_x \leqslant i \leqslant 50$, на каждом шаге учитывая остаток от предыдущего шага

## ВЫЧИТАНИЕ 1

Программа PGV1 предназначена для вычитания из «сверхчисла» 1.

Алгоритм реализаций следующий:

«Сверхчисло» $X$ представляется как $X = X_1 X_2 \ldots X_{50}$.

Если $X_{50} > 1$, то $X'_{50} = X_{50} - 1$ и заканчивается вычисление, в противном случае $X'_{50} = 10^{15} - 1$ и вычитаем 1 уже от $X_{49}$ аналогичным образом.

Вход — $X$. Выход — $X = X \dot{-} 1$.

Вызов — CALL PGV1 $(X)$.

## ВЫЧИТАНИЕ K

Процедура PGVK предназначена для вычитания числа $K$ из «сверхчисла».

Вход — $X, K$. Выход — $X = X \dot{-} K$.

Вызов — CALL PGVK $(X, K)$.

Алгоритм реализации PGVK, как и алгоритм PGV1, начинает свою работу с представления «сверхчисла» $X$ как $X_1 X_2 \ldots X_{50}$.

Если $X_{50} > K$, то $X_{50} = X_{50} \dot{-} K$ и вычисление заканчивается, в противном случае вычисляется $X_{50} = 10^{15} + X_{50} - K$ и вычитаем 1 уже от $X_{49}$, как в программе PGV1.

## СЛОЖЕНИЕ «СВЕРХЧИСЕЛ»

Программа PGSS предназначена для сложения двух «сверхчисел». Она реализуется как циклический процесс сложения $X_i + Y_i = Z_i$, учитывая перенос.

Заранее, конечно, определяются их порядки $K_x$, $K_y$ и цикл повторяется $K = \max\{K_x, K_y\}$ раз.

Вход — «сверхчисла» $X$, $Y$. Выход — «сверхчисло» $Z = X + Y$.

Вызов — CALL PGSS$(X, Y, Z)$.

## УМНОЖЕНИЕ «СВЕРХЧИСЕЛ»

Программа PGUS предназначена для умножения двух «сверхчисел» $X$, $Y$ соответственно порядка $K_x$ и $K_y$, если $K_x + K_y \leqslant 50$.

Программа PGUS использует подпрограмму PROG1 и реализует следующий алгоритм.

Вычисляются порядки $K_x$ и $K_y$.

Если один из аргументов нуль, то печатается соответственное сообщение и программа заканчивает свою работу с нулевым результатом, в противном случае, если $K_y > K_x$, то есть «сверхчисло» $Y$ больше «сверхчисла» $X$, то с помощью подпрограммы PROG1 вычисляется $Z = Y \cdot X$, а если и это условие не выполняется, то вычисляется $Z = X \cdot Y$ опять с помощью подпрограммы PROG1.

Вход — «сверхчисла» $X$, $Y$. Выход — «сверхчисло» $Z = X \cdot Y$.

Вызов — CALL PGUS$(X, Y, Z)$.

## ВЫЧИТАНИЕ «СВЕРХЧИСЕЛ»

Программа PGVS предназначена для вычитания «сверхчисел».

Вход — «сверхчисла» $X$, $Y$. Выход — «сверхчисло» $Z = X \dot{-} Y$.

Вызов CALL PGVS $(X, Y, Z)$.

Блок-схема PGVS показана на рис. 2.

## УМНОЖЕНИЕ «СВЕРХЧИСЛА» НА 10 (100)

Программа PGU10 (100) предназначена для умножения «сверхчисла» на натуральное число 10 (100).

Алгоритмы реализации этих программ почти одинаковые. Сперва определяется порядок «сверхчисла» — $K_x$. Циклически $K_x$ раз на 10 (100) умножается $X_i$, учитывая перенос. По-разному организуются только переходы.

Вход — «сверхчисло» $X$. Выход — «сверхчисло» $X = X \cdot 10$ (100).

Вызов — CALL PGU10 $(X)$. CALL PGU100 $(X)$.

