PRADIS

THE DESCRIPTION OF LANGUAGE PradiSLang

THE SOFTWARE FOR SIMULATION OF NON-STATIONARY PROCESSES IN MECHANICAL SYSTEMS AND SYSTEMS OF OTHER PHYSICAL NATURE

VERSION 4.2

Contents

THE ALPHABET OF LANGUAGE PradiSLang

In language *PradiSLang* following numerals are used:

Capital and lower case letters of the Latin alphabet	A-Z, a-z
Capital and lower case letters of the Russian alphabet	А-Я, а-я
Digits	0-9
Signs of monadic arithmetical operations	+ -
The giving sign	=
Parentheses	) (
Decimal point	.
Separators and derivative indications	";: '/<CR>
Title and subtitle indications	$ #
Numerals of comments	} {
Underlining numeral	_
White space	

GENERAL PROVISIONS

SYNTACTICAL UNITS

All **instructions** of language *PradiSLang* are recorded in the first 72 positions of a line. Any instruction can be continued on the next line, thus the last line should be completed by the matching separator. The amount of lines of prolongation is not restricted.

The real number is recorded in the usual shape or in shape with the decimal multiplier. White spaces in a mantissa and an order are not admissible. Instances of correctly recorded real numbers:

-3-3.5-.5 1.7E-10 15.7 e-3

The whole positive numbers are recorded by the sequence of digits which is not containing a white space. The whole negative numbers and whole 0 in *PradiSLang* are not used.

Names in the program are used for definition of references to the programs realising models of elements, the program of calculation of output variables, fragments and in other cases stipulated more low. The name is a sequence no more than from 8 numerals, necessarily beginning with the character. All names in *PradiSLang*, except for names of fragments, are recorded by capital letters of the Latin alphabet. Admissible numerals in a fragment name are capital and lower case letters of the Latin and Russian alphabet, digits and an underlining numeral.

Identifiers in the program are used for a label of argument lists, the concrete constructive elements making presented installation, output variables and in other cases stipulated more low. The identifier is a sentence from no more, than 32 numerals (capital and lower case letters, digits, white spaces, points and underlining numerals), beginning with the character. Only the first white space from several, standing successively, is meaning and disjoints words going into a sentence.

Comments can be switched on in the program text. The comment any group of numerals bracketed {} is considered. The enclosure of comments is not admitted.

White spaces in the program, except specially stipulated cases, are ignored.

PROGRAM STRUCTURE

PradiSLang refers to to type of not procedural (descriptive) languages and includes the means necessary for the description of structure and parametres of examined engineering installation.

Generally the program in language *PradiSLang* consists of following main bodies:

- the description of data;
- the description of modelled installation;
- the description of the image of installation;
- the description of the job for calculation and representation of outcomes.

According to it at the program there can be **sections** of the description of data, descriptions of installation, the description of the image of installation, the description of the job for calculation, descriptions of the job for representation. Each section of the program begins **title**. At section of the description of installation there can be **subsections**, each of which begins **a subtitle**.

Titles and subtitles are recorded after numerals $ and # by capital letters of the Latin or Russian alphabet. Instances of titles and subtitles:

```
$FRAGMENT
  # OUTPUT :
$ЗАМЕНИТЬ
$КОНЕЦ
```

The program on *PradiSLang* should contain without fail title $ PRINT and be completed by title $ END. The information which is in the program to the first indication of title ($) and after title $ END, is not analysed, however it should not contain the numerals which are not going into the alphabet of language.

The installation description consists of one or several sections of the description of installation. Further in this deed **the section of the description of installation** matches to the term a synonym **an installation fragment** (or simplis **fragment**).

For the description of structure of installation subsections **of the description of structure of installation** are used. During the description of structure the installation model is represented in the form of a population of models of the elements which have been switched on in libraries of a complex. In the capacity of elements at the structure description the fragments which description in the program text precedes the description of a current fragment can be used also. Elements incorporate among themselves in certain points of physical space. The condition of these points is characterised by a quantity **of generalised co-ordinates** (or **a degree of freedom**) and their derivatives. The amount of a degree of freedom in each point of physical space depends generally on dimensions of a quantity of the space, used models of elements and a mode of the joint of these elements. Further in this deed concept **the node** is used as a synonym of concept **an installation degree of freedom**. Installation nodes are foliated by positive integers.

At the description of each of elements the certain amount of nodes to which this element incorporates is set. When it is a question of a degree of freedom of a separate element, conveniently to operate with concept **an element branch**. So, further in this deed concept the element branch is used as a synonym of concept **an element degree of freedom**. Numbers of branches of an element are defined by order of their job at the element description. For example, if the element is connected to installation in nodes with numbers 30, 1 and 27 its first branch the branch connected to a node 30 is considered, second - connected to a node 1, the third - connected to a node 27. Thus, number of a branch of an element cannot exceed amount of a degree of freedom of this element.

During calculation *PradiSLang* operates with the variables named further the interior. To **interior variables** refer to, for example in the mechanic, forces, migrations, velocities, accelerations etc., i.e. the magnitudes gained immediately during the solution. The amount of such variables even at insignificant sizes of model is great, and in many cases can attain hundreds and thousand. Therefore is inexpedient to save the information on each interior variable. Immediately to representation the magnitudes computed on various algorithms with use of interior variables and presented as **output variables** are accessible. The information on output variables is saved after a solving. Output variables are presented in matching subsections **of the description of a leading-out for a fragment**.

In the program text in language *PradiSLang* there is a possibility to define the image of installation during calculation. In section **of the description of the image** of installation correspondence between the models of elements which are going into the description of structure of installation, and their graphical images is defined. Groups of graphical images are merged in **image stratums**. There is a possibility of separate control of the image in each of stratums. At desire, the installation image can be generated automatically from standard graphical images under the description of structure of installation.

Models of elements, programs of calculation of output variables, the program of implementation of graphical images and the program of implementation of the image of installation demand job of certain amount of parametres which are presented in the form of **argument lists**. The argument lists used at the description of installation and the job, as a rule, are defined in separate sections **of the description of data**. Sections of the description of data

are not obligatory in the program in language *PradiSLang*. The data which are referring to to the description of installation, its image or to the job description, it is possible to set immediately in the text of matching sections. However, use of sections of the description of data for creation of the named argument lists largely facilitates operation on preparation of data, improves readership of the program and facilitates its further maintenance (for example, there is accessible such possibility as substitution of parametres in the generated model).

In sections **of the description of the job** besides transient calculation such possibilities, as substitution of parametres in already generated model without realisation of repeated generation, restoration of a condition of calculation from the discontinued place and saving of a condition of calculation at will of the user are realised. In each job in language *PradiSLang* presence of the description of representation of outcomes is supposed. Thus **represented variable** can be any of output variables. In other words, the assemblage of represented variables in each concrete case is a nonempty subset of assemblage of output variables. There is a possibility to shape jobs only for representation of outcomes spent before calculations in which it is possible to change composition of represented variables, scales for graphs of represented variables, to change time window boundary lines for which there is a representation of outcomes, to use others, than has been provided earlier, representation programs, arbitrarily to combine represented variables etc.

Instance of the elementary program in language *PradiSLang*:

```
{The description of given}
$ DATA:
     Mass = 7.5;
     Stiffness coefficient = 2; the Scale = 1
{The installation description}
$ FRAGMENT:
   # # BASE : 1
   # STRUCTURE:
         Spring ' K (1 2; the Stiffness coefficient);
         Mass ' M (2; Mass);
         Force ' F (2; 1.)
   # # OUTPUT :
         Cargo migration ' S (2; the Scale)
         Cargo velocity ' V (2; the Scale)
         Force to a spring ' X (I:Пружина; 1.)
{The job description}
{The description of the job for calculation}
$ $ RUN :
         Calculation of oscillations of a cargo ' SHTERM (END=1.0;
         Cargo velocity = (0.5),
         Cargo migration = (-0.1,0.1),
         Force to a spring)
{The description of the job for representation of outcomes}
$ $ PRINT :
         Cargo oscillations ' DISP (;
         Cargo velocity = (0.5),
        Migration of a cargo,
        Cargo migration = (-0.005,0.),
        Force to a spring)
$ END
```

THE DESCRIPTION OF DATA - SECTION $ DATA

Use **of the named argument lists** allows to improve considerably readership of the program in language *PradiSLang* and ensures a possibility of a modification of values of parametres at operation with the generated model of installation. For the description of the named argument lists the special section of the description of data serves. The section begins with title:

$ DATA: ***Name***

Where *a name* - a name of section of the description of data.

This *name* defines, that the section of data presented under this title, refers to to the section of the description of installation with the same name (the fragment with the same name). Thus the data unit description should precede the description of a fragment to which there match these data. One matches to each fragment and only one named data unit.

At the beginning of each program in language *PradiSLang* there can be one not named title $ DATA. The data presented in not named title $ DATA, are global and accessible to all program.

Behind title $ DATA on the next lines the description of argument lists in shape is given:

The identifier = the list
The list = number
Or the list = the list, number
Or the list = ~идентификатор [(number = number..., number = number)]
[, the list]

индентификатор — The argument list identifier.

The list — Shaped for use in the description of installation and its image an argument list.

Number — The real number setting the matching parametre value.

Number — Number of parametre which is substituted for in an argument list with the identifier *~индентификатор* on parametre, set by a real number standing after an equals sign.

~индентификатор — The identifier of the argument list defined earlier.

The sequence of real numbers and their amount should match strictly to a sequence and amount of parametres of that element at which description it is supposed to use the given argument list.

The description of each argument list should separate from other descriptions of argument lists a semicolon or the extremity of a line. Each argument list can place in one or several lines of the text of the program. In the latter case each transferable line should be completed by the matching separator ("," or "=").

Instance. It is required to create the global data unit for all program and the local data unit for a fragment "Pendulum".

```
$ DATA:
    Scale = 1.;
    Modulus = 2.E11
$ DATA: Pendulum
    Point And = 0., 0.;
    Point In = 1., 0.;
    Material = 0.001, the Modulus;
    Mass = 1.;
    Gravity(1); 1)
    Parametres of a pendulum 1 = the Point And,
                          Point In,
```

```
                        Material;
Parametres of a pendulum 2 = the Point And,
                        Point In (2=1.5),
                        Material;
```

The local data unit, accessible "Pendulum" from a fragment will be as a result generated the global data unit which argument lists will be accessible in all program, and. In this local data unit following argument lists will be defined:

```
Point And = 0., 0.;
Point In = 1., 0.;
Material = 0.001, 2.E11;
Mass = 1.;
Gravity(1); 1)
Parametres of a pendulum 1 = 0., 0., 1., 0., 0.001, 2.E11;
Parametres of a pendulum 2 = 0., 0., 1., 1.5, 0.001, 2.E11;
```

THE INSTALLATION DESCRIPTION - SECTION $ FRAGMENT

The section of the description of installation (fragment) begins with title:

$ FR [AGMENT]: ***Name***

Name - a fragment name.

Generally the installation structure can be presented one or several fragments switched on each other. Thus always there is only one fragment of the uppermost level including fragments named (necessarily) of the lower level. The upper level fragment can have a name, but can be and not named.

The extent of an enclosure of fragments each other is not restricted Each fragment consists of the subsections having following subtitles:

The job of base nodes - a subtitle # BASE (it is not obligatory; if is, follows right after title $ FRAGMENT).

The description of structure of a fragment - a subtitle # STRUCTURE (it is obligatory and follows at once a subtitle # BASE or after title $ FRAGMENT).

The description of exterior nodes of a fragment (the nodes serving for the joint with nodes of fragments which structure will switch on a presented fragment) - a subtitle # EXTERNAL (if necessary).

The description of output variables - a subtitle # OUTPUT (if necessary; however at least in one fragment of the job the subtitle # OUTPUT) should be defined.

The demand to print the debugging information on structure of installation with the direction of identifiers and the names which have been switched on in a fragment of models of elements (including going into switched on fragments), and also global numbers of nodes to which these elements incorporate - a subtitle # MAP (if necessary).

Subsections #EXTERNAL, #MAP and #OUTPUT should follow subsection #STRUCTURE. The Sequence of these subsections not существен.

THE DESCRIPTION OF BASE NODES OF THE FRAGMENT - THE SUBTITLE # BASE

This subtitle serves for the description of nodes, the kinematic or other which potential performances are accepted for 0 (a potential, temperature, pressure etc.). For example, in the mechanic are timbered nodes, in pneumatics - the nodes connected to an aerosphere etc. Further such nodes we will name **base**.
Structure of the description of base nodes of a fragment:

 # # BASE : *узел*$_1$ **[***узел*$_2$ **[... ***узел*$_n$**]] [;]**

узел$_j$ – number of a node of the fragment presented as base.

It is necessary to mean, that for any fragment the base node should be without fail certain. If the fragment of the lower level will be included further the description of structure of a fragment of an upper level and it is necessary to save attaching one of timbered nodes of a fragment of the lower level (necessarily present at the structure description) should be presented as exterior and in the description of a fragment of an upper level also is presented as base.

If the subsection is absent, in the capacity of the base the first node which has met in the description of structure is accepted.

Instance.

If in system in the capacity of the base nodes with numbers 5 and 6 after title $ FRAGMENT in the program there should be a subtitle # BASE are accepted:

```
# # BASE : 5, 6
```

THE DESCRIPTION OF STRUCTURE OF THE FRAGMENT – SUBSECTION # STRUCTURE

The subsection begins the subtitle with the same name:

```
# STRUCT [URE]:
```

After a subtitle on the next lines the description of structure of installation - the list of descriptions of elements or the fragments which have been switched on in the given fragment is given. Elements of this list - descriptions of elements switched on in a fragment or fragments - separate from each other the separator*;* or the extremity of a line. Each description of models of elements switched on in a fragment or fragments can place on several lines of the text of the program. Thus transferable line should be completed by the matching separator ("," - a comma, "'" - an apostrophe, ";" - a semicolon).

The description of models of the elements which are switched on in a fragment, looks like the following:

The identifier ' a name (узел$_1$*, узел*$_2$*..., узел*$_n$*; the list)*

The identifier — The identifier of model of an element.

Name — Name of model of an element from among the models which have been switched on in libraries of a complex.

узел$_j$ — Number of a node of a fragment to which it is connected j-я an

element branch. Node number is set as an arbitrary positive integer without the sign. For effective operation of the algorithms accepted in a complex it is better, if numbers of nodes are set without essential droppings. The amount of nodes of connection of model of an element should match to its description strictly.

The list — Argument list of model of an element. It is set by the rules defined earlier for title $ DATA. The amount and sequence of parametres should match completely to amount and a sequence of the parametres defined in the documentation on model of an element.

The description of turning on presented before fragments in a current fragment is made according to following syntax:

The identifier ' a name (узел$_1$, узел$_2$..., узел$_n$ [; ~идентификатор1 = ~идентификатор2...])

The identifier — The identifier of a fragment generated by common rules.

Name — Name of a fragment from among defined earlier.

узел$_j$ — Number of a node of a presented fragment to which it is connected j-й an exterior node of a switched on fragment. The amount and a sequence of nodes of connection of a switched on fragment should match strictly to amount and a sequence of the nodes defined in subsection #EXTERNAL of the switched on fragment.

~идентификатор1 — The identifier of the argument list defined for a fragment switched on in the description. This argument list is substituted for on an argument list which identifier is specified to the right of an equals sign.

~идентификатор2 — The identifier of the argument list defined for a current fragment which substitutes for all intromissions *~идентификатор1*, meeting in the description of a switched on fragment.
Attention! Those are substituted for and only those argument lists which are explicitly defined in the description of structure or in the description of output variables of a switched on fragment.

As separators for numbers of nodes, except commas, white spaces can serve. The compiler **discriminates** model of an element and a fragment, having an equal name.

Instance.

The description of models of the elements which are switched on in a fragment, can look as follows:

```
Spring ' K (1 2; 10);
       ' M (2; Mass)
Pendulum ' STRGNO (1,2,3,10;
                   Point And, 2., 2.5,
                   Material (2=0.7e11))
```

The description of the fragments which are switched on in a current fragment:

```
Spring ' the Stop (1 2);
```

```
Column ' S (1,2,3,10; the Point And = the Point
```

In reduced instances it is supposed, that earlier in sections of the description of data argument lists are defined:

```
Point And, the Material, Mass, the Point With,
```

Models of elements are switched on in complex composition:

```
K, STRGNO, M,
```

And in matching sections of the description of installation fragments are defined:

```
S, the Stop
```

Fragment S has 4 nodes presented in subsection #EXTERNAL, and a fragment the Stop - 2 nodes.

THE DESCRIPTION OF EXTERIOR NODES OF THE FRAGMENT - THE SUBTITLE # EXTERNAL

Syntax of the description **of exterior nodes** of a fragment (i.e. nodes of a fragment which serve for turning on of a presented fragment in fragments of higher level) is defined as follows:

 # EXT [ERNAL]: *узел*$_1$ **[***узел*$_2$ **[***узел*$_n$**]] [;]**

узел$_j$ - number of a node of the fragment presented as j-й an exterior node.

In line any whole amount of numbers of nodes can be recorded, and the separator necessarily should follow last node in line (if it is not last in the list) ",".

Instance of the description of exterior nodes:

```
#EXT: 3, {a base node}
       6,4 {a leg of the clown}
```

THE DESCRIPTION OF THE LEADING-OUT FOR THE FRAGMENT – SUBSECTION # OUTPUT

The subsection begins a subtitle:

```
# OUTPUT :
```

After a subtitle on the next lines the description of output variables - the list of descriptions of ringings of matching programs of calculation of output variables is given. Elements of this list - descriptions of ringings of separate programs of calculation of output variables - separate from each other the separator “;” (semicolon) or the extremity of a line. Each description of an output variable can place on several lines. In this case the transferable line should be completed by the matching separator ("," - a comma, "'" - an apostrophe, ";" - a semicolon).

The description of an output variable (= the description of a ringing of the matching program of calculation of an output variable) looks like the following:

The identifier ' a name (переменная$_1$ *[переменная*$_2$ *[..., переменная*$_n$*]]; the list)*

The identifier — The identifier of an output variable.

Name — Name of the program of calculation of output variables from among the programs which have been switched on in libraries of a complex.

переменная$_j$ — The index on j-ю the interior variable transmitted in the program of calculation of output variables. The amount and an order of indexes in the ringing description should match to their amount and a sequence in the documentation for the program of calculation of output variables. Syntax of the description of the index on an interior variable is defined more low.

The list — Argument list of the program of calculation of output variables. It is set by the rules defined earlier for title $ DATA. The amount and sequence of parametres should match completely to amount and a sequence of the parametres defined in the documentation for the program of calculation of output variables.

Admissible indexes on interior variables are:

узел$_j$ — Number of a node of the presented fragment, which potential performances are transmitted in the program of calculation of output variables (the basic potential variable and two its derivatives on a time). It is meant potential performances of a node:

In the mechanic - node migration *узел*$_j$ and its derivatives on a time;

In hydraulics, a pneumatics - integral on a time from pressure, pressure and its derivative on a time for *узла*$_j$;

In thermodynamics - integral from temperature, temperature and its derivative on a time for *узла*$_j$;

Generally - value of a variable (in which terms the solution of system of the differential equations is searched), characterising a node condition, and values of its derivatives on a time.

узел$_j$ ’ — The first derivative on a time of the basic potential variable *узла*$_j$ (accordingly a velocity, pressure, temperature, a potential).

узел$_j$ ” — Flexon on a time of the basic potential variable *узла*$_j$ (acceleration, a derivative from pressure, a derivative from temperature, a derivative from a potential).

I:*идентификатор* [(*ветвь*$_j$)]
F:*идентификатор* [(*ветвь*$_j$)]
Q:*идентификатор* [(*ветвь*$_j$)] — The index on **the data-flow variable** transmitted in the program of calculation of output variables. All three notations of the index on a data-flow variable are equivalent. For an element with the identifier *the identifier* characterises a condition *ветви*$_j$ this element. It is necessary to notice, that foliation of branches is local for an element, therefore value *ветвь*$_j$ should not exceed amount of nodes of this element.

So, for a two-nodal element ветвьj can accept values 1 or 2. By default it is supposed *ветвь*$_j$ = 1. It is meant a data-flow variable:

In the mechanic - force (moment) with which the system acts on an element. For definition of an indication of the index on a mechanical data-flow variable it is preferable to use the first or second notation of the index (with indications I or F);

In hydraulics, in pneumatics - the charge or the magnitude equivalent to it directed from system to an element. For definition of an indication of the index on such data-flow variable it is preferable to use the first or third notation of the index (with an indication I or Q);

In thermodynamics - the thermal stream directed from system to an element. As well as in the previous case, the first or third notation (with an indication I or Q) is preferable;

Generally - value of a variable (concerning which the conservation relations - 3rd Newton's law, a Kirchhoff's law for currents etc. are formulated), characterising a stream of this magnitude from system to an element. The first notation of a data-flow variable (with an indication I) is preferable.

W: *the identifier* [(*N*)] — The index on N-ю (by default - on 1st component) for an element with the identifier *the identifier*, which value is required to be transmitted a component **of a working vector** in the program of calculation of output variables.

Accessible components of a working vector of each model of an element are resulted in its description.

Instance. The description of ringings of programs of calculation of output variables can look as follows:

```
Force on a spring ' X (I:Пружина; 1.);
Migration on a X-axis ' S (3; 1);
Velocity on a X-axis ' V (3; the Scale);
Velocity on a X-axis ' X (3 '; the Scale);
Leg acceleration ' ROUT (1 ", 2"; 1);
Response in a pendulum leg ' ROUT (I:Маятник (1),
                                   I:Маятник (2); 1)
Energy of a spring ' X (W:Пружина; 1.);
```

In reduced instances it is supposed, that earlier in sections of the description of data the argument list is defined:

```
Scale,
```

Programs of calculation of output variables are switched on in complex composition:

```
S, X, V, ROUT,
```

And in matching sections of the description of installation elements with identifiers are defined:

Spring, Pendulum

THE HARDCOPY OF STRUCTURE OF THE GLOBAL FRAGMENT - THE SUBTITLE # MAP

In an assay value of the description of installation *PradiSLang* gains the information on structure of Jacobian determinant of system of the differential equations. Thus it always uses the interior foliation of nodes differing from foliation of nodes, set by the user. Besides, for decrease of computing expenditures after parse execution optimum renumbering of nodes as a rule is carried out.

For deriving of the information on definitive foliation of nodes the text of a global fragment switches on a subtitle

 # MAP:

In this case after parse execution on printing the list of models of elements, their identifiers and numbers of the nodes gained after execution of renumbering is given out.

All messages of programs of a computing kernel refer to these numbers of nodes and number of models of elements. Besides, this information is necessary in case of use of means DEBUG of the program of integration.

THE IMAGE OF INSTALLATION IN THE COURSE OF CALCULATION - SECTION $ SHOW

The section of the description of the image of installation in the course of calculation follows after global section of the description of structure directly ahead of sections of the description of the job. The section of the description of the image of installation begins with title:

$ SHOW:

After title on the next lines the description of the image of installation - the list of descriptions of images of the separate stratums which have been switched on in the image of the given fragment is given. Elements of this list separate from each other the separator (“;” - a semicolon) or the extremity of a line. The description of each stratum of the image can place in several lines of the text of the program. In this case each transferable line should be completed by the matching separator (“,” - a comma, “;” - a semicolon, “‘“ - an apostrophe).

In parametres of a stratum of the image the certain image scale and its colour are set. It is characterised by the certain point of view of the observer and, was possiblly, the description of driving of this point. The amount and sequence of parametres should match completely to amount and a sequence of the parametres defined in the description of the program of implementation of the image of a stratum (LAYER).

The description of a separate stratum looks as follows:

~идентификатор **' LAYER** *([[идентификатор$_1$] [(имя$_1$; список$_1$)]*
[, [идентификатор2] [(имя2; список2)]]...];
~список [; узел1 узел2... узел9])

~идентификатор — The stratum identifier.

идентификатор$_j$ — The identifier of model of the element, which graphical image is switched on, in the stratum description. If at the description of any graphical image the model identifier is not underlined, it is

considered, that the graphical image is motionless (is linked with motionless axes). If in the stratum description there is no description of any graphical images, this stratum switches on images of all elements of the installation having identifiers. In this case the image of each element is under construction with use of the graphical image accepted for this element by default (a "standard" graphical image).

имя$_j$ — Name of a graphical image if for the installation image the standard graphical image or a graphical image of the user is used not. If the name of a graphical image is not set, for the element image the graphical image of this element accepted by default is used.

список$_j$ — Argument list of a non-standard graphical image. It is set according to the rules defined earlier for title $ DATA. The amount and sequence of parametres should match completely to amount and a sequence of the parametres defined in the documentation under the program of implementation of a non-standard graphical image.

~список — Argument list of the program of implementation of the image of a stratum. It is set by the rules defined earlier for title $ DATA. The amount and sequence of parametres should match completely to amount and a sequence of the parametres defined in the documentation under the program of implementation of the image of a stratum.

узел$_j$ — The list of a degree of freedom with which the program of implementation of the image of a stratum is linked. If the list of a degree of freedom is not set, it is considered, that the program of implementation of the image of a stratum is linked with a base node (= with motionless axes).

The description of the image of installation can look as in the instances resulted more low.

Instance 1.

```
$ SHOW:
    The installation image ' LAYER (; stratum Parameters;
                                1 2 3 4 5 6 7 8 9)
```

In the reduced description the installation image switches on all elements having identifiers for which graphical images are defined. The argument list "stratum Parameters" is defined earlier in section $ DATA. The program of implementation of the image of a stratum is linked with nodes 1-9.

Instance 2.

```
The part image ' LAYER (the Spring,
                        Skew field (SILUET; a skew field Head loop);
                        Stratum parametres);
The bolt image ' LAYER (the Bolt (BOLT; 1.1,1,0),
                        (KONTUR; bolt Strengthening);
```

```
                                    Parametres of a stratum 2);
```

In the reduced description the installation image consists of two stratums. Identifiers of stratums - "the part Image" and "the bolt Image". Both presented stratums are linked with motionless axes.

Elements are switched on in stratum composition "part Image" with identifiers "Springs" and "Skew field". For the image of an element "Spring" the graphical image by default is used, for the image of an element "Skew field" the non-standard graphical image SILUET is used. Parametres of this graphical image are set by an argument list "a skew field Head loop", and stratum parametres - an argument list "stratum Parameters". The specified argument lists are certain in section $ DATA.

The element is switched on in stratum composition "bolt Image" with identifier "Bolt". For the element image the non-standard graphical image BOLT is used. Parametres of this graphical image are set by the list from three parametres. The element linked with motionless axes is switched on in this stratum of the image also. A name of this graphical image KONTUR, the argument list "bolt Strengthening" is presented in section $ DATA. Parametres of the program of implementation of the image "Parameters of a stratum 2" also are presented in section $ DATA.

THE JOB DESCRIPTION

The information on sections of the description of the job is resulted more low in that order in which they can follow in the program text in language *PradiSLang*. Thus it is necessary to mean, that in any case the program should be completed by title $ END.

SUBSTITUTION OF PARAMETERS IN THE GENERATED MODEL - SECTION $ REPLACE

The section $ REPLACE is applied to substitution of argument lists in already generated operating program. It is used in the event that it is not necessary to change model structure, the list of output variables and the description of the image of model during calculation. The section $ REPLACE allows to realise new calculation of model with the job of any new parametres of models of elements, programs of calculation of output variables or programs of implementation of the image without execution of new assemblage of an operating program. Act of instructions on substitution of the parametres resulted in section $ REPLACE, is saved only throughout current calculation (i.e., argument lists in a database of already generated model do not vary). If the following calculation for this model happens without REPLACE in it parametres from the initial text of the job will be used. The section begins with title:

```
$ REPLACE:
```

After title on the next lines the description of substituted for argument lists is given. One substituted for argument list separates from another the separator “;” (semicolon) or the extremity of a line. Syntax of the description of substituted for argument lists:

Name = the list

Name — Name of a substituted for argument list. **Attention!** Those are substituted for and only those argument lists which explicitly contain in the description of structure of a global fragment, the description of its output variables or in section of the description of the image of

installation during calculation.

The list — The argument list set by rules, defined for section $ DATA. The amount and a sequence of parametres in this list should match to amount and a sequence of parametres in the substituted for list.

The section $ REPLACE can follow after section $ DATA or to be always on the lips first in the text of program *PradiSLang*. In case of possibility use $ REPLACE the description of installation and the description of the image of installation during calculation in the program should be absent.

Instance of substitution of the argument lists which are explicitly present at the description of structure. The text of the job for an operating program looks as follows:

```
$ DATA:
Stiffness coefficient = 1.0E6
Value of a positive allowance = 1
Positive allowance parametres = Value of a positive allowance,
Stiffness coefficient
$ FRAGMENT:
# # BASE : 1
# # STRUCT :
Mass 1 ' M (2; 5)
Mass 2 ' M (3; 5)
Stop ' UPRL (2,3; positive allowance Parameters)
Spring ' K (1,2; the Stiffness coefficient)
Force ' F (3;-1E3)
# # OUTPUT :
Velocity ' V (3; 1)
$ $ RUN :
Calculation ' SHTERM (END=1)
$ $ PRINT :
Outcome ' DISP ()
$ END
```

Procedure of substitution of parametre "Stiffness coefficient" in the data unit for presented above a fragment:

```
$ REPLACE:
Stiffness coefficient = 90000
$ $ RUN :
Calculation ' SHTERM (END=1)
$ $ PRINT :
Outcome ' DISP ()
$ END
```

As a result for an element "Spring" the argument list will be redefined:

```
Stiffness coefficient = 90000
```

Because the substituted parametre is present explicitly at its argument list. Whereas for an element "Stop" in which argument list the substituted parametre contains not explicitly, the argument list will be set:

```
Positive allowance parametres = 1, 1.0E6
```

RESTORATION OF THE CONDITION OF CALCULATION - TITLE $ RESTORE

The title $ RESTORE is used in the event that for the given model of an engineering system integration with saving of a condition of calculation already was spent and it is required to continue calculation from the moment of last saving. The title can be always on the lips only first in the text of job *PradiSLang* or follow right after section $ REPLACE. In case of possibility use $ RESTORE the description of installation and the description of the image of installation during calculation in the program should be absent.

CALCULATION EXECUTION - SECTION $ RUN

The section begins title:

```
$ $ RUN :
```

This section in the program can be only one. If the title $ RUN follows after the description of installation or is first in the program text in language *PradiSLang*, calculation for the set model is executed from a zero instant. In this case outcomes of all previous calculations for the given model are lost. If the title $ RUN follows after title $ RESTORE, trying becomes to rebuild a condition of calculation from last place of saving.

After title $ RUN on the next lines there is a description of ringings of programs of integration. The description of separate ringings is disjointed by a numeral ";" (semicolon) