*Publ. RIMS, Kyoto Univ.*
**11** (1976), 297-337

# Formula Manipulations Solving Linear Ordinary Differential Equations (II)

By

Shunro WATANABE*

## §1. Introduction

Let us consider linear ordinary differential equations of 2nd order with analytic function coefficients of the form

$$(1.1) \qquad \frac{d^2y}{dx^2}+F(x)\frac{dy}{dx}+G(x)y=0\,.$$

This equation has been studied by many famous mathematicians, for example Euler, Gauss, Kummer, Fuchs, Riemann, Schwarz, etc. mostly in the 19th century. One of the main themes of them was related to the integration method of (1.1), namely they seek the criterion whether the general solution of (1.1) is representable by well known functions using some elementary transformations with respect to the independent and dependent variables. Their results were mostly concerned with the equations (1.2) which have only three regular singular points, $a_1$, $a_2$, $\infty$ ([9], [10]).

$$(1.2) \qquad \frac{d^2y}{dx^2}+\frac{Ax+B}{(x-a_1)(x-a_2)}\frac{dy}{dx}+\frac{Cx^2+Dx+E}{(x-a_1)^2(x-a_2)^2}y=0$$

Schwarz, Klein, Cayley and others calculated the representations when the general solution of (1.2) is represented by algebraic functions ([7], [8]).

In 1941 Hukuhara applied this idea to the hypergeometric equations of confluent type

Communicated by S. Takasu, June 4, 1974.
* Department of Mathematics, Tsuda College, Tokyo.

$$\frac{d^2y}{dx^2} + \frac{Ax+B}{x}\frac{dy}{dx} + \frac{Cx^2+Dx+E}{x^2}y=0, \tag{1.3}$$

and using notations similar to Riemann's $P$-functions, he obtained his results similar to that of (1.2). Also he tried to remove an apparent regular singular point by proper transformations, and to reduce the equations to the equations of (1.2) or (1.3) ([1]). In 1949, and 1952 Hukuhara and Ohashi determined all the type of the equation (1.2) whose general solution is represented by known functions, and all their representations ([3], [4]).

In section 2 we describe an algorithm which calculates the integration from the equation (1.1). This algorithm consists of four parts: (1) the calculations of the local informations such as the form of power series expansion at a singular point, (2) the transformations of the equation by changes of variables, (3) the calculations of the general solution of (1.2) and (1.3), (4) our integration method with a strategy using the above (1), (2) and (3), mostly consisting of removal methods of apparent singular points. This algorithm uses the many results of the mathematicians described above.

The purpose of this paper is to report the experiment which implements the above algorithm on a computer and executes this program. We intended to achieve two objectives by this experiment, one is to prepare an automatic solver of differential equations of a certain range, and the other is to get some suggestions to the tools for implementing such algorithm.

To implement such algorithm, it is usually regarded to be unavoidable to use programming language of list processing. But considering all the types of formulas which appear in the process of the execution of our program, it turned out that we can represent all the formulas by array structures. For example, $(2+3\sqrt{-1})/4$ and $3x^2-4x+5$ are translated to $(2, 3, -1, 4)$ and $(3, -4, 5)$ respectively. Therefore for our case, we can write this program by FORTRAN. We must remark that this program demands relatively small memories and the execution of this program is fast and it is executable in many computers. In section 3 we show some remarks to translate a formula to an array, and in Appendix 2, 3, 4 we show a typical subprogram of FORTRAN and some examples of calculation by computer.

A general formula manipulation contains some formulas whose type cannot be guessed before execution. Where by the term 'type', we mean a pattern of a formula. For such a case we must adopt some list processing facilities. But it is advantageous to restrict interpretation in execution to a small range as far as possible. For developing this idea we must lay the foundation to our formula manipulations. In section 4 we offer a programming language $L$ for formula manipulations. The language $L$ is constructed on the basic facilities of ALGOL–60 and the $L$ has the defining facilities of variables, types, and operations. For example we can define types so that $(2+3\sqrt{-1})/4$ and $(-3+8\sqrt{-1})/2$ have the same type, or the different types. By the $L$ we can write the program which fulfils the pattern recognitions and pattern matching of formulas, and the automatic activations of operators which depends on the sets of operation declarations. Let us consider the following example,

$$\textbf{letter}\ v;\ \textbf{integer}\ a, b;\ldots; v: =a+b;$$

where **letter** is a name of a type which could accept any formula. In our language L the value of $v$ after the execution of $v: =a+b$ is interpreted as follows; if the values of $a$ and $b$ are 5, $-2$ respectively then the value of $v$ is 3, but if no integer has been assigned to $a$ and $b$ then the value of $v$ is the character string $a+b$ itself. This interpretation of a formula leads us to the view point from which the applications of rules that are usually called as axioms and theorems, are treated naturally. The general treatment of axioms and theorems in formula manipulations are very complicated and difficult, and it seems almost impossible to implement such a programming system from the practical viewpoint. But what we want is restricted to the field of mathematics which could treat polynomials and rational functions of a few variables and their elementary operations and so on. By these manipulations, many mathematical fields, containing differential and integral calculus, could be treated by computers. Generally speaking, a set of rules needs one evaluation algorithm of formulas. But only one evaluation algorithm is sufficient for our purpose. Our algorithm is characterized by the facilities that can treat the following rules: if $o$ is $+$ or $\cdot$ then

$$aob = boa; \qquad (aob)oc = ao(boc),$$

therefore in our language $L$ we can define other rules which are consistent with the above two rules ([13]~[20]).

## §2. The Integration Algorithm of the Equation (1.1)

**begin** MAIN program;

$j: =0; \qquad k: =0;$

where $j$ is a suffix of dependent variable $y$ and $k$ is a suffix of independent variable $x$.

Read $n$, $E$;

where $n$ is a problem number, $E$ is an equation of the form

$$(E) \qquad y'' + F(x)y' + G(x)y = 0,$$

$$F(x) = q(x)/p(x), \qquad G(x) = r(x)/p(x)^2,$$

where $p(x)$, $q(x)$, and $r(x)$ are polynomials with integer coefficients, hereafter we use $x$, $y$ in place of $x_k$, $y_j$.

1: Determine all of the singular points of $E$; we describe them by $\alpha_1, \alpha_2, \ldots, \alpha_s, \infty$, where $\alpha_i$'s are obtained by factoring the polynomial $p(x)$ within integer coefficients. We suppose that the degree of the factor of $p(x)$ is less than or equal to two. Thus each $\alpha_i$ is a regular singular point of $E$ and it has one of the following forms.

$$(S) \qquad a, \quad a/d, \quad (a + b\sqrt{c})/d,$$

where $a$, $b$, $c$, and $d$ are integers.

2: Print the equation $E$ and its singular points $\alpha_1, \ldots, \alpha_s, \infty$.

$i: =1$; $n: =0$; $i$ is the suffix of $\alpha$, $n$ is the number of the apparent singular points. If $s=0$ then go to 7;, where $s$ is the number of singular points except infinity.

3: $F_i(x): =(x-\alpha_i)F(x); \qquad G_i(x): =(x-\alpha_i)^2 G(x);$

$f_i(x, \lambda): =\lambda^2 + (F_i(x) - 1)\lambda + G_i(x); \qquad f_{i0}(\lambda): =f_i(\alpha_i, \lambda);$

where $f_{i0}(\lambda)$ is the characteristic equation of $E$ at $x=\alpha_i$, and $f_{i0}(\lambda)$ is the polynomial of $\lambda$ whose coefficients have the form of $a+b\sqrt{c}$.

Calculate two roots $\lambda_1$, $\lambda_2$ ($\mathrm{Re}\,\lambda_1 \geqq \mathrm{Re}\,\lambda_2$) of $f_{i0}(\lambda)=0$;

Print $f_{i0}(\lambda)$ and its factors;

If $\lambda_1-\lambda_2$ is not integer or zero then go to 5;

Note that if we put $f_{i0}(\lambda)=\alpha\lambda^2+\beta\lambda+\gamma$ then $\lambda_1-\lambda_2=\sqrt{\beta^2-4\alpha\gamma}/\alpha$, where $\alpha$, $\beta$, $\gamma$ have the form of $a+b\sqrt{c}$.

$m:=\lambda_1-\lambda_2$; $g_0:=1$;

Calculate and print $f_1(\lambda)$, $g_1(\lambda),\dots, f_{m-1}(\lambda)$, $g_{m-1}(\lambda)$, $f_m(\lambda)$, $g_r(\lambda)$; where they are defined as follows.

$$f_n(\lambda)=\frac{1}{n!}\frac{\partial^n}{\partial x^n}f_i(x,\lambda)\Big|_{x=\alpha_i},$$

$$g_{nr}(\lambda)=f_1(\lambda+n-1)g_{n-1}(\lambda)+\cdots+f_n(\lambda)g_0(\lambda),$$

$$g_n(\lambda)=-f_0(\lambda+n)^{-1}g_{nr}(\lambda), \qquad g_r(\lambda)=g_{nr}(\lambda),$$

where $g_n(\lambda)$ is the coefficient of formal power series solution $y$ of $E$:

$$y=\sum_{n=0}^{\infty} g_n(\lambda)(x-\alpha_n)^{\lambda+m},$$

and each $g_n(\lambda)$ is a rational function of $\lambda$ whose coefficients are of the form $a+b\sqrt{c}$.

If $g_r(\lambda)$ is not zero then go to 5;

$l_i:=0$; $n:=n+1$;

where $l_i$ is the flag which holds the information whether $\alpha_i$ is apparent or not, if $l_i$ is 0 then apparent.

If $\lambda_2$ is not equal zero then go to 4;

Find the first non zero element $g_{\mu i}(\lambda)$ out of $g_1(\lambda)$, $g_2(\lambda),\dots, g_{m-1}(\lambda)$;, therefore $2\leqq\mu_i\leqq m-1$, $\mu_i$ might be used when we try to remove $\alpha_i$. Print $\mu_i$;

4: Print '$\alpha_i$ is apparent', go to 6;

5: $l_i:=1$; Print '$\alpha_i$ is not apparent';

6: $i:=i+1$; If $i\leqq s$ then go to 3;

7: Calculate and print the rank $r+1$ of $E$ at $\infty$; where

$$r=\max(\deg(F), \deg(G)/2). \qquad (\deg=\text{degree})$$

If $r+1\leqq 0$ then calculate and print the characteristic equation of $E$ at $\infty$; if we put $F_0(x)=xF(x)$, $G_0(x)=x^2G(x)$, then it is $f_{\infty 0}(\lambda)\equiv\lambda(\lambda+1)-F_0(\infty)\lambda+G_0(\infty)=0$.

If $s<3$ then go to 11 else if $n=0$ then go to 12;

$i:=1$;

8: If $l_i=1$ then go to 9 else if $\lambda_2=0$ then go to 10;

Substitute $(x-\alpha_i)^{\lambda_2}y_{j+1}$ for $y_j$ of $E$, and rearrange $E$ to the form of $E$, and rename it as $E1$;

$$(E1) \qquad y''+F1(x)y'+G1(x)y=0,$$

where $F1(x)=q1(x)/p1(x)$, and $G1(x)=r1(x)/p1(x)^2$.

If $\deg(p1)\geqq\deg(p)$ then go to 9 else the transformation succeeded. Print $y_{j+1}:=(x-\alpha_i)^{-\lambda_2}y_j$, and print the equation $E1$;

$$E:=E1;\ j:=j+1;\ n:=n-1;\ s:=s-1;$$

By this transformation $\alpha_i$ is removed from the singular points. This fact is based on the theorem which asserts that if $x=\alpha_i$ is the apparent singular point of $E$ and the exponents $\lambda_1, \lambda_2$ are 1, 0 respectively then $x=\alpha_i$ is the regular point of $E$. In this case $E$ have two solutions of the form

$$y_k=(x-\alpha_i)^{\lambda_k}\{g_{k0}+g_{k1}(x-\alpha_i)+g_{k2}(x-\alpha)^2+\cdots\}, \quad k=1, 2.$$

9: $i:=i+1$; if $i\leqq n$ then go to 8 else go to 12.

10: $v:=\min(\lambda_1, \mu_1)$. Deform $E$ as follows

$$G(x)^{-1}y''+F(x)G(x)^{-1}y'+y=0,$$

and differentiate this equation by $x$, then substitute $(x-\alpha_i)^{v-1}y_{j+1}$ for $y'_j$ and rearrange this equation to the form of $E1$.

If $\deg(p1)\geqq\deg(p)$ then go to 9 else this transformation succeeded. Print $y_{j+1}:=(x-\alpha_i)^{-v+1}y'_j$; $j:=j+1$; go to 1. This transformation based on the following inference: At $x=\alpha_i$ $E$ has two power series solutions of the form,

$$y=(x-\alpha_i)^{\lambda_1}\{g_{10}+g_{11}(x-\alpha_i)^1+g_{12}(x-\alpha_i)^2+\cdots\}, \qquad g_{10}\neq 0,$$

$$y=g_{20}+g_{2\mu_i}(x-\alpha_i)^{\mu_i}+g_{2\mu_i+1}(x-\alpha_i)^{\mu_i+1}+\cdots, \qquad g_{20}\neq 0, \quad g_{2\mu_i}\neq 0.$$

From this fact we can guess that the equation which satisfies $y'$ has the following solutions at $x=\alpha_i$,

$$y'=(x-\alpha_i)^{\lambda_1-1}\{h_{10}+h_{11}(x-\alpha_i)^1+\cdots\}, \qquad h_{10}\neq 0,$$

$$y'=(x-\alpha_i)^{\mu_i-1}\{h_{20}+h_{21}(x-\alpha_i)^1+\cdots\}, \qquad h_{20}\neq 0.$$

11: If $s=2, 1, 0$ then we call subroutine SOLT3, SOLT2, and SOLT1 correspondingly; These subroutines determine whether the general solution of $E$ is representable by known functions or not, and if it is representable then ISOLT: $=1$, prints the solution else ISOLT: $=0$.

If ISOLT$=1$ then we exit from this main program;

12: Deform $E$ as follows,

$$(E2) \qquad x^2y''+xF_0(x)y'+G_0(x)y=0,$$

and if $F_0(x)$ and $G_0(x)$ are rational functions of $x^l$ ($l=2, 3$), then replace $x_k^l$ of $E2$ by $x_{k+1}$, we name the equation thus obtained as $El$. Print $x_{k+1}$: $=x_k^l$; go to 1;

**end** MAIN program;

**begin** SOLT3;

Input parameters of SOLT3 are singular points $\alpha_1, \alpha_2, \infty$ of $E$, and their characteristic equations $f_{10}(\lambda)=0, f_{20}(\mu)=0$, and $f_{\infty 0}(\nu)=0$, and the rank $r+1$ of $\infty$.

If $r+1>0$ then go to 19 else assign $\lambda, \mu, \nu$ as follows.

$$\lambda: =\lambda_1-\lambda_2, \text{ where } \operatorname{Re}\lambda_1\geqq\operatorname{Re}\lambda_2, \qquad f_{10}(\lambda_i)=0, \ i=1, 2.$$

$$\mu: =\mu_1-\mu_2, \text{ where } \operatorname{Re}\mu_1\geqq\operatorname{Re}\mu_2, \qquad f_{20}(\mu_i)=0, \ i=1, 2.$$

$$\nu: =\nu_1-\nu_2, \text{ where } \operatorname{Re}\nu_1\geqq\operatorname{Re}\nu_2, \qquad f_{\infty 0}(\nu_i)=0, \ i=1, 2.$$

In this case the equation which has the singular points, and characteristic equations above mentioned is uniquely determined, and we denote the general solution of $E$ as

$$y=P\begin{Bmatrix}\alpha_1 & \alpha_2 & \infty & \\ \lambda_1 & \mu_1 & \nu_1 & x \\ \lambda_2 & \mu_2 & \nu_2 & \end{Bmatrix},$$

and we call this as Riemann's $P$-function.

If none of $\lambda+\mu+\nu$, $\lambda+\mu-\nu$, $\lambda-\mu+\nu$, $\lambda-\mu-\nu$ is an odd integer then go to 18 else if none of $\lambda+\mu+\nu$, $\lambda+\mu-\nu$, $\lambda-\mu+\nu$, $\lambda-\mu-\nu$ is 1 or $-1$ then go to 15;

Rewrite suffix so that $\lambda_2+\mu_2+\nu_2=0$, and we define

$\lambda:=\lambda_1-\lambda_2$, $\mu:=\mu_1-\mu_2$, $\nu:=\nu_1-\nu_2$;

Print $y=(x-\alpha_1)^{\lambda_2}(x-\alpha_2)^{\mu_2}$; This based on the formula

$$y=P\begin{Bmatrix}\alpha_1 & \alpha_2 & \infty & \\ \lambda_1 & \mu_1 & \nu_1 & x \\ \lambda_2 & \mu_2 & \nu_2 & \end{Bmatrix}=(x-\alpha_1)^{\lambda_2}(x-\alpha_2)^{\mu_2}P\begin{Bmatrix}\alpha_1 & \alpha_2 & \infty & \\ 0 & 0 & \lambda_2+\mu_2+\nu_2 & x \\ \lambda & \mu & \nu & \end{Bmatrix},$$

where $\lambda_2+\mu_2+\nu_2=0$.

If $\lambda\neq 1$ and $\mu\neq 1$ and $\nu\neq 1$ then go to 13;

If $\mu=1$ then we define $M(x)=-x+\alpha_1+\alpha_2$ else if $\nu=1$ then we define $M(x)=\alpha_1 x/(x-\alpha_1)$. Print $x_{k+1}:=M(x_k)$ and exchange $\lambda_i$, $\mu_i$, $\nu_i$ corresponding to this transformation; For example if $\mu=1$ then exchange $\lambda_i$ and $\mu_i$. Thus the new $\lambda$ equals 1.

If $\mu=0$ then print '$A+B\log(x-\alpha_2)$' else print '$A+B(x-\alpha_2)^{\mu}$'; This based on the following formulas.

$$P\begin{Bmatrix}\alpha_1 & \alpha_2 & \infty & \\ 0 & 0 & 0 & x \\ 1 & \mu & -\mu & \end{Bmatrix}=A+B(x-\alpha_2)^{\mu},\ \mu\neq 0,$$

$$P\begin{Bmatrix}\alpha_1 & \alpha_2 & \infty & \\ 0 & 0 & 0 & x \\ 1 & 0 & 0 & \end{Bmatrix}=A+B\log(x-\alpha_2).$$

Go to 14;

13: Print $\left\{A+B\int(x-\alpha_1)^{\lambda-1}(x-\alpha_2)^{\mu-1}dx\right\}$;

14: ISOLT: $=1$; we exit from SOLT3;

15: Exchange the signs of $\lambda$, $\mu$, $\nu$ so that $\lambda+\mu+\nu$ is an odd integer and rewrite the suffixes to satisfy the following conditions: $\lambda=\lambda_1-\lambda_2$, $\mu=\mu_1-\mu_2$, $\nu=\nu_1-\nu_2$;

16: $n:=(\lambda+\mu+\nu-1)/2$; If $n\geqq 0$ then we set $\lambda:=-\lambda$, $\mu:=-\mu$, $\nu:=-\nu$, $\lambda_0:=\lambda_1$, $\mu_0:=\mu_1$, go to 16;

$m:=-n$; If $\nu$ is not integer or $\nu<-m$ or $\nu>-1$ then go to 17;

$m:=m+\nu$; If $m=0$ then print

$$y=(x-\alpha_1)^{\lambda_0}(x-\alpha_2)^{\mu_0}\left\{A+B\int(x-\alpha_1)^{-\lambda-1}(x-\alpha_2)^{-\mu-1}dx\right\},$$

go to 14; If $m\neq 0$ then print

17: $$y=(x-\alpha_1)^{\lambda_0}(x-\alpha_2)^{\mu_0}D^{m-1}\left[(x-\alpha_1)^{\lambda+m-1}(x-\alpha_2)^{\mu+m-1}\left\{A+B\int(x-\alpha_1)^{-\lambda-m}(x-\alpha_2)^{-\mu-m}dx\right\}\right],$$

go to 14; This depends on the following formulas.

$$y=P\left\{\begin{matrix}\alpha_1 & \alpha_2 & \infty & \\ \lambda_1 & \mu_1 & \nu_1 & x \\ \lambda_2 & \mu_2 & \nu_2 & \end{matrix}\right\}=(x-\alpha_1)^{\lambda_0}(x-\alpha_2)^{\mu_0}P\left\{\begin{matrix}\alpha_1 & \alpha_2 & \infty & \\ 0 & 0 & m & x \\ \lambda & \mu & \nu+m & \end{matrix}\right\},$$

if $\nu+1\neq 0,\ -1,\ldots,\ -(m-1)+1$ then

$$P\left\{\begin{matrix}\alpha_1 & \alpha_2 & \infty & \\ 0 & 0 & m & x \\ \lambda & \mu & \nu+m & \end{matrix}\right\}=\frac{d^{m-1}}{dx^{m-1}}P\left\{\begin{matrix}\alpha_1 & \alpha_2 & \infty & \\ 0 & 0 & 1 & x \\ \lambda-m-1 & \mu+m-1 & \nu+1 & \end{matrix}\right\}.$$

18: Search $(\lambda, \mu, \nu)$ in the following Schwarz's table, if it is found then #: =the corresponding number else #: =99; From #1 to #15 is called as Schwarz's table.

| # | $\lambda$ | $\mu$ | $\nu$ | $k, m, l$=integer |
|---|---|---|---|---|
| 1 | $1/2+k$ | $1/2+m$ | $r$ | $r$=any complex number |
| 2 | $1/2+k$ | $1/3+m$ | $1/3+1$ | $k+m+l$=even integer |
| 3 | $2/3+k$ | $1/3+m$ | $1/3+1$ | |
| 4 | $1/2+k$ | $1/3+m$ | $1/4+1$ | $k+m+l$=even integer |
| 5 | $2/3+k$ | $1/4+m$ | $1/4+1$ | |
| 6 | $1/2+k$ | $1/3+m$ | $1/5+1$ | $k+m+l$=even integer |
| 7 | $2/5+k$ | $1/3+m$ | $1/3+1$ | 〃 |
| 8 | $2/3+k$ | $1/5+m$ | $1/5+1$ | 〃 |
| 9 | $1/2+k$ | $2/5+m$ | $1/5+1$ | 〃 |
| 10 | $3/5+k$ | $1/3+m$ | $1/5+1$ | 〃 |
| 11 | $2/5+k$ | $2/5+m$ | $2/5+1$ | 〃 |
| 12 | $2/3+k$ | $1/3+m$ | $1/5+1$ | 〃 |
| 13 | $4/5+k$ | $1/5+m$ | $1/5+1$ | 〃 |
| 14 | $1/2+k$ | $2/5+m$ | $1/3+1$ | 〃 |
| 15 | $3/5+k$ | $2/5+m$ | $1/3+1$ | 〃 |
| 16 | $1/2+k$ | $1/4+m$ | $1/4+1$ | 〃 |
| 17 | $1/2+k$ | $1/3+m$ | $1/6+1$ | 〃 |
| 18 | $0+k$ | $0+m$ | $0+1$ | 〃 |

where $\lambda$, $\mu$, $\nu$ have the form of $\sqrt{i+j\sqrt{k/m}}$, $i, j, k, m$ are integers.

If $\# > 18$ then go to 19;

Select the signs of $\lambda, \mu, \nu$ so that they satisfy the conditions: $\lambda = \lambda_0 - p$, $\mu = \mu_0 - q$, $\nu = \nu_0 - r$, where $0 < \lambda_0, \mu_0, \nu_0 < 1$ and $p$, $q$, and $r$ are zero or positive integers. Make transformation of independent variable $x_{k+1} = L_1(x_k)$ so that $\alpha_1, \alpha_2, \infty$ are mapped to 0, 1, $\infty$ respectively; where

$$y = P\begin{Bmatrix} 0 & 1 & \infty & \\ \lambda_1 & \mu_1 & \nu_1 & x \\ \lambda_2 & \mu_2 & \nu_2 & \end{Bmatrix}, \quad \lambda = \lambda_1 - \lambda_2,\ \mu = \mu_1 - \mu_2,\ \nu = \nu_1 - \nu_2.$$

Print $x_{k+1} = L_1(x_k)$, $k := k+1$; If $p+q+r$ is an odd integer and $\lambda_0 = 1/2$ then $q := q+1$, $sw := 1$ else $sw := 0$;

Make transformation $x_{k+1} = L_2(x_k)$ so that $p \geqq q \geqq r \geqq 0$, and $p$, $q$, and $r$ correspond to 0, 1, and $\infty$;

Print $x_{k+1}=L_2(x_k)$; $k:=k+1$;

Print $y_j=x^{\lambda_2}(x-1)^{\mu_2}y_{j+1}$; where

$$y_{j+1}=P\begin{Bmatrix} 0 & 1 & \infty & \\ 0 & 0 & \delta & x \\ \lambda & \mu & \nu+\delta & \end{Bmatrix}.$$

$j:=j+1$; If $sw=1$ then print $y_j=\sqrt{x}D_x y_{j+1}$; $j:=j+1$;
$m':=(p+q-r)/2$, $n':=(p-q+r)/2$, print the formula:
$y_j=(x-1)^{\mu-m'-r-n'}D_x^{n'}[(x-1)^{-\mu+r+m'}D_x^{m'}\{x^{-\xi-r}D_t^r(x^\xi y_{j+1})\}]$, $t=1/x$;
where

$$y_{j+1}=P\begin{Bmatrix} 0 & 1 & \infty & \\ 0 & 0 & \xi & x \\ \lambda_0 & \mu_0 & \nu_0+\xi & \end{Bmatrix}.$$

$j:=j+1$; Make transformation $x_{k+1}=M(x_k)$ so that $(\lambda_0, \mu_0, \nu_0)$ corresponds to the entry of Hukuhara-Ohashi's table (type $1\sim5$) or Schwarz' table (type $6\sim15$); Print $x_{k+1}=M(x_k)$; If $\infty$ is transformed to 0 or 1 by this transformation then print $y_j=x^\xi y_{j+1}$ or $y_j=(x-1)^\xi y_{j+1}$; We define $G$ by

$$y=P\begin{Bmatrix} 0 & 1 & \infty & \\ 0 & 0 & \eta & x \\ \lambda & \mu & \nu+\eta & \end{Bmatrix}=G(\lambda, \mu, \nu, \eta, x).$$

If type number of $(\lambda, \mu, \nu)$ is less or equal 5 then print the representations of $y$ which had been calculated by Hukuhara and Ohashi else print the algebraic function $z(x)$, which had been calculated by Schwarz, Klein, Brioschi, Cayley and others; We call the latter as Cayley's table. Print

$$y_{j+1}=(C_1\cdot z(x)+C_2)\{x^{\lambda-1}(1-x)^{\mu-1}/z'(x)\}^{1/2};$$

Go to 14;

Hukuhara-Ohashi's table

| | |
|---|---|
| 1 | $G(1/2,\ 1/2,\ v,\ v/2,\ x)=(\sqrt{x}\pm\sqrt{x-1})^{v}$ |
| 2 | $G(1/3,\ 1/2,\ 1/3,\ -1/12,x)=\left\{\sqrt{3}(t+1)\pm 2\sqrt{t^2+t+1}\right\}^{1/4},\ t^3=x$ |
| 3 | $G(1/3,\ 1/3,\ 2/3,\ -1/6,\ x)=\Big[\sqrt{3}\left\{4^{1/3}x^{1/3}(1-x)^{1/3}+1\right\}$ $\pm 2\sqrt{4^{2/3}x^{2/3}(1-x)^{2/3}+4^{1/3}x^{1/3}(1-x)^{1/3}+1}\Big]^{1/4}$ |
| 4 | $G(1/2,\ 1/3,\ 1/4,\ -1/24,\ x)=\Big[\sqrt{3}\left\{(t-1)^{1/3}+(t+1)^{1/3}\right\}$ $\pm 2\sqrt{(t-1)^{2/3}+(t^2-1)^{1/3}+(t+1)^{2/3}}\Big]^{1/4},\ t^2=x$ |
| 5 | $G(1/4,\ 1/4,\ 2/3,\ -1/12,\ x)=\Big[\sqrt{3}\left\{(\sqrt{x}-\sqrt{x-1})^{2/3}+(\sqrt{x}+\sqrt{x-1})^{2/3}\right\}$ $\pm 2\sqrt{(\sqrt{x}-\sqrt{x-1})^{4/3}+1+(\sqrt{x}+\sqrt{x-1})^{4/3}}\Big]^{1/4}$ |

Cayley's table (#1~#15)

We assume that $k=m=1=0$, and $1/v=n$.

| # | $x$ : | $x-1$ : | $1$ |
|---|---|---|---|
| 1 | $4z^n$ | $(z^n+1)^2$ | $(z^n-1)^2$ |
| 2 | $(z^4+2\sqrt{3}iz^2+1)$ | $12\sqrt{3}iz^2(z^4-1)^2$ | $-(z^4-2\sqrt{3}iz^2+1)^3$ |
| 3,5 7,8 | $4z$ | $(z+1)^2$ | $(z-1)^2$ |
| 4 | $(z^8+14z^4+1)^3$ | $(z^{12}-33z^8-33z^4+1)$ | $-108(z^5-z)^4$ |
| 6 | $(z^{20}-228z^{15}+494z^{10}+228z^5+1)$ | $(z^{30}-522z^{25}-10005z^{20}-10005z^{10}+522z^5+1)$ | $-1728(z^{11}+11z^6-z)^5$ |
| 9 | $(z-4)^3$ | $(z-1)(z+8)^2$ | $27z^2$ |
| 10 | $z(z+8)^3$ | $(z^2-20z-8)$ | $-64(z-1)^3$ |
| 11 | $4(z^2-z+1)^3$ | $(2z^3-3z^2-3z+2)^2$ | $-27z^2(z-1)^2$ |
| 12 | $z^3(z+5)^2(z+8)$ | $(z^3+9z^2+12z-8)$ | $-64(3z-1)$ |
| 13 | $(z^2+14z+1)^3$ | $(z^3-33z^2-33z+1)^2$ | $-108z(z-1)^4$ |
| 14 | $(64z+189)$ $(64z^2+133z+49)$ | $z(4096z^3+18816z^2+25725z+12005)^2$ | $-27\cdot 7^7(z+1)^2$ |
| 15 | $-(5z-27)\cdot(125z^3-25z^2-265z-243)^3$ | $(-3125z^5+9375z^4+18750z^3+8750z^2+30750z+19683)$ | $1382400000z^3\cdot(z+1)^2$ |

| | |
|---|---|
| 16 | $x=\mathfrak{P}(z)$, where $\mathfrak{P}(z)$ is Weierstrass' pe function. |
| 17 | $x=\mathfrak{P}'(z)^2$ |
| 18 | $x=\lambda(z)$, where $\lambda(z)$ is the elliptic modular function defined by $\lambda(z)=(\mathfrak{P}((1+z)/2)-\mathfrak{P}(z/2))/(\mathfrak{P}(1/2)-\mathfrak{P}(z/2))$. |

19: ISOLT: =0; Exit from SOLT3;

**end** SOLT3;

**begin** SOLT2;

Input parameters of SOLT2 are singular points $\alpha$, $\infty$ and the characteristic equations of $\alpha$ and $\infty$, and the rank $r+1$ of $\infty$. Our equation $E$ has the following form

$$(E3) \qquad y''+f(x)/(x-\alpha)\cdot y'+g(x)/(x-\alpha)^2\cdot y=0,$$

where $f(x)$ and $g(x)$ are polynomials, and $\alpha$ is a rational number.

If $r+1>0$ then go to 22;

Let $\sigma_1, \sigma_2$ be the two roots of the characteristic equation $\sigma(\sigma-1)+f(\alpha)\sigma+g(\alpha)=0$ at $\alpha$, where $f(x)$ and $g(x)$ are constants (rational numbers), and $E3$ is the equation of Euler type.

20: If $\sigma_1=\sigma_2$ then print $y_j=(x-\alpha)^\sigma\{A+B\cdot\log(x-\alpha)\}$ else print $y_j=A(x-\alpha)^{\sigma_1}+B(x-\alpha)^{\sigma_2}$;

21: ISOLT: =1; Exit from SOLT2;

22: If $r+1>1$ then go to 23; Print $x_{k+1}=x_k-\alpha$; $k$: $=k+1$;

The transformed equation has the following form,

$$(E4) \qquad y''+(ax+b)/x\cdot y'+(cx^2+dx+e)/x^2\cdot y=0,$$

where $a$, $b$, $c$, $d$, and $e$ are rational numbers.

Let $\sigma_1, \sigma_2$ be the two roots of $\sigma(\sigma-1)+b\sigma+e=0$, and let $\lambda_1, \lambda_2$ be the two roots of $\lambda^2+a\lambda+c=0$.

If $\lambda_1\neq\lambda_2$ ($a^2\neq 4c$) then go to 25 else $y_j$ is represented by Hukuhara's $P$-function of confluent type,

$$y=P\left\{\begin{matrix} \overbrace{\quad\infty^*\quad} & & 0 & \\ \lambda_1 & \beta & \sigma_1 & x \\ \lambda_2 & -\beta & \sigma_2 & \end{matrix}\right\}, \qquad \beta^2/4+(d+b\lambda_1)=0.$$

This represents the general solution of $E4$ under the condition $\lambda_1=\lambda_2$, and this has the asymptotic expansions at $\infty$ of the form

$$y_i:=\exp(\lambda_1 x+\beta_2\sqrt{x})x^{1/2}(c_{i0}+c_{i1}x^{-1/2}+c_{i2}x^{-1/2}+\cdots),\quad c_{i0}\neq 0,\qquad i=1,2,$$

where $\beta_1=\beta$, $\beta_2=-\beta$.

If $b\lambda_1+d=0$ then go to 24;

23: ISOLT: $=0$; Exit from SOLT2.

24: Print $y_j=\exp(\lambda_1 x)y_{j+1}$; This corresponds to the formulas

$$y_j=P\left\{\begin{matrix}\overbrace{\quad\infty^*\quad} & 0 & \\ \lambda_1 \quad 0 & \sigma_1 & x\\ \lambda_1 \quad 0 & \sigma_2 & \end{matrix}\right\}=\exp(\lambda_1 x)y_{j+1},\ \text{and}$$

$$y_{j+1}=P\left\{\begin{matrix}\overbrace{\quad\infty^*\quad} & 0 & \\ 0 \quad 0 & \sigma_1 & x\\ 0 \quad 0 & \sigma_2 & \end{matrix}\right\}=\begin{cases}Ax^{\sigma_1}+Bx^{\sigma_2}, & \sigma_1\neq\sigma_2\\ A+B\cdot\log x, & \sigma_1=\sigma_2\end{cases}.$$

$j:=j+1$; $\alpha:=0$; Go to 20;

25: $\mu_1:=(d+b\lambda_1)/(\lambda_1-\lambda_2)$; $\mu_2:=-(d+b\lambda_2)/(\lambda_1-\lambda_2)$;
In this case $y_j$ is represented by Hukuhara's $P$-function

$$y_j=P\left\{\begin{matrix}\overbrace{\quad\infty\quad} & 0 & \\ \lambda_1 \quad \mu_1 & \sigma_1 & x\\ \lambda_2 \quad \mu_2 & \sigma_2 & \end{matrix}\right\},\qquad \lambda_1\neq\lambda_2\,.$$

This represents the general solution of $E4$ under the condition $\lambda_1\neq\lambda_2$, and this has the asymptotic expansions at $\infty$ of the form

$$y_{ji}=\exp(\lambda_i x)x^{-\mu_i}\{c_{i0}+c_{i1}x^{-1}+c_{i2}x^{-2}+\cdots\},\qquad c_{i0}\neq 0,\quad i=1,2.$$

If none of $\mu_i+\sigma_j$ is an integer then go to 27;

Put on suffix to $\mu$ and $\sigma$ so that $\mu_1+\sigma_1$ is zero or a positive integer; Let $\lambda_i$ correspond to $\mu_i$;

If $\mu_1+\sigma_1=0$ then go to 26 else $n:=\mu_1+\sigma_1-1$; Print

$$y_j=e^{\lambda_1 x}\ x^{\sigma_1}\ D^n\left[e^{(\lambda_2-\lambda_1)x}\ x^{\sigma_2-\sigma_1+n}\left\{A+B\int e^{(\lambda_1-\lambda_2)x}x^{\sigma_1-\sigma_2-n-1}dx\right\}\right];$$

This based on the following theorem obtained by Hukuhara.

If the two $P$-functions

$$y=P\begin{Bmatrix} \overbrace{\infty}^{} & 0 & & \\ \lambda_1 & \mu_1 & \sigma_1 & x \\ \lambda_2 & \mu_2 & \sigma_2 & \end{Bmatrix},\qquad z=P\begin{Bmatrix} \overbrace{\infty}^{} & 0 & & \\ \lambda_1' & \mu_1' & \sigma_1' & x \\ \lambda_2' & \mu_2' & \sigma_2' & \end{Bmatrix}$$

satisfy (1) $\lambda_1-\lambda_2=\lambda_1'-\lambda_2'$ (2) $p=\mu_1'-\mu_1'+\sigma_1-\sigma_1$, and $q=\mu_2'-\mu_2+\sigma_1'-\sigma_1$ are integers, and if we select positive integers $m, m', n, n'$ so that $n'-n=p$, $m'-m=q$ then

$$z=e^{\lambda_2' x}x^{\sigma_1'}D^{m'}\left[e^{(\lambda_1-\lambda_2)x}D^{n'}\left[x^{\sigma_2-\sigma_1+m+n}D^m\left\{e^{(\lambda_2-\lambda_1)x}D^n\left(e^{-\lambda_2 x}x^{-\sigma_2}y\right)\right\}\right]\right],$$

where $D^k=d^k/dx^k$. Go to 21;

26: Print $y_j=e^{\lambda_1 x}x^{\sigma_1}\left\{A+B\int e^{(\lambda_2-\lambda_1)x}x^{\sigma_2-\sigma_1-1}dx\right\}$;

This use the following formula,

$$e^{\lambda x}x^{\sigma}P\begin{Bmatrix} \overbrace{\infty}^{} & 0 & & \\ \lambda_1 & \mu_1 & \sigma_1 & x \\ \lambda_2 & \mu_2 & \sigma_2 & \end{Bmatrix}=P\begin{Bmatrix} \overbrace{\infty}^{} & & 0 & \\ \lambda_1+\lambda & \mu_1-\sigma & \sigma_1+\sigma & x \\ \lambda_2+\lambda & \mu_2-\sigma & \sigma_2+\sigma & \end{Bmatrix}.$$

Go to 21;

27: If $\lambda_1\neq i$, or $\lambda_2\neq -i$ then go to 23;

If we can select $k$ so that both $p=\mu_1-1/2+\sigma_1-k$, and $q=\mu_2-1/2+\sigma_1-k$ are integers then we determine $k$ out of the possible $k$'s that gives the minimum of $|p|+|q|$ else go to 23;

If $p\leqq 0$ then $n':=0$ and $n:=-p$ else $n':=p$ and $n:=0$;

If $q\leqq 0$ then $m':=0$ and $m:=-q$ else $m':=q$ and $m:=0$;

Print the following formula and its comment,

$$y_j=e^{-ix}x^{\sigma}D^{m'}\left[e^{2ix}D^{n'}x^{m-n-2k}D^m\left[e^{-2ix}D^n\left[e^{ix}x^kB_k(x)\right]\right]\right],$$

where $B_k(x)$ is the general solution of the Bessel's equation of order $k$;

$$(B_k) \qquad y''+1/x\cdot y'+(x^2-k^2)/x^2\cdot y=0.$$

Go to 21;
**end** SOLT2;
**begin** SOLT1;
Input parameters of SOLT1 are the equation and the rank $r+1$ at $\infty$.
If $r+1>1$ then ISOLT: $=0$ and exit from SOLT1;
If $r+1\leqq 1$ then our equation is $y''+ay'+by=0$.
Calculate the two roots $r_1, r_2$ of the characteristic equation $r^2+ar+b=0$;
If $a^2=4b$ then print $y_j=e^{r_1x}(A+Bx)$ else print $y_j=Ae^{r_1x}+Be^{r_2x}$;
ISOLT: $=1$; Exit from SOLT1;
**end** SOLT1.

## §3. The Implementation of Our Algorithm by FORTRAN

Our algorithm is a typical example of formula manipulations, therefore it is natural to describe this algorithm by a list processing programming language. But from the practical view point, it is preferable to use the most usual programming language as FORTRAN. We decomposed this algorithm to about 60 subroutines, and each subroutine maps a set of integers to another set of integers. For realizing these decompositions, we must calculate all the formulas which might appear in the computation of the algorithm, before its execution. For example, after the substitution of $(x-\alpha_i)^{\lambda_2}y_{j+1}$ for $y_j$ of $E$, we must calculate the representation of $F1(x)$ and $G1(x)$ by $\alpha_i$, $\lambda_2$, $F(x)$ and $G(x)$. Similarly after the substitution of $(x-\alpha_i)^{\nu-1}y_{j+1}$ for $y'_j$ of $E$, we must calculate the representation of $F1(x)$ and $G1(x)$ by $\alpha_i$, $\nu$, $F(x)$ and $G(x)$. Thus if the number of formulas which must be calculated before computation is finite, and if the algorithm which is indicated by these formulas can be translate to the procedure whose input and output parameters are integer arrays, then we can describe this algorithm by FORTRAN. These are not always possible and these reducing calculations may be complicated or long enough to demand a computer. For these cases, if we would like to describe those algorithms in a program then we must use the programming language which has the facilities of list processing and the facilities that use the formulas obtained by computations as parts of the program by automatic criteria. In section 4 we shall offer a programming language $L$ with the facilities above mentioned.

To explain our FORTRAN programming, consider the algorithm which determines whether a regular singular point is apparent or not. It is sufficient to show some of the data structures, i.e. integer arrays, and some of the specifications of the subroutines that are used in the program.

To simplify our explanation, we assume that $F(x)$ and $G(x)$ are rational functions with no parameters. But the assumption that the coefficients of $F(x)$ and $G(x)$ are integers, is essential. First explain our basic subroutines.

Polynomials of the form (3–1) are represented by the integer array (3–1′) of length 20 as follows.

(3–1) $$a_0+a_1x+a_2x^2+\cdots+a_nx^n, \quad a_i=\text{integer}$$

(3–1′)

| | | | | | | |
|---|---|---|---|---|---|---|
| $n+2$ | $a_0$ | $a_1$ | $a_2$ | $\cdots$ | $a_n$ | |

20

Rational functions of the form (3–2) are represented by the integer array (3–2′) of length 43 as follows.

(3–2) $$\frac{a(a_0+a_1x+a_2x^2+\cdots+a_nx^n)}{b(b_0+b_1x+b_2x^2+\cdots+b_mx^m)^k},$$

where $a$, $b$, $a_i$, $b_j$, and $k$ are integers.

(3–2′)

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $a$ | $n+2$ | $a_0$ | $a_1$ | $a_2$ | $\cdots$ | $a_n$ | | |
| $b$ | $m+2$ | $b_0$ | $b_1$ | $b_2$ | $\cdots$ | $b_m$ | | $k$ |

21

The specifications of subroutines which correspond to the operations of the polynomials and rational functions are as follows. First we explain the kinds of parameters.

| | | |
|---|---|---|
| $I, J, K, L, N$ | : | integers, $NR$: rational numbers, |
| $A, B, C, P, Q, R$ | : | polynomials, |
| $A1, B1, C1$ | : | rational functions whose $k$ part is 1, |
| $A2$ | : | rational functions. |

**Subroutines table**

| | | |
|---|---|---|
| $IQ(I, J, K)$ | : | $K: = G.C.M.$ of $I$ and $J$; $I: = I/K$; $J: = J/K$; |
| $PPA(A, B, C)$ | : | $C: = A + B$; |
| $PPB(A, B, C)$ | : | $C: = A - B$; |
| $PPM(A, B, C)$ | : | $C: = A \cdot B$; |
| $PPD(A, B, Q, R, N)$ | : | $Q: =$ the quotient of $N \cdot A$ divided by $B$; $R: =$ the remainder, $N \cdot A = B \cdot Q + R$; $N: = b^n$, and $n = \deg(A) - \deg(B) + 1$, |
| $PSP(A, B, C)$ | : | $C: =$ the result of substitution of $B$ for $x$ of $A$; |
| $PPDF(A, B)$ | : | $B: = dA/dx$; |
| $PPSI(A, I, NR)$ | : | $NR: =$ the result of substitution of $I$ for $x$ of $A$; |
| $PCF(A, L)$ | : | $L: =$ the G. C. M. of all $A(i)$; $A(i): = A(i)/L$; |
| $PF1(A, P, Q, K)$ | : | $P \cdot Q$ is the result of factorization of $A$; $\deg(A) = 2$, $K: =$ if reducible then 1 else 0; |
| $RA(A1, B1, C1)$ | : | $C1: = A1 + B1$; |
| $RCH(C1)$ | : | if the numerator of $C1$ is 0 then the denominator of $C1$ is 1; |
| $RMOD(A1, B)$ | : | the numerator and the denominator of $A1$ are replaced by the remainders divided by $B$; |
| $RM(A1, B1, C1)$ | : | $C1: = A1 \cdot B1$; |
| $RDF(A2)$ | : | $A2: = dA2/dx$; |
| $RSI(A2, I, NR)$ | : | $NR: =$ the result of substitution of $I$ for $x$ of $A2$; $NR(1)$ is the numerator, $NR(2)$ is the denominator. |

With these subroutines we determine whether a regular singular point is apparent or not as follows. Input parameters are the equation $E$ and the singular points $\alpha_1, \dots, \alpha_s, \infty$.

$$(E) \qquad y'' + F(x)y' + G(x)y = 0\,.$$

$E$ is stored in the integer array $SA$, and singular points are stored in the integer array $T1$ as follows.

| | |
|---|---|
| $SA(*, 1)$ | $G(x)$ |
| $SA(*, 2)$ | $F(x)$ |

43

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $T1(*, 1)$ | $\alpha_1$ | $\omega_1$ | $c_1$ | $b_1$ | $a_1$ | 0 | $n_1$ | 4 | $a_0^1$ | $a_1^1$ | $a_2^1$ | 0 | $d_1$ | $k_1$ | $r_1^1$ | $r_2^1$ | $\mu_1$ | $1_1$ |
| ⋮ | | | | | | | | | | | | | | | | | | |
| $T1(*, 5)$ | $\alpha_5$ | $\omega_5$ | $c_5$ | $b_5$ | $a_5$ | 0 | $n_5$ | 4 | $a_0^5$ | $a_1^5$ | $a_2^5$ | 0 | $d_1$ | $k_1$ | $r_1^5$ | $r_2^5$ | $\mu_5$ | $1_5$ |

If $\omega_i=1$ then $\alpha_i$ is a singular point else $\alpha_i$ is a regular point. $a_ix^2+b_ix+c_i=0$ defines $\alpha_i$, where $a_i$ or $b_i\neq 0$. To simplify our explanation we assume that $\alpha_i$ is an integer. Deform the equation $E$ to the form

$$(E4)\qquad (x-\alpha_i)^2y''+(x-\alpha_i)F_0(x)y'+G_0(x)y=0,$$

and store $G_0(x)$ to SA(*, 1), $F_0(x)-1$ to SA(*, 2). We consider that SA represents $f(x, \lambda)\equiv\lambda^2+(F_0(x)-1)\lambda+G_0(x)$. With $f(x, \lambda)$, we calculate $f(\alpha_i, \lambda)$ and store to PA(7) and $T1(7\sim 13, i)$.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $PA(*)$ | $n$ | 4 | $a_0$ | $a_1$ | $a_2$ | 0 | $d$ |

$: f(\alpha_i, \lambda)=n(a_2^i\lambda^2+a_1^i\lambda+a_0^i)/d.$

If we can factorize $a_2^i\lambda^2+a_1^i\lambda+a_0^i$ to $(q_1\lambda+q_0)(r_1\lambda+r_0)$, then store this to W0.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $W0(*, 1)$ | 3 | $q_0$ | $q_1$ | | | | |
| $W0(*, 2)$ | 3 | $r_0$ | $r_1$ | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $F0(*, 1)$ | 3 | $q_0$ | $q_1$ | | | $v$ | $k$ |
| $F0(*, 2)$ | 3 | $r_0$ | $r_1$ | | | | |

If the difference of two roots of $(q_1\lambda+q_0)(r_1\lambda+r_0)=0$ is integer other than zero then $k:=$the difference else $k:=0$, and if the difference is zero then $v:=2$ else $v:=1$. These are stored into $F0$. If $k=0$ then $\alpha_i$ is not apparent.

We assume that $k>0$. With $f(x, \lambda)$ in $SA$ and with

$$f_m(\lambda)=\frac{1}{m!}\frac{\partial^m}{\partial x^m}f(x, \lambda)\Big|_{x=\alpha_i},$$

we can calculate $f_m(\lambda)$ and store it to $FM(*, m+1)$ consecutively. We set $g_0(\lambda):=1$. We assume that $f_0(\lambda), f_1(\lambda), \ldots, f_i(\lambda)$ and $g_0(\lambda), g_1(\lambda), \ldots, g_{i-1}(\lambda)$ have been already calculated and these $g_m(\lambda)$ are stored in $GM(*, m+1)$. We calculate $g_{ri}(\lambda)$ by

$$g_{ri}(\lambda)=f_1(\lambda+k-1)g_{i-1}(\lambda)+\cdots+f_i(\lambda)g_0(\lambda),$$

and store $g_{ri}(\lambda)$, the rational function of $\lambda$, to the integer array of length 43, then we calculate $g_i(\lambda)$ by

$$g_i(\lambda)=f_0^{-1}(\lambda+i)\cdot g_{ri}(\lambda).$$

Thus if $g_{ri}(\lambda)$ is divisible by $q_1\lambda+q_0$ then $x=\alpha_i$ is apparent else $x=\alpha_i$ is not apparent. For we have the relation

$$f_{i0}(\lambda)=(q_1\lambda+q_0)(q_1(\lambda+k)+r_2),$$

in the case of $k>0$.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $FM(*, 1)$ | $n^0$ | 4 | $a_0^0$ | $a_1^0$ | $a_2^0$ | 0 | $d^0$ | $f_0(\lambda)$, |
| $FM(*, 2)$ | $n^1$ | 3 | $a_0^1$ | $a_1^1$ | 0 | 0 | $d^1$ | $f_1(\lambda)$, |
| | | | | | | | | |
| $FM(*, N)$ | $n^N$ | 3 | $a_0^N$ | $a_1^N$ | 0 | 0 | $d^N$ | $f_{N-1}(\lambda)$, |

| | |
|---|---|
| $GM(*, 1)$ | $g_0(\lambda)$ |
| $GM(*, 2)$ | $g_1(\lambda)$ |
| | |
| $GM(*, N)$ | $g_{N-1}(\lambda)$ |

where $f_m(\lambda)$ represent the following polynomials.

$$f_0(\lambda)=n^0(a_2^0\lambda^2+a_1^0\lambda+a_0^0)/d^0,$$

$$f_1(\lambda)=n^1(\qquad a_1^1\lambda+a_0^1)/d^1,$$

......

Our FORTRAN program consists of about 5000 cards, and it needs about 65k words, where 1 word=32 bit. Its execution time per one problem is 1 second at most, and it is almost negligible compared with the printing time.

## §4. On the Evaluation of Formulas and the Types of Formulas in the General Formula Manipulations

We propose an evaluation algorithm of any formula $f$ appeared in $v := f$ etc. The right hand side of it might express not only a numerical value but also any formula. Hereafter we call such $f$ simply a formula. First we recall the evaluation algorithm of $f$ by a human. We use explicitly or implicitly many rules such as the commutative and associative law, and use many procedures such as the addition, multiplication of polynomials, etc.. Moreover in many cases we proceed our calculation as far as possible, and in a few cases we cease our calculation on the way, in spite of the possibility of proceeding it. For example, consider the calculation of a definite integral

$$g=\int_a^b h(x)dx\,.$$

If the value of $h(x)$ is itself, the value of $g$ is the right hand side itself, on the other hand if the value of $h(x)$ is $3x^2$ then the value of $g$ is $b^3-a^3$, moreover if the value of $a, b$ are $1, 2$ then the value of $g$ is 7. But in a few cases we wish that the value of $g$ is the formula $\int_1^2 3x^2dx$ itself. Here we assume that there are definitions of procedures which calculate $x^3$ from $3x^2$, $b^3-a^3$ from $x^3\Big|_a^b$, and 7 from $2^3-1^3$. Our intention is that these natural evaluations are proceeded without any indication except the existence or no existence of definitions of procedures.

We shall construct our language $L$ on the basic facilities and notations and notions of ALGOL–60.

### 4–1. The Syntax of $L$

The form of our program illustrated by (B) is called a block, where $D_i$, $\mu_i$, and $S_i$ are called declaration, label, and statement respectively.

$$(B) \qquad \textbf{begin}\, D_1;\cdots;D_n;[\mu_1:]S_1;\cdots;[\mu_n:]S_m\,\textbf{end},$$

$\mu_i\in N$ and $N$ is the set of all identifiers. $[\mu_i:]S_i$ represents $\mu_i: S_i$ or $S_i$. The $S_i$ of $\mu_i: S_i$ is called the statement whose label is $\mu_i$. The classification of declarations by their objects are as follows: (1d) variable (2d) type declarator (3d) operation (4d) transformation rule (5d) procedure or function. The classification of statements are as follows: (1s) block (2s) assignment (3s) conditional (4s) go to (5s) procedure or function.

In the following (1d)~(5d) and (1s)~(5s), we will explain the forms of declarations and statements of $L$.

(1d) Let $\chi$ be **real**, **integer**, **letter**, or a type declarator which was declared by the form of (2d), then the declaration of variables has one of the following forms,

$$(Dt) \qquad \begin{cases} \chi \quad u,\, o\ \textbf{chain}\ \chi \quad v,\ o_1\ \textbf{chain}\ o_2\ \textbf{chain}\ \chi \quad w,\dots \\ \chi \wedge \gamma \quad u_1,\ \chi \wedge \gamma(l_1,\dots,l_{n-1}) \quad u_2, \end{cases}$$

where $\chi$, $u$, $v$, $w$, $u_1$, $u_2$, $\gamma$, $l_i \in N$, and $o$, $o_1$, $o_2 \in 0$. $O$ is the set of operator symbols, $O=\{+,\ -,\ \cdot,\ /,\ \uparrow,\ *,\dots\}$. $\gamma$ is the name of a Boolean procedure, and $l_1,\dots,l_{n-1}$ are the actual parameters of $\gamma$, which are subformulas of $u_2$.

(2d) Let $g1,\dots,gn$ be the formulas then the declaration of a type declarator $\chi$ except **integer**, **real**, **letter** has one of the following forms,

$$(Dt) \qquad \begin{cases} \textbf{DT}\ \chi\ \textbf{is}\ (o,\, g1,\dots,gn), & o \in O \\ \textbf{DT}\ \chi\ \textbf{is}\ \sigma(g1,\dots,gn), & \chi,\ \sigma \in N, \end{cases}$$

where $\sigma$ may be the name of a procedure of function type.

(3d) Let $g1,\dots,gn$ and $h1,\dots,hm$ be formulas, then the declaration of an operation on certain formulas has one of the following forms,

$$(Do) \qquad \begin{cases} \textbf{DO}\ (o,\, g1,\dots,gn)\ \textbf{is}\ (o',\, h1,\dots,hm), & o,\ o' \in O \\ \textbf{DO}\ (o,\, g1,\dots,gn)\ \textbf{is}\ \sigma'(h1,\dots,hm), & \sigma,\ \sigma' \in N, \end{cases}$$

where $(o,\ g1,\dots,\ gn)$ may be of the form $\sigma\ (g1,\dots,\ gn)$.

(4d) Let $g1,\dots,gn$ be subformulas of $g$, and $\gamma$ the name of a Boolean procedure, then the declaration of a transformation rule between $g$ whose type is $\chi$ and $gi$ has the following form,

$$(Dr) \qquad \textbf{DR}\ \chi \quad g(\gamma(g1,\dots,gn))\ \textbf{is}\ gi.$$

(5d) Let $\phi, \phi_i$ be type declarators of the form $\chi$ or $o$ **chain** $\chi$ etc., then the declaration of a procedure has one of the following form. We call the former the subroutine type and the latter the function type.

$$(Dp) \qquad \begin{cases} \textbf{procedure } \mu(\varphi_1,\dots,\varphi_n);\ \phi_1\varphi_1;\ \dots;\ \phi_n\varphi_n;\ S_1\,, \\ \phi\ \textbf{procedure } \sigma(\varphi_1,\dots,\varphi_n);\ \phi_1\varphi_1;\ \dots;\ \phi_n\varphi_n;\ S_2\,, \end{cases}$$

where $\mu, \sigma, \varphi_1,\dots, \varphi_n \in N$, $S_i$ is a statement, $S_2$ must contain at least one assignment statement of the form $\sigma: =\cdots$.

(1s) A block has the form of $(B)$.

(2s) Let $V$ be the set of all variables, and let $F$ be the set of all formulas, then an assignment statement has the form of $v: =\phi$, or $v: =f$, where $v\in V, \phi\notin F, f\in F$. We define $V$ as follows. The identifier which was declared by a type declarator is a variable. Thus $u, v, w, u_1, u_2$ of $(Dv)$ are variables. Let $\chi$ be the type declarator defined by $(Dt)$, and let $u$ be the variable declared by '$\chi\ u$', and if $gi$ is a variable then the $gi$ part of $u$ can be referred by $u\_gi$, and this is also a variable. Therefore if necessary, we can use the variable of the form $u\_g^1_{i_1}\_\cdots\_g^n_{i_n}$. Let us consider the variable $v$ and $w$ of $(Dv)$, as we shall see later, $v$ and $w$ can have the value of the form $(o, r1,\dots, rn)$, and $(o_1, (o_2, s11,\dots, s1n_1),\dots, (o_2, sm1,\dots, smn_n))$ respectively, where $n$ and $m$ are determined in the execution time and are referred by $lgt(v)$ and $lgt(w)$. These $ri$, $sij$, and $(o_2, sk1,\dots, skn_k)$ can be referred by $v(i)$, $w(i, j)$, and $w(k)$ respectively. And these $v(i)$ etc. are called subscripted variables.

Let $I$ and $R$ be the set of all integers and all real (floating point) numbers respectively, then we can define $F$, the set of all formulas, by the following rules, and $gi$ is called a subformula of $(o, g1,\dots gn)$ etc.

$$(F) \qquad \begin{cases} 1)\ \ I, R, V\subset F\,, \\ 2)\ \ \text{if } g1,\dots, gn\in F \text{ then } (o, g1,\dots, gn)\in F, \text{ and } \sigma(g1,\dots, gn)\in F, \end{cases}$$

where $\sigma$ may be the procedure name of function type.

(3s) The conditional statements have the following form

**if** $B$ **then** $S1$ **else** $S2$,

where $B$ is a Boolean expression and S1, S2 are statements. A relational expression $R$ is a Boolean expression, and if $B$, $B1$, and $B2$ are Boolean expressions then $\neg B$, $B1 \wedge B2$, $B1 \vee B2$, and $B1 \equiv B2$ are also Boolean expressions. Let $Tp$ be the set of all types, and $T$ be the function which maps a formula to its type, the element of $Tp$. Let $ti$ be an element of $Tp$ or $T(fi)$, $(fi \in F)$ then a relational expression $R$ has one of the following forms,

$$(Re) \qquad t1 < t2, \quad t1 \leqq t2, \quad t1 = t2, \quad t1 \geqq t2, \quad t1 > t2.$$

Let $Dt$ be the set of all type declarators, then we can define $Tp$ by the following two rules,

$$(Tp) \quad \begin{cases} 1) \ I, R, Dt \subset Tp, \\ 2) \ \text{if } t1, \ldots, tn \in Tp \text{ then } (o, t1, \ldots, tn) \in Tp, \text{ and } \sigma(t1, \ldots, tn) \in Tp. \end{cases}$$

(4s) The form of go to statement is **go to** $\mu_i$.

(5s) The form of a procedure statement is $\mu(f1, \ldots, fn)$ where $fi \in F$ and $\mu$ is the procedure name of subroutine type.

### 4–2. The Semantics of *L*

(1) We define the function $T$ which maps $f \in F$ to its type $T(f) \in Tp$, as follows. We assume that $i \in I$, $r \in R$, and $v \in V$.

$$(T) \quad \begin{cases} T(i) = i, \ T(r) = r, \ T(v) = \text{the type declarator of } v, \\ T((o, g1, \ldots, gn)) = (o, T(g1), \ldots, T(gn)), \qquad gi \in F, \ o \in O, \\ T(\sigma(g1, \ldots, gn)) = \sigma(T(g1), \ldots, T(gn)), \qquad \sigma \in N. \end{cases}$$

Later we shall explain $T(v)$ more precisely.

(2) We define a partial order relation on $Tp$ as follows,

$$(R) \quad \begin{cases} 1) \ \textbf{integer} < i, \quad \textbf{real} < r, \\ 2) \ \text{if } \textbf{DT } \chi \textbf{ is } (o, g1, \ldots, gn) \text{ then } \chi < (o, T(g1), \ldots, T(gn)), \\ 3) \ \text{if } \textbf{DT } \chi \textbf{ is } \sigma(g1, \ldots, gn) \text{ then } \chi < \sigma(T(g1), \ldots, T(gn)), \\ 4) \ o \textbf{ chain } \chi < (o, \chi, \ldots, \chi), \quad \textbf{chain } \chi < (\chi, \ldots, \chi), \\ 5) \ \textbf{letter} < t, \text{ where } t \text{ is any type other than } \textbf{letter}, \\ 6) \ \chi < \chi \wedge \gamma(l_1, \ldots, l_{l-1}), \end{cases}$$

$$\left|\begin{array}{ll} 7) & \chi=\chi, \\ 8) & \text{if } \chi_i \leqq \phi_i \text{ and there is at least one } i \text{ such that } \chi_i<\phi_i \text{ then} \\ & (o, \chi_1,\dots, \chi_n)<(o, \phi_1,\dots, \phi_n), \text{ and } \sigma(\chi_1,\dots, \chi_n)<\sigma(\phi_1,\dots, \phi_n). \end{array}\right.$$

(3) We classify formulas by their phases of appearances. A 'program formula' is the character string which is written in the proper place of our program. Let $f$ be a program formula. We call the value of $f$ just before its evaluation the 'pre-value' of $f$, and write $Vr(f)$. Similarly we call the value of $f$ just after its evaluation the 'post-value' of $f$, and write $Vs(f)$. Both $Vr(f)$ and $Vs(f)$ depend on the set of declarations $D$ which effects on $f$ and depend on the history of execution $H$ until the evaluation of $f$. Therefore we write $Vr(f, D, H)$, $Vs(f, D, H)$, but if we could easily guess those $D$ and $H$ then we might omit those $D$ and $H$. The explanation of assignment statements are unavoidable to define $Vr(f)$ and $Vs(f)$. The conditions that allow to execute our assignment statement $v:=f$ are as follows,

$$(Ac) \quad \left\{\begin{array}{ll} 1) & \text{if } T(v)=\textbf{letter} \text{ or } \quad T(v)=\textbf{letter}\wedge\cdots \text{ then } T(v)\leqq T(Vs(f)), \\ 2) & \text{if } T(v)\neq\textbf{letter} \text{ and } T(v)\neq\textbf{letter}\wedge\cdots \text{ then } T(v)<T(Vs(f)). \end{array}\right.$$

(4) The definition of pre-value of $f$:

$$(Vr) \quad \left\{\begin{array}{lll} 1) & Vr(i, D, H)=i, \quad Vr(r, D, H)=r, & i\in I,\ r\in R, \\ 2) & Vr(v, D, Hd)=v, & v\in V, \\ 3) & Vr(v, D, Ha)=Vs(f, D, Ha), & v\in V, f\in F, \\ 4) & Vr((o, f1,\dots, fn), D, H)=(o, Vr(f1, D, H),\dots, Vr(fn, D, H)), & \\ & Vr(\sigma(f1,\dots, fn), D, H)=\sigma(Vr(f1, D, H),\dots, Vr(fn, D, H)), & \end{array}\right.$$

where $H$ represents an arbitrary history, $D$ represents an arbitrary set of declarations. $Hd$ represents the interval beginning with the declaration of $v$, or the execution of $v:=\phi$, ending with the execution of $v:=\cdots$, and excluding this execution. $Ha$ represents the interval beginning with the execution of $v:=f$, ending with the execution of another $v:=\cdots$, and excluding this execution.

(5) To define $Vs(f, D, H)$, we must use another pre-value of $f$ for

compilation $Vc(f, D, H)$. The definition is as follows.

$$(Vc)\quad \begin{cases} 1)\ \ Vc(i, D, H)=i, \quad Vc(r, D, H)=r, & i\in I,\ r\in R\,, \\ 2)\ \ Vc(v, D, Hd)=v, & v\in V, \\ 3)\ \ Vc(v, D, Ha)=Vc(f, D, Ha), & T(v)=\textbf{letter} \text{ or } \textbf{letter}\wedge\cdots, \\ \qquad\qquad =v\qquad\qquad , & T(v)\neq\textbf{letter} \text{ and } \textbf{letter}\wedge\cdots, \\ 4)\ \ Vc((o, f1,\dots,fn), D, H)=(o, Vc(f, D, H),\dots, Vc(f, D, H))\,, & \\ \quad\ \ Vc(\sigma(f1,\dots,fn), D, H)=\sigma(Vc(f, D, H),\dots, Vc(f, D, H))\,. & \end{cases}$$

(6) Using built-in function 'eval', we define $Vs(f, D, H)$ as follows,

$$(Ev)\qquad Vs(f, D, H)=\mathrm{eval}(Vc(f, D, H), Vr(f, D, H), D, H)\,.$$

6–1) The function $Frc$ is defined by the relation

$$Vr(f, D, H)=Frc(Vc(f), D, H)\,.$$

6–2) Any variable $v$ which was declared by '$\chi$ $v$' may be used in a definition of operation, then the variable $v$ is called 'formal' except when it is doubly declared by '**actual** $v$'.

6–3) A formula $f$ is called a compound formula if and only if it has the form $(o, f1,\dots,fn)$ or $\sigma(f1,\dots,fn)$, we call the latter a compound formula of function type.

6–4) A compound formula $f$ of the form $(o, f1,\dots,fn)$ is called active if there is a definition of operation of the form $(Do)$, and $f$ satisfies the condition: Let $g$ be $(o, g1,\dots, gn)$ in $(Do)$, then the system of equations $g1=f1,\dots, gn=fn$ with respect to the formal variables $\varphi_1,\dots,$ $\varphi_k$ in $g$ has at least a solution $\alpha=(\alpha_1,\dots,\alpha_k)$. We call $\alpha$ the matching parameter of $g$ for $f$. Let $g(\alpha)$ be $Vs(g)$ after the replacement of $\varphi_j$ by $\alpha_j$ where $T(\varphi_j)\leqq T(\alpha_j)$, $(j=1, 2,\dots, k)$ then $g(\alpha)=f$. (Refer to Appendix 1.)

6–5) A compound formula $f$ of the form $\sigma(f1,\dots,fn)$ is called active if $f$ satisfies 6–4) replacing $g$ by $\sigma(g1, \dots, gn)$ or if there is a declaration of procedure of function type which has the form

$$(Fu)\qquad \phi\ \textbf{procedure}\ \sigma(\varphi_1;\dots, \varphi_n);\ \phi_1\varphi_1,\dots, \phi_n\varphi_n;\ S,$$

where $\phi, \phi_1,\dots, \phi_n$ are type declarators, and $f$ satisfies the following conditions:

$$(Ac')\quad\begin{cases}\text{if } \phi_i=\textbf{letter} \text{ or } \phi_i=\textbf{letter}\wedge\cdots \text{ then } \phi_i\leqq T(fi),\\ \text{if } \phi_i\neq\textbf{letter} \text{ and } \phi_i\neq\textbf{letter}\wedge\cdots \text{ then } \phi_i< T(fi).\end{cases}$$

(7) With those notions we define the eval as follows.

7–1) eval$(i, i, D, H)=i$, eval$(r, r, D, H)=r$, $i\in I$, $r\in R$, where $D$ and $H$ are arbitrary set of declarations and history.

7–2) eval$(v, f, D, H)=f$, $v\in V, f\in F$.

7–3) eval $(f, f', D, H)=Exc(Frc(V(Vc(\tilde{f}, D, H))))$ after the execution of the program $w1:=f1;\cdots; wm:=fm$, where $f=(o, f1,\ldots, fm, v1,\ldots)$, $\tilde{f}=(o, w1,\ldots, wm, v1,\ldots)$, $fj$ are compound formulas, $vj$ are variables or constants, and $wj$ are variables generated by the system. $T(wj)$ is the largest type which may accept $Vs(fj)$, and which can be determined in the phase of $V$.

7–4) When $f=\sigma(f1,\ldots, fn, v1,\ldots)$, eval $(f, f', D, H)$ is similarly defined as in the 7–3).

7–5) Let $k\equiv(o, k1,\ldots, kn)$ or $\sigma(k1,\ldots, kn)$, and let $r\equiv V(k, D, H)$. We assume $o\neq +$ and $o\neq \cdot$ in a) and b).

a) If $k$ is active due to the declaration $(Do)$ then

$$r=V(Vc((o', w1,\ldots, wp, hq',\ldots, hm'))) \text{ or } r=V(Vc(\sigma(\cdots)))$$

after the execution of $w1:=h1';\cdots; wp:=hp';$, where $hj'$ is the formula which is obtained form $hj$ by replacing its formal variables $\varphi_1,\ldots, \varphi_r$ by the corresponding matching parameters $\alpha_1,\ldots, \alpha_r$ of $g$ for $k$. We assume $h1',\ldots, hp'$ are compound formulas and $hq',\ldots, hm'$ are variables or constants, and $wj$ are variables generated by the system.

b) If there is no declaration of operation definition corresponding to $k$, or even if there is such declaration, if $k$ is not active then,

$$r=(o, k1,\ldots, kn) \text{ or } r=\sigma(k1,\ldots, kn).$$

c) When $o=+$ or $\cdot$, $r=Exc(Frc(V(k, D, H)))$ is determined by the following procedure: $j:=1$;

11: $k':=STAND(k)$; we denote again $k'=(o, k1,\ldots, kn)$; if there is a partition of $\{1, 2,\ldots, n\}=\{i_1,\ldots, i_l\}\cup\{i_{l+1},\ldots, i_n\}$ such that

$\kappa=(o, k_{i_1},\ldots, k_{i_n})$ is active due to an operation definition then $k=Vc((o, wj, k_{i_l+1},\ldots, k_{i_n}))$ after the execution of $wj:=\kappa$, where $V(\kappa, D, H)$ is evaluated by the rules of a) and b) of 7–5) allowing $o=+$ or $\cdot$; $j:=j+1$; go to 11 else $r:=Exc(Frc(V(k')))$;

For brevity we denote this value $r=Vs(STAND(k))$.

d) *STAND* is the name of standard procedure of function type which transforms an $o$-chain of the form $(o, k_1,\ldots, k_{i-1}, (o, k_{i_1},\ldots, k_{in_i}),\ldots, k_n)$ to the $o$-chain $(o, k_1,\ldots, k_{i_1},\ldots, k_{in_i},\ldots, k_n)$ until the operator $o$ is removed from all the sub-formulas, and rearrange the sub-formulas according to the proper rules of precedence.

e) We assume that $k$ is not active in the sense of a), but if there is an active formula $\tilde{k}$ in the formulas which are transformed from $k$ by the rules of the form

$$(Dr) \qquad \mathbf{DR}\ \chi\ g(\gamma(g1,\ldots, gn))\ \mathbf{is}\ gi,$$

then we define $r=V(\tilde{k}, D, H)$.

If a formula $h=(o, h1,\ldots, hn)$ satisfies the conditions $\chi<T(h)$ and $Vs(\gamma(h1,\ldots, hn))=\mathbf{true}$, where $\gamma$ is the Boolean expression whose value depends only upon $T(hi)$, then $h$ may be identified with $hi$. Therefore a formula $k$ could be transformed to its identified formulas $\tilde{k}$ by replacing $k$ or its sub-formulas with its identified formulas.

7–6) Let $k\equiv\sigma(k1,\ldots, kn)$, or $(o, k1,\ldots, kn)$ and $r\equiv Exc(k, D, H)$. We define $r$ by dividing to the following cases.

f) If $k$ is active due to the declaration $(F_u)$, then

$$r=Vs(\sigma, D, He),$$

where $He$ is the state just after the execution of $S$.

g) If there is no declaration for $\sigma$, or even if there is such declaration, $k$ is not active, or $k=(o, k1,\ldots, kn)$ then

$$r=\sigma(k1,\ldots, kn) \text{ or } r=(o, k1,\ldots, kn)\,.$$

h) The evaluation method described at e) is similarly applicable to the case of $k=\sigma(k1,\ldots, kn)$.

## §5. An Example of Programming by *L*

### 5–1. The Program

We must remark that the formula in *L* could be written by the so called external formula in place of the internal formula, for example the external formula of $(+, a, b)$ is $a+b$.

**begin comment** (1) Substitute $y$ of $(Ey)$ by $f\cdot z$, then represent $F1$, $G1$ of $(Ez)$ by $F, G, f$. (2) Substitute $f$ of $(Ez)$ by $(x-\alpha)^{\lambda}$, then represent $F1$, $G1$ of $(Ez)$ by $F$, $G$, $(x-\alpha)$, $\lambda$. (3) Substitute $F$, $G$ of $(Ez)$ by $(x+3)\cdot(x^2+x)^{-1}$, and $(x^3+2\cdot x^2-3)\cdot(x^3+2\cdot x^2+x)^{-1}$ respectively, then we represent the coefficients of $(Ez)$ by the rational functions of $x$, where

$$(Ey)\qquad (y')'+F\cdot y'+G\cdot y=0,\qquad\qquad (Ez)\qquad (z')'+F1\cdot z'+G1\cdot z=0,$$

and $'$ is the differential by $x$. The results are as follows,

(1) $F1=2\cdot f^{-1}\cdot f'+F,\qquad G1=f^{-1}\cdot(f')'+f^{-1}\cdot f'\cdot F+G,$

(2) $F1=2\cdot\lambda\cdot(x-\alpha)^{-1}+F,\qquad G1=\lambda\cdot(\lambda-1)\cdot(x-\alpha)^{-2}+\lambda\cdot(x-\alpha)^{-1}\cdot F+G,$

(3) $(z')'+3\cdot x^{-1}\cdot z'+z=0;$

```
Boolean procedure indep(f, g); letter f, g;
    begin if f contains g then indep: =false else indep: =true end;
letter x, f, g, h, λ, μ; letter ∧ indep(x) v;
+chain letter c; integer k, m, n; integer 1;
DT term is k·x↑n;
DT pol is+chain term; pol p, q, r;
DT ratf is p·q↑(−1); ratf s, t;
DT fd is f*g;
DR pol p(1gt(p)=1 ∧ p(1)__n=0) is p(1)__k;
DR ratf s(s__q=1) is s__p;
DO m·f+n·f is (m+n)·f     ;
DO m·f+f   is (m+1)·f     ;
DO f+f     is 2·f         ;
DO f·c     is LMC(f, c)   ;
DO 0·f     is 0           ;
```

**DO** $1\cdot f$ **is** $f$ ;
**DO** $f\uparrow\lambda\cdot f\uparrow\mu$ **is** $f\uparrow(\lambda+\mu)$ ;
**DO** $f\uparrow\lambda\cdot f$ **is** $f\uparrow(\lambda+1)$ ;
**DO** $f\cdot f$ **is** $f\uparrow 2$ ;
**DO** $(f\uparrow\lambda)\uparrow\mu$ **is** $f\uparrow(\lambda\cdot\mu)$ ;
**DO** $f\uparrow 0$ **is** $1$ ;
**DO** $f\uparrow 1$ **is** $f$ ;
**DO** $1\uparrow\lambda$ **is** $1$ ;
**DO** $\mathbf{d}(f\uparrow\nu)$ **is** $\nu\cdot f\uparrow(\nu-1)\cdot\mathbf{d}f$ ;
**DO** $\mathbf{d}(f\cdot g)$ **is** $\mathbf{d}f\cdot g+f\cdot\mathbf{d}g$ ;
**DO** $\mathbf{d}c$ **is** $DCL(c)$ ;
**DO** $\mathbf{d}\nu$ **is** $0$ ;
**DO** $p+q$ **is** $PA(p, q)$ ;
**DO** $p\cdot q$ **is** $PM(p, q)$ ;
**DO** $p\div q$ **is** $PD(p, q, r, 1)$ ;
**DO** $s+t$ **is** $RA(s, t)$ ;
**DO** $s\cdot t$ **is** $RM(s, t)$ ;
**DO** $s\uparrow(-1)$ **is** $RI(s)$ ;
**DO** $\mathbf{d}s$ **is** $RD(s)$ ;
**DO** $f\cdot(g*h)$ **is** $(f\cdot g)*h$ ;

**ratf procedure** $RA(s, t)$; **ratf** $s, t$;
$RA:=(s\_\_p\cdot t\_\_q+s\_\_q\cdot t\_\_p)\cdot(s\_\_q\cdot t\_\_q)\uparrow(-1)$;
**ratf procedure** $RM(s, t)$; **ratf** $s, t$;
$RM:=(s\_\_p\cdot t\_\_p)\cdot(s\_\_q\cdot t\_\_q)\uparrow(-1)$;
**ratf procedure** $RI(s)$; **ratf** $s$; $RI:=s\_\_q\cdot s\_\_p\uparrow(-1)$;
**ratf procedure** $RD(s)$; **ratf** $s$;
$RD:=(\mathbf{d}s\_\_p\cdot s\_\_q+(-1)\cdot s\_\_p\cdot\mathbf{d}s\_\_q)\cdot(s\_\_q\cdot s\_\_q)\uparrow(-1)$;
**pol procedure** $PA(p, q)$; **pol** $p, q$;
**begin** add $p$ and $q$, store the result to $PA$ **end**;
**pol procedure** $PM(p, q)$; **pol** $p, q$;
**begin** multiply $p$ and $q$, store the result to $PM$ **end**;
**pol procedure** $PD(p, q, r, n)$; **pol** $p, q, r$; **integer** $n$;
**begin** $n:=$(degree of $p$) ( - degree of $q$)$+1$; divide $n\cdot p$ by $q$, the quotient is $PD$, the remainder is $r$ **end**;
**pol procedure** $PQ(p, q)$; **pol** $p, q$;

```
    begin the greatest common divisor of p and q is PQ end;
+chain letter procedure LMC(f, c); letter f; +chain letter c;
    begin integer i, n; n: =lgt(c);
      for i: =1 step 1 until n do LMC(i): =f·c(i) end;
+chain letter procedure DCL(c); +chain letter c;
    begin integer i, n; n: =lgt(c);
      for i: =1 step 1 until n do DCL(i): =dc(i) end;
ratf procedure RSIMP(s); ratf s;
    begin pol fc; fc: =PQ(s__p, s__q);
      s__p: =s__p÷fc; s__q: =s__q÷fc
    end;
·chain letter procedure Cut(ai, bj);
     ·chain letter ai; letter bj;
    begin integer k, m, n; n: =lgt(ai); m: =1;
      for k: =1 step 1 until n do if ai(k)≠bj then
        begin Cut(m): =ai(k); m: =m+1 end
    end;
procedure EQIVR(a, b, c);
     +chain letter a; chain letter b; +chain fd c;
    begin integer i, j, k, la, lb;
      la: =lgt(a); lb: =lgt(b); k: =1;
      for j: =1 step 1 until lb do
      begin integer n; +chain letter w; n: =1;
        for i: =1 step 1 until la do
          if ⌉ indep(a(i), b(j)) then
          begin w(n): =Cut(a(i), b(j)); n: =n+1 end
        c(k): =w*b(j); k: =k+1
      end
    end;
+chain letter Ey, Ez, E; chain letter b; letter y, z;
ratf F, F1, G, G1, r;
S1:   b: =(z, dz, d(dz));
S2: Ey: =G·y+F·dy+d(dy);
S3:   y: =r·z;
S4: EQIVR(Ey, b, Ez);
```

$S5$: $h:=Ez(3)\_f\uparrow(-1)$;

$S6$: $E:=h\cdot Ez$;

$S7$: $r:=x+1$;

$S8$: $F:=(x+3)\cdot(x\uparrow 2+x)\uparrow(-1)$;

$S9$: $G:=(x\uparrow 2+2\cdot x+(-3))\cdot(x\uparrow 3+2\cdot x\uparrow 2+x)\uparrow(-1)$;

$S10$: $F1:=RSIMP(E(2))$;

$S11$: $G1:=RSIMP(E(1))$;

$S12$: $Ez:=G1\cdot z+F1\cdot \mathbf{d}z+\mathbf{d}(\mathbf{d}z)$;

PRINT ($Ez$)

**end**;

**5–2. The Execution of Our Program**

We explain the pre-values and the post-values of the variables under consideration just after the execution of the statement labelled $Si$. We use the mixed representation of formulas, namely some parts of which are the internal representations and the other parts of which are external representations.

$S1)$ $Vr(b)\ =(z, \mathbf{d}z, \mathbf{d}(\mathbf{d}z))$

$S2)$ $Vr(Ey)\ =(+, G\cdot y, F\cdot \mathbf{d}y, \mathbf{d}(\mathbf{d}y))$

$S3)$ $Vr(y)\ =r\cdot z$

$S4)$ $Vs(Ey)\ =(+, G\cdot r\cdot z, F\cdot r\cdot \mathbf{d}z, F\cdot z\cdot \mathbf{d}r, 2\cdot \mathbf{d}r\cdot \mathbf{d}z, r\cdot \mathbf{d}(\mathbf{d}z), z\cdot \mathbf{d}(\mathbf{d}r))$

$Vs(Ez)\ =(+, (+, G\cdot r, F\cdot \mathbf{d}r, \mathbf{d}(\mathbf{d}r))*z, (+, F\cdot r, 2\cdot \mathbf{d}r)*\mathbf{d}z, r*\mathbf{d}(\mathbf{d}z))$

$S5)$ $Vr(h)\ =r\uparrow(-1)$

$S6)$ $Vr(E)\ =(+, (+, G, F\cdot \mathbf{d}r\cdot r\uparrow(-1), r\uparrow(-1)\cdot \mathbf{d}(\mathbf{d}r))*z,$
$(+, F, 2\cdot \mathbf{d}r\cdot r\uparrow(-1))*\mathbf{d}z, 1*\mathbf{d}(\mathbf{d}z))$

$S7)$ $Vr(r)\ =x+1$

$S8)$ $Vr(F)\ =(x+3)\cdot(x\uparrow 2+x)\uparrow(-1)$

$S9)$ $Vr(G)\ =(x\uparrow 3+2\cdot x\uparrow 2+(-3))\cdot(x\uparrow 3+2\cdot x\uparrow 2+x)\uparrow(-1)$

$S10)$ $Vs(E(2))=(3\cdot x\uparrow 2+6\cdot x+3)\cdot(x\uparrow 3+2\cdot x\uparrow 2+x)\uparrow(-1)$

$Vr(F1)\ =3\cdot x\uparrow(-1)$

$S11)$ $Vs(E(1))=(x\uparrow 3+2\cdot x\uparrow 2+x)\cdot(x\uparrow 3+2\cdot x\uparrow 2+3)\uparrow(-1)$

$Vr(G1)\ =1$

$S12)$ $Vr(Ez)\ =z+3\cdot x\uparrow(-1)\cdot \mathbf{d}z+\mathbf{d}(\mathbf{d}z)$

To show our evaluation algorithm, let us trace the execution of the assignment statement of ($S4$), we omit all $D$ and $H$.

$S4)$ $Vs(Ey)\ =\mathrm{eval}(Vc(Ey), Vr(Ey))$

$=\mathrm{eval}((+,\ G\cdot(r\cdot z),\ F\cdot\mathbf{d}(r\cdot z),\ \mathbf{d}(\mathbf{d}(r\cdot z))),\ '')$

$=Exc(Frc(V(Vc((+,\ w1,\ w2,\ w3)))))$ after the exec. of

$w1:=G\cdot(r\cdot z);\ w2:=F\cdot\mathbf{d}(r\cdot z);\ w3:=\mathbf{d}(\mathbf{d}(r\cdot z));$

$Vs(w1)=\mathrm{eval}\ ((\cdot,\ G,\ (\cdot,\ r,\ z)),\ '')=(\cdot,\ G,\ r,\ z)$

$Vs(w2)=\mathrm{eval}\ ((\cdot,\ F,\ \mathbf{d}(r\cdot z)),\ '')$

$=Exc(Frc(V(Vc((\cdot,\ F,\ w4)))))$ after the exec. of

$w4:=\mathbf{d}(r\cdot z);$ , where $T(w4)=$ **letter.**

$Vs(w4)=\mathrm{eval}\ (\mathbf{d}(r\cdot z),\ '')=Exc(Frc(V(Vc(\mathbf{d}(r\cdot z)))))$

$=Exc(Frc(V(\mathbf{d}(r\cdot z))))$

$=Exc(Frc(V(Vc((+,\ w5,\ w6)))))$ after the exec. of

$w5:=\mathbf{d}r\cdot z;\ w6:=r\cdot\mathbf{d}z;$

$Vs(w5)=z\cdot\mathbf{d}r$

$Vs(w6)=r\cdot\mathbf{d}z$

$Vs(w4)=r\cdot\mathbf{d}z+z\cdot\mathbf{d}r$

$Vs(w2)=Exc(Frc(V((\cdot,\ F,\ r\cdot\mathbf{d}z+z\cdot\mathbf{d}r))))$

$=Exc(Frc(LMC(F,\ r\cdot\mathbf{d}z+z\cdot\mathbf{d}r)))$

$=F\cdot r\cdot\mathbf{d}z+F\cdot z\cdot\mathbf{d}r$

We omit the detail of calculations for $Vs(w3)$ and $Vs(Ey)$.

$Vs(w3)=(+,\ 2\cdot\mathbf{d}r\cdot\mathbf{d}z,\ r\cdot\mathbf{d}(\mathbf{d}z),\ z\cdot\mathbf{d}(\mathbf{d}r))$

$Vs(Ey)=Vs(STAND((+,\ (\cdot,\ G,\ r,\ z),\ (+,\ (\cdot,\ F, r, \mathbf{d}z),\ (\cdot,\ F,\ z, \mathbf{d}r)),$
$(+,\ (\cdot,\ 2,\ \mathbf{d}r,\ \mathbf{d}z),\ (\cdot,\ \mathbf{d}(\mathbf{d}z)),\ (\cdot,\ z,\ \mathbf{d}(\mathbf{d}r))))))$

## Appendix 1.

Let us consider a system of equations,

$$(Eq)\qquad g1=f1,\dots,\ gn=fn,\quad (gi,\ fi\in F,\ i=1,\dots,\ n)$$

with respect to the formal variables $\varphi_1,\dots,\varphi_k$ which are contained in $gi$ and not contained in $fi$. We use the following procedure which solves $(Eq)$.

(1) If $g1,\dots,gn$ are linear with respect to $\varphi_1,\dots\varphi_k$, the procedure is Gauss' elimination method.

(2) If $g1,\dots,gn$ are not linear with respect to $\varphi_1,\dots,\varphi_k$, we seek $gi$ which contains two formal variables at most. Such equation has the form

$$(o,\ \varphi_i,\ h_i\dots)=f_i \text{ or } (o,\ \varphi_i,\ \varphi_j,\ h_i,\dots)=f_i\,.$$

The former could be solved if $o=+,\ \cdot,\ \uparrow$, etc., generally the latter has no solution or has many solutions. But in many cases we can seek the candidates of the solutions. Thus we can seek the candidates of the solutions of $(Eq)$, and if one of them really satisfies $(Eq)$, it is our solution.

**Example**: Under the following declarations,

**letter** $a,\ b,\ c,\ d,\ x$; **actual** $x$;

**DO** $a\cdot c\cdot x\uparrow 2+(a\cdot d+b\cdot c)\cdot x+b\cdot d$ **is** $(a\cdot x+b)*(c\cdot x+d)$;
$6\cdot x\uparrow 2+13\cdot x+5$ matches with $a\cdot c\cdot x\uparrow 2+(a\cdot d+b\cdot c)\cdot x+b\cdot d$, and we get a system of equations:

$$a\cdot c=6,\ a\cdot d+b\cdot c=13,\ b\cdot d=5.$$

Using the above procedure (2), we get a solution, i.e.

$$a=2,\ b=1,\ c=3,\ d=5.$$

## Appendix 2.

```
      SUBROUTINE RA                                                     RFC00000    1
C  ** ADD TWO RATIONAL FUNCTIONS A, B AND THE RESULT IS STORED TO C     RFC00010
      COMMON /SW/ISW1 /FROBR/A,B,C                                      RFC00030
      INTEGER  A(42), B(42), C(42)                                      RFC00040
      INTEGER AN(20),AD(20),BN(20),BD(20),CN(20),CD(20), W(20),W1(20)   RFC00050
      EQUIVALENCE ( A( 2),AN( 1)),( A(23),AD( 1)),( B( 2),BN( 1))       RFC00060
      EQUIVALENCE ( B(23),BD( 1)),( C( 2),CN( 1)),( C(23),CD( 1))       RFC00070
      IF ( ISW1 )              1 , 2, 1                                 RFC00080    2
    1 WRITE(6,90)                                                       RFC00090    3
    2 C(22)      = A(22)*B(22)                                          RFC00100    4
      CALL PPM( AD, BD, CD )                                            RFC00110    5
      CALL PPM( AN, BD, CN )                                            RFC00120    6
      ND         = A( 1)* B(22)                                         RFC00130    7
      MCN        =CN( 1)                                                RFC00140    8
      DO  100 I=2,MCN                                                   RFC00150    9
  100 CN(I)      =ND*CN(I)                                              RFC00160   11
      CALL PPM( AD, BN,  W )                                            RFC00170   12
      ND         = A(22)* B( 1)                                         RFC00180   13
      MW         = W( 1)                                                RFC00190   14
      DO  200 I=2,MW                                                    RFC00200   15
  200 W(I)       =ND* W(I)                                              RFC00210   17
      CALL PPA( CN, W, W1)                                              RFC00215   18
      DO  300 I=1,20                                                    RFC00220   19
  300 CN(I)      =W1(I)                                                 RFC00230   21
      CALL PCF( CN, C(1) )                                              RFC00240   22
      DO  400 I=2,20                                                    RFC00250   23
  400 CN(I)      =CN(I)/C(1)                                            RFC00260   25
      CALL IQ(  C( 1), C(22), IW )                                      RFC00270   26
      CALL RCH                                                          RFC00280   27
      IF ( ISW1 )              3 , 4, 3                                 RFC00290   28
    3 WRITE(6,91)                                                       RFC00300   29
      WRITE(6,92) C                                                     RFC00310   30
    4 RETURN                                                            RFC00320   31
   90 FORMAT(1H0,    2HRA)                                              RFC00330   32
   91 FORMAT(1H0,9X,2HRA)                                               RFC00340   33
   92 FORMAT(1H0,18X,11I9/28X,10I9)                                     RFC00350   34
      END                                                               RFC00360   35
```

## Appendix 3.

The number of our test equations is about 60, and in this paper we display only 10 typical equations and their integrations. $Bn(x)$ is the general solution of Bessel's equation of order $n$, and $D$ is $d/dx$.

(2) $4x^2y''+4xy'+(4x^2-1)y=0, \qquad y=x^{-1/2}e^{1x}\left\{A+B\int e^{-2ix}x^0dx\right\}.$

(3) $(x^2-x)^2y''-2xy=0, \qquad y=x(x-1)^{-1}\left\{A+B\int x^{-2}(x-1)^2dx\right\}.$

(5) $(x^2-x)^2y''+4(x^2-x)y'+2xy=0,$

$$y=x^5(x-1)^{-2}D\left[x^{-4}(x-1)^2\left\{A+B\int x^3(x-1)^{-3}dx\right\}\right].$$

(8) $(x^2+1)^2y''+x(x^2+x)y'+2(x^2+1)y=0,\ y=(\sqrt{x_1}\pm\sqrt{x_1-1})^{2\sqrt{2}i},$

$$x_1=\frac{(x-i)}{-2i}$$

(9) $(x^3-x)^2y''-x(x^2-1)(x^2-2)y'+(x^6-3x^4+2x^2)y=0,$

$$y=x\left\{A+B\int(x+1)^{-1/2}(x-1)^{-1/2}dx\right\}.$$

(15) $x^2y''+4xy'-4x^2y=0,\ y=e^{-2x}x^0D\left[x^{-2}e^{4x}\left\{A+B\int e^{-4x}x^1dx\right\}\right].$

(17) $(x^3-x)^2y''-x(x^2+1)(x^2-1)y'+(x^2-1)^3y=0,\ y=xB1(x).$

(19) $4y''+(-x^2+6)y=0,\ y=x_1^{1/2}e^{-1/4x_1}\left\{A+B\int x^{1/2x_1}x_1^{-3/2}dx_1\right\},\quad x_1=x^2.$

(24) $(x^3-x)y''+x(x^2-1)(-x^2+x+1)y'-x^3y=0,$

$$y=(x-1)^{(-3+\sqrt{5})/4}(x+1)^{(-1-\sqrt{5})/4}D\left[(x-1)^{(2-\sqrt{5})/2}(x+1)^{(2+\sqrt{5})/2}\right.$$

$$\left.\times\left\{A+B\int(x-1)^{(\sqrt{5}-4)/2}(x+1)^{(-\sqrt{5}-4)/2}dx\right\}\right].$$

(29) $24^2x^2(x-1)^2y''+24x(x-1)(14x-16)y'-11x(x-1)y=0,$

$y$ has no expression by the known functions.

## Appendix 4.

```
PROBLEM      (  9)

Y0''+F(X0)*Y0'+G(X0)*Y0 = 0

WHERE

                    (-     X$2+    2     )
      F (X) =-------------------------------
                    (      X$3-     X    )

                    (      X$4-    3X$2+    2      )
      G (X) =------------------------------------------
                    (      X$3-     X    )$2

SINGULAR POINTS ARE
        0        1        -1       INFINITY

F0(L) =                  L$2-   3L+   2

CHARACTERISTIC EQUATION AT    0  IS

          L$2-   3L+    2=(     L-    1)*(     L-    2)=0

F1(L) =         0

      GR(L) =    0

F2(L) =             ( -1L+   1)

      G2(L) =    0

SINGULAR POINT   0 IS APPARENT.

F0(L) = (  1/  2)*(  2L$2-    L     )

CHARACTERISTIC EQUATION AT    1  IS

         2L$2-    L      =      L       *(    2L-    1)=0

,AND NOT APPARENT

F0(L) = (  1/  2)*(  2L$2-    L     )

CHARACTERISTIC EQUATION AT   -1  IS

         2L$2-    L      =      L       *(    2L-    1)=0

,AND NOT APPARENT

RANK OF INFINITY IS  0

CHARACTERISTIC EQUATION AT INFINITY IS

          L$2+   2L+   1 = 0
```

```
PROBLEM       ( 19)

YO''+F(XO)*YO'+G(XO)*YO = 0

WHERE

      F (X) =     0

                      (-      X$2+    6     )
      G (X) =---------------------------------
                              4

SINGULAR POINTS ARE
     INFINITY

RANK OF INFINITY IS  2

     X 1 = X 0$ 2

YO''+F(X1)*YO'+G(X1)*YO = 0

WHERE

                  2
      F (X) =----------------------
                      (     4X    )

                      (-      X$2+    6X    )
      G (X) =---------------------------------
                      (     4X    )$2

SINGULAR POINTS ARE
        0       INFINITY

FO(L) = (  1/  2)*(  2L$2-    L     )

CHARACTERISTIC EQUATION AT    0  IS

         2L$2-     L     =     L      *(   2L-   1)=0

,AND NOT APPARENT

RANK OF INFINITY IS  1

CHARACTERISTIC EQUATION AT INFINITY IS

        16L$2-   1       = 0

     YO = EXP(( -1/  4)*X1)*X1**(  1/  2)*

(A+B*INT( EXP((  1/  2)*X1)*X1**( -3/  2))DX1)
```

```
PROBLEM      ( 24)
Y0''+F(X0)*Y0'+G(X0)*Y0 = 0
WHERE
                 (-     X$2+     X  +   1     )
      F (X) =---------------------------------------------
                 (      X$3-     X   )
                 (-     X$3 )
      G (X) =-----------------------------------
                 (      X$3-     X   )$2
SINGULAR POINTS ARE
        0        1        -1      INFINITY
F0(L) =                LS2-   2L
CHARACTERISTIC EQUATION AT     0  IS
          L$2-    2L      =      L       *(    L-    2)=0
F1(L) =               -1L
      G1(L) =      0
F2(L) =          0
      GR(L) =      0
F3(L) =             ( -1L-   1)
                   1
      G3(L) =---------
                   3
SINGULAR POINT    0 IS APPARENT.
                                                  AND ITS EXPONENT IS    2
F0(L) = (   1/   4)*(   4L$2-   2L-   1)
CHARACTERISTIC EQUATION AT     1  IS
         4L$2-    2L-    1
,AND NOT APPARENT
F0(L) = (   1/   4)*(   4L$2-   6L+   1)
CHARACTERISTIC EQUATION AT    -1  IS
         4L$2-    6L+    1
,AND NOT APPARENT
RANK OF INFINITY IS  0
CHARACTERISTIC EQUATION AT INFINITY IS
          L$2+    2L       = 0
```

```
Y0'=(X0- 0)$( 1)*Y1

TRANSFORMED EQUATION BY TYPE  2IS AS FOLLOWS

Y1''+F(X0)*Y1'+G(X0)*Y1 = 0

WHERE

                     (     4X   +    1     )
       F (X) =----------------------------------
                     (      X$2-    1      )

                     (      X    )
       G (X) =----------------------------------
                     (      X$2-    1      )$2

SINGULAR POINTS ARE
         1        -1      INFINITY

F0(L) = (   1/   4)*(  4L$2+   6L+   1)

CHARACTERISTIC EQUATION AT     1   IS

          4L$2+    6L+    1

,AND NOT APPARENT

F0(L) = ( -1/   4)*( -4L$2-   2L+   1)

CHARACTERISTIC EQUATION AT    -1   IS

          4L$2+    2L-    1

,AND NOT APPARENT

RANK OF INFINITY IS  0

CHARACTERISTIC EQUATION AT INFINITY IS

           L$2-    3L        = 0

     Y1 = (X0- 1)**(( -3+ 1*S(  5))/  4)*

           (X0+ 1)**(( -1- 1*S(  5))/  4)*

   D( 1)( (X0- 1)**((  2- 1*S(  5))/  2)*

           (X0+ 1)**((  2+ 1*S(  5))/  2)*

(A+B*INT( (X0- 1)**(( -4+ 1*S(  5))/  2)*

           (X0+ 1)**(( -4- 1*S(  5))/  2)* DX0))
```

## References


[1] Hukuhara, M., *Integration methods of linear ordinary differential equations II (linear equations)*, Iwanami 1941 (in Japanese).

[2] Hukuhara, M., On Fuchsian equations which are changeable to each other by linear transformations, *Jour. of Tsuda College*, 2 (1970), 1–12 (in Japanese).

[3] Hukuhara, M. and Ohashi, S., On the determination of Types of Riemann's $P$-function which can be expressed by elementary functions, *Sugaku*, (1949), 227–230 (in Japanese).

[4] Hukuhara, M. and Ohashi, S., On Riemann's $P$-function which is expressed by elementary functions, *Sugaku*, (1952), 27–29 (in Japanese).

[5] Kimura, T., On Riemann's Equations which are Solvable by Quadratures, *Funkcial. Ekvac.*, **12** (1969), 269–281.

[6] Kimura, T., On Fuchsian Differential Equations reducible to Hypergeometric Equations by Linear Transformations, *Funkcial. Ekvaci.*, **13** (1970), 213–232.

[7] Schwarz, H. A., Über diejenigen Fälle, in welchen die Gaussische hypergeometrische Reihe eine algebraische Function ihres vierten Elements darstellt, *J. Reine Angew. Math.*, (1872), 292–335.

[8] Cayley, A., On the Schwarzian Derivative, and the Polyhedral Functions. *Trans of the Cambridge Phil. Soc.*, **III**. Part I (1881), 5–68, *Oeuvre* 745.

[9] Picard, C., *Traité d'analyse*, **3**, chapitre II–III. 1928.

[10] Goursat, É., *Leçons sur les Series Hypergéometriques et sur quelque fonctions s'y rattachent*, I. Propriétés Générales de L'équation D'Euler et DE GAUSS II. Intégrales Algébriques probleme D'inversion.

[11] Kamke, E., *Differentialgleichungen, Lösungsmethoden und Lösungen*, 1948, New York, Chelsea.

[12] Murphy, G., *Ordinary Differential Equations and their Solutions*. Van Nostrand, Reinhold, 1960.

[13] McCarthy, J., et al., LISP 1.5 *Programmer's Manual*. The M. I. T. Press. 2nd edition 1969.

[14] Naur, P., et al., Revised Report on the Algorithmic Language ALGOL–60, *Comm. ACM.*, **6** (Jan. 1963), 1–17.

[15] Iwamura, T., Kakehi, K., Sakuma, K., Simauti, T., Wada, E. and Yoneda, N., ALGOL–N, *Comment. Math. Univ. St. Pauli.*, **XXI–1** (1972).

[16] van Wijngaarden (editor) et al., *Report on the algorithmic language* ALGOL 68, The Mathematisch Centrum, Amsterdam, MR101, Oct. 1969.

[17] Martin, W., Fateman, R., Moses, J. and Wang, P., *The* MACSYMA, Papers 1970 Second Symposium on Symbolic and Algebraic Manipulation held March 23–25, 1971 in Los Angels.

[18] Bobrow, J. (editor), *Symbol Manipulation Languages and Techniques*, North-Holland, 1968.

[19] Hearn, A. C., REDUCE 2, *User's Manual*, University of Utah, 1973.

[20] Watanabe, S., Formula Manipulations Solving Linear Ordinary Differential Equations I. *Publ. RIMS, Kyoto Univ.*, **6** (1970), 71–111.