## ПОДПРОГРАММА PROG1

Подпрограмма PROG1 отдельная процедура в программе PGUS и предназначена для умножения двух «сверхчисел» по следующему алгоритму. «Сверхчисло» $X$, $Y$ можно представить в виде

$$X = \sum_{i=1}^{n} x_i \cdot 10^{14(i-1)} \quad \text{и} \quad Y = \sum_{j=1}^{k} y_j \cdot 10^{14(j-1)},$$

где $n, k \leqslant 50$; $X = x_n x_{n-1} \cdots x_2 x_1$; $Y = y_k y_{k-1} \dots y_2 y_1$.

Пусть $X \geqslant Y$, т. е. $n \geqslant k$.

В представление $X$, $Y$ каждое $0 \leqslant x_i \leqslant 10^{14} - 1$; $0 \leqslant y_j \leqslant 10^{14} - 1$, но такие $x_i$, $y_j$ можно разбить на две части $x_{i_1}$, $x_{i_2}$, где каждый из них $0 \leqslant x_{i_1}, x_{i_2} \leqslant 10^7 - 1$. Соответственно $y_j$. Следовательно, X, $Y$ можно представить в виде

$$X = \sum_{i=1}^{2n} \breve{x}_i \cdot 10^{7(i-1)} \quad \text{и} \quad Y = \sum_{j=1}^{2k} y_j \cdot 10^{7(j-1)}$$

и результат

$$Z = X \cdot Y = z_{n+k} z_{n+k-1} \cdots z_2 z_1 = \breve{z}_{2(n+k)} \tilde{z}_{2(n+k)-1} \cdots \breve{z}_2 \tilde{z}_1,$$

где $0 \leqslant z \leqslant 10^{14} - 1$ и $0 \leqslant \breve{z}_i \leqslant 10^7 - 1$ (в описании алгоритма $n \equiv 2n$; $k \equiv 2k$).

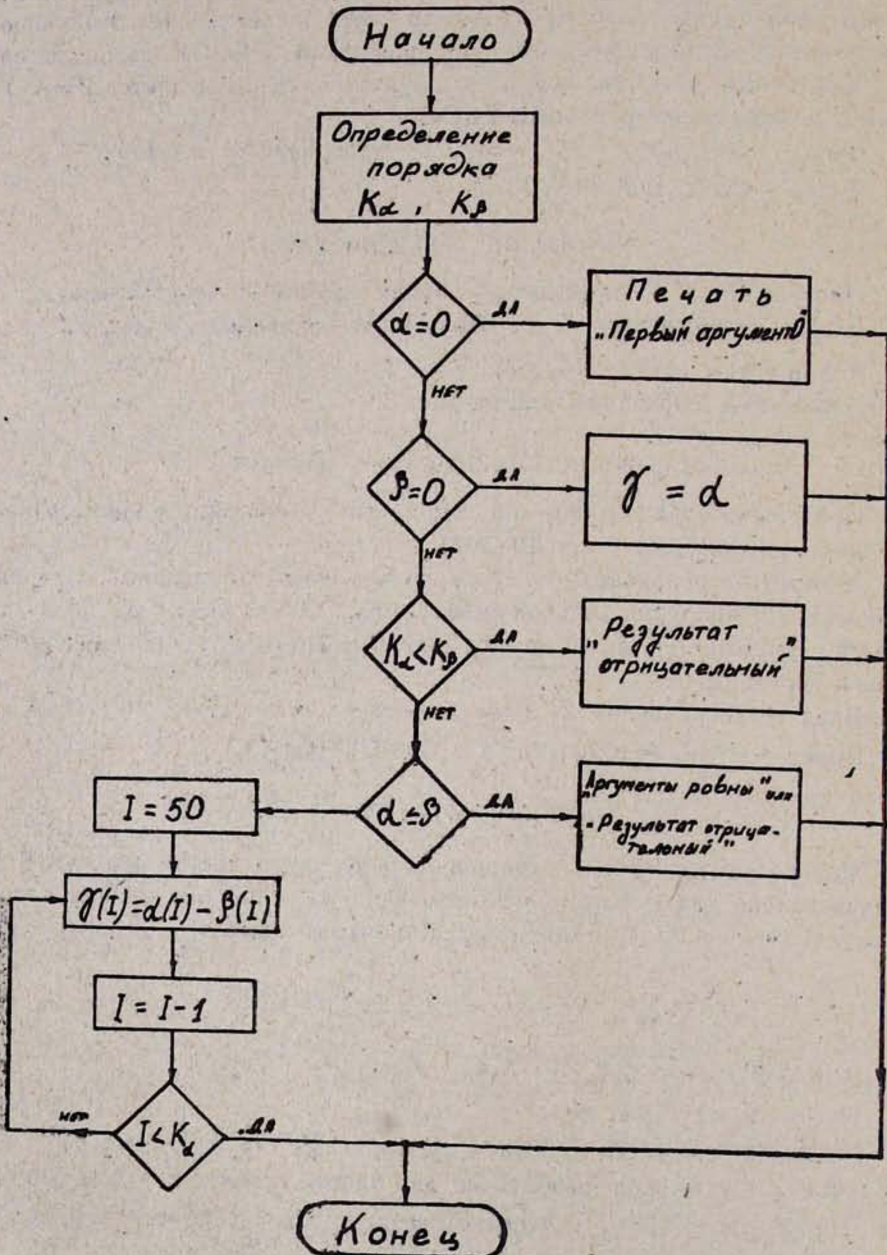


Рис. 2. Блок-схема PGVS

Шаг 1 $\tilde{x}_1 \cdot \bar{y}_1 = \alpha$; $\alpha = \alpha'' - \alpha'$; $\bar{z}_1 = \alpha'$; где $\alpha < 10^{14} - 1$;

$\alpha', \alpha'' < 10^7 - 1$;

Шаг 2 $\tilde{y}_1 \cdot \tilde{x}_2 + \tilde{y}_2 \cdot \bar{x}_1 + \alpha'' = \alpha$; $\alpha = \alpha'' - \alpha'$; $\tilde{z}_2 = \alpha'$; условия те же;

Шаг 3 $\tilde{y}_1 \cdot \tilde{x}_3 + \tilde{y}_2 \cdot \bar{x}_2 + \tilde{y}_3 \cdot \tilde{x}_1 + \alpha'' = \alpha$; $\alpha = \alpha'' - \alpha'$; $\tilde{z}_3 = \alpha'$;

Шаг $i \leqslant n - 1$ $\tilde{y}_1 \cdot \tilde{x}_i + \tilde{y}_2 \cdot \tilde{x}_{i-1} + \cdots + \tilde{y}_i \cdot \tilde{x}_1 + \alpha'' = \alpha$;

$$\alpha = \alpha'' - \alpha'; \quad \tilde{z}_i = \alpha';$$

Шаг $n$ $\tilde{y}_1 \cdot \tilde{x}_n + \tilde{y}_2 \cdot \tilde{x}_{n-1} + \cdots + \tilde{y}_{n-k+1} \cdot \tilde{x}_k + \alpha'' = \alpha$;

$\alpha = \alpha'' - \alpha'$; $\tilde{z}_n = \alpha'$.

Так мы получили последние $n$ членов результата $Z = X \cdot Y$.

Шаг $n+1$ $\tilde{y}_2 \cdot \tilde{x}_n + \tilde{y}_3 \cdot \tilde{x}_{n-1} + \cdots + \tilde{y}_j \cdot \tilde{x}_{n+2-j} + \cdots$

$$\ldots + \bar{y}_k \cdot \tilde{x}_{n+2-k} + \alpha'' = \alpha;$$

$$\alpha = \alpha'' - \alpha', \quad \tilde{z}_{n+1} = \alpha';$$

Шаг $n+k-1$ $\bar{y}_{k-1} \cdot \tilde{x}_n + \tilde{y}_k \cdot \tilde{x}_{n-1} + \alpha'' = \alpha$; $\alpha = \alpha'' - \alpha'$;

$$\tilde{z}_{n+k-1} = \alpha';$$

Шаг $n+k$ $\tilde{y}_k \cdot \bar{x}_n + \alpha'' = \alpha$; $\alpha = \alpha'' - \alpha'$; $\tilde{z}_{n+k} = \alpha'$; $\bar{z}_{n+k+1} = \alpha''$;

Итак получим

$$Z = X \cdot Y = \sum_j \tilde{z}_j \cdot 10^{7(j-1)} = \sum_i z_i \cdot 10^{14(i-1)}.$$

Вход — «сверхчисла» A1, A2, числа K1, K2.

Выход — «сверхчисло» $Z = \mathrm{A1} \cdot \mathrm{A2}$.

Вызов — CALL PROG1 (A1, A2, K1, K2, $Z$). Где параметры K1 и K2 объявлены как FIXED(2).

## ПОДПРОГРАММА PVK1

Подпрограмма PVK1 написана как отдельная процедура и предназначена для вычисления старшего разряда при вычислении корня из «сверхчисла».

Вход — $T$. Выход — $C$, $A$, $B$. Где $C = \max\{x/x^2 < T\}$. $B = T - C^2$.

Вызов — CALL PVK1($T$, $C$, $A$, $B$). Формальные параметры предварительно объявлены так: C FIXED(1), ($T$, $A$, $B$) FIXED(2).

Блок — схема PVK1 показана на рис. 3.

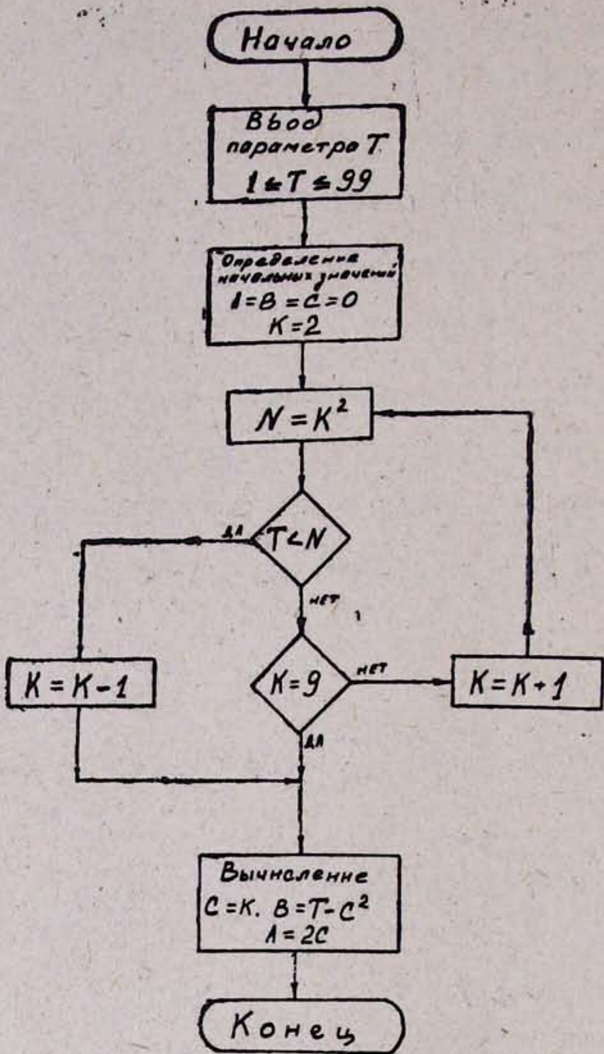


Рис. 3. Блок-схема PVK1

## ПОДПРОГРАММА PVK2

Подпрограмма PVK2 написана как отдельная процедура и предназначена для вычисления младших (не самого левого) разрядов результата вычисления корня из «сверхчисла». Кроме очередного разряда PVK2 вычисляет также удвоенный результат и промежуточная разница на каждом шаге.

Вход — «сверхчисла» $A$, $B$. Выход — $A$, $B$, $C$.

Где $C = \max\{k/k\cdot(10\cdot A + k) \leqslant B\}$.

Вызов — CALL PVK2($A$, $B$, $C$).

Блок-схема PVK2 показана на рис. 4.

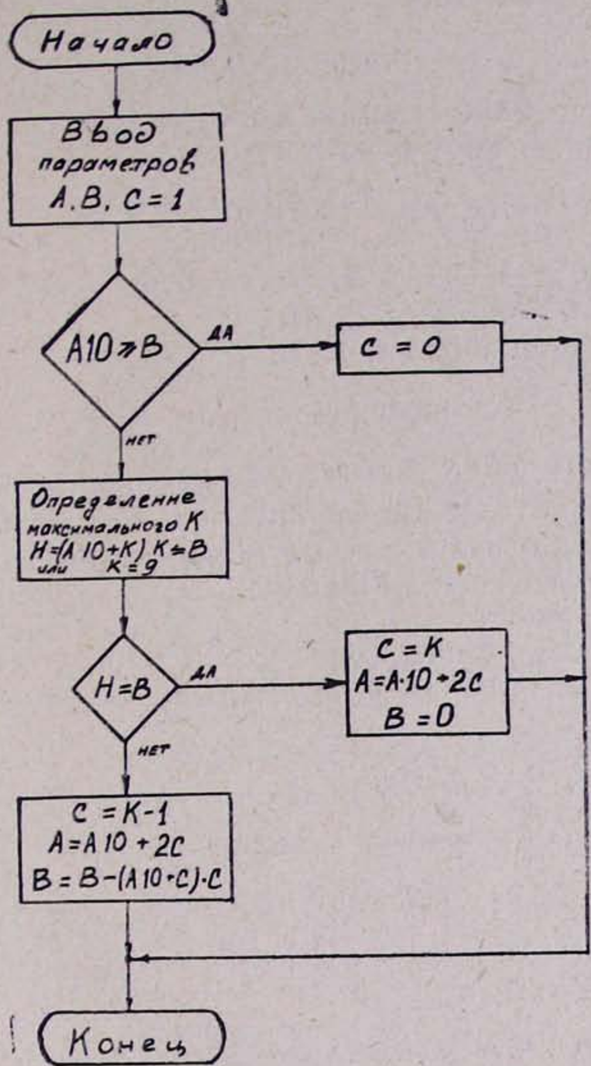


Рис. 4. Блок-схема PVK2

ВЫЧИСЛЕНИЯ КОРНЯ ИЗ «СВЕРХЧИСЛА»

Программа PGKS предназначена для вычисления корня из «сверхчисла». Сама программа имеет две подпрограммы — PVK1, PVK2 и реализует известный из школьного курса алгоритм вычисления корня из натурального числа, в частности из «сверхчисла».

«Сверхчисло» $X$ разделяется на пары.

Для самой левой пары — $T$, работает подпрограмма PVK1, которая вычисляет $C$, $A$, $B$, а для остальных пар вычисление продолжается уже по подпрограмме PVK2. Результат каждого шага запоминается и в конце получается новое «сверхчисло» $X = [\sqrt{X}]$.

Вход — «сверхчисло» $X$. Выход — «сверхчисло» $X = [\sqrt{X}]$.

Вызов — CALL PGKS ($X$).

### ПЕРЕЧИСЛЕНИЕ ПАР

Программа PKPP отдельная процедура и предназначена для вычисления номеров пар по формуле

$$C(x, y) = \frac{(x+y)(x+y+1)}{2} + x,$$

где как $x$, $y$, так и $C(x, y)$ есть «сверхчисла».

Вход — $X$, $Y$. Выход — $C(X, Y)$.

Вызов — CALL PKPP ($X$, $Y$, $C$).

### ПЕРЕЧИСЛЕНИЕ НАБОРОВ

Программа PKPN предназначена для кодирования наборов $Z = C^{n+1}(\alpha_1, \alpha_2, \ldots, \alpha_n, 0)$, где как $Z$, так и $\alpha_i$ «сверхчисла».

Вход — двумерный массив $A$ с размерами $(20\times 50)$, где каждый элемент массива есть число FIXED (15).

Выход — «сверхчисло» $Z$.

Вызов — CALL PKPN ($A$, $Z$).

Программа PKPN реализует известный алгоритм кодирования $n$-ок по формуле

$$C^{n+1}(x_1, x_2, \ldots, x_{n+1}) = C(x_1, C(x_2, C \cdots C(x_{n-1}, C(x_n, x_{n+1})) \cdots ),$$

где $C(x, y)$ есть Канторовская нумерация пар.

### КОДИРОВАНИЕ НАБОРОВ

Программа PGKN предназначена для кодирования пар (действие, набор) по формуле

$$K(d, \alpha_1, \alpha_2, \ldots, \alpha_n) = C_0 + d + a \cdot c^{n+1}(\alpha_1, \ldots, \alpha_n 0),$$

где $C_0 = 30$ (константа), $d \leqslant 5$ — код действия, $a = 6$ — количество базисных действий, $\alpha_1, \alpha_2, \ldots, \alpha_n$ — «сверхчисла»,

$$c^{n+1}(\alpha_1, \alpha_2, \ldots, \alpha_n 0) = c(\alpha_1, c, \ldots, c(\alpha_n, 0))$$

есть Канторовское перечисление наборов.

Вход — $d, \alpha_1, \alpha_2, \ldots, \alpha_n$. Выход — «сверхчисло» $Z$.

Вызов — CALL PGKN ($D$, $A$, $Z$).

### ДЕКОДИРОВАНИЕ

Процедура PGDN предназначена для декодирования «сверхчисла». Вход — «сверхчисло» $E$. Выход — код действия $d$, набор «сверхчисел» $\alpha_1, \alpha_2, \ldots \alpha_n$.

Вызов — CALL PGDN($E$, $D$, $A$), где $E$ — «сверхчисло»; $D$ FIXED(2); $A$ (20,50) FIXED(15).

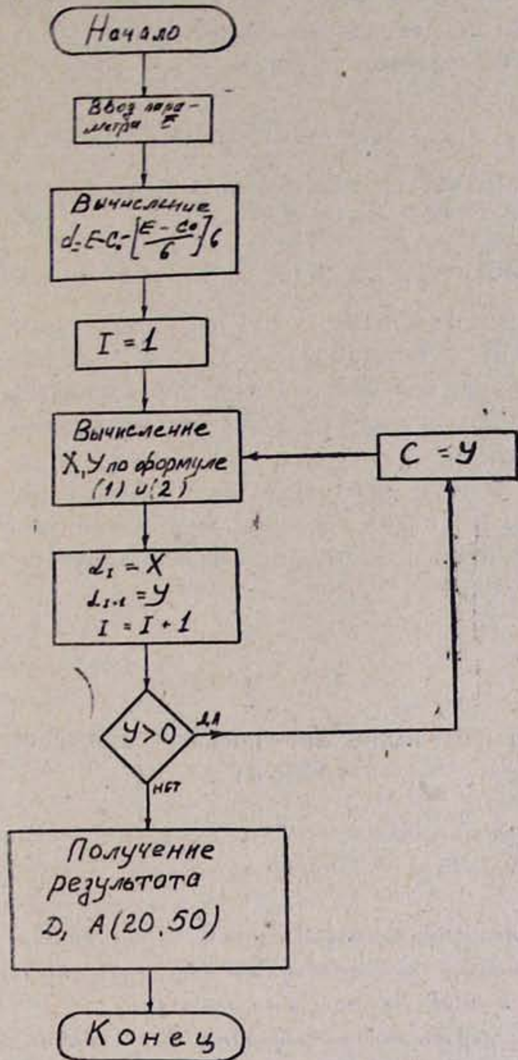


Рис. 5. Блок-схема PGDN

Если $K(d, \alpha_1, \alpha_2, \dots, \alpha_n) = E = c_0 + d + a\cdot c^{n+1}(\alpha_1, \alpha_2 \dots \alpha_n\ 0)$, то

$$d = E \dot{-} c_0 \dot{-} \frac{[E \dot{-} c_0]}{a}\cdot a;$$

$$c^{n+1}(\alpha_1, a_2, \dots, \alpha_n, 0) = \left[\frac{E \dot{-} d \dot{-} c_0}{a}\right] = n = c(X, Y); \quad i = 1$$

(1) $$X = n \dot{-} \frac{1}{2}\left[\frac{[\sqrt{8n+1}]+1}{2}\right]\cdot\left[\frac{[\sqrt{8n+1}]+1}{2} \dot{-} 1\right] =$$

$$= n \dot{-} \frac{1}{2}\cdot\alpha\cdot(\alpha \dot{-} 1);$$

(2) $Y = \alpha \dot{-} X \dot{-} 1 = \alpha \dot{-} 1 \dot{-} X; \quad \alpha_i = X; \quad \alpha_{i+1} = Y; \quad i = i + 1,$

если $y > 0$, то повторить цикл, уже при $n = y = \alpha_{i+1}$;

Этим алгоритмом мы декодируем «сверхчисло».

Блок-схема PGDN показана на рис. 5.

## ВЫЧИСЛЕНИЕ ФУНКЦИИ h

Программа PGVH предназначена для вычисления функции $h$, по формуле

$$h(q) = K(0,\ 0,\ K(0,\ 5,\ K(3 < q,\ q >, 0))).$$

Вход — $q$. Выход — $h(q)$.

Вызов — CALL PGVH ($Q$).

Функция $h$ имеет большое значение при вычислении «неподвижной» точки, и тесно связана как с базисными функциями, так и с универсальной функцией.

Программа PGVH реализует вычисление $h$ по данной формуле. Так что, при замене кодов базисных функции и универсальной функции, должна меняться и формула вычисления функции $h$, следовательно и программа PGVH.

Ա. Ա. ՊԵՏՐՈՍՅԱՆ

## ՀԻՄՆԱԿԱՆ ԲԱԶԻՍԱՅԻՆ ՖՈՒՆԿՑԻԱՆԵՐ՝ ԾՐԱԳՐԵՐԻ ԱՎՏՈՄԱՏ ՍԻՆԹԵԶՄԱՆ ՀԱՄԱՐ

*Այս աշխատանքում դիտարկվում է ծրագրերի սինթեզման խնդրի լուծման եղանակներից մեկը, հիմնվելով ռեկուրսիվ ֆունկցիաների տեսության մեթոդների վրա։*

*Առաջին պարագրաֆում ապացուցվում է, որ ռեկուրսիվ ֆունկցիաների ցանկացած թույլատրելի համարակալման մեջ* $\{\varphi_i\}$, *կարելի է գտնել ոչ բացահայտ տեսքով տրված հավասարման լուծումը։*

*Ապացուցման ընթացքում օգտագործում ենք «գերթվերի» կոդավորումը և դեկոդավորումը, ինչպես նաև լրացուցիչ* $f$ *և* $h$ *ֆունկցիաները։*

*Տրվում է հատուկ համարակալում* $\{\varphi_i\}$, *որը բաղկացած է բազիսային* $\{\varphi_i / i \leqslant 30\}$ *ֆունկցիաներից և փակ է այն վեց օպերացիաների նկատմամբ, որոնք սահմանվում են աշխատանքում։*

*Երկրորդ պարագրաֆում քննարկվում են այն տարրական գործողությունները, որոնց միջոցով իրագործվում են* $f$, $h$ *ֆունկցիաները, կոդավորումը։ Այս պարագրաֆում նկարագրում են նաև այդ գործողությունները ԵՍ-1045 էՀՄ-ի վրա իրացնելու բլոկ-սխեմաները։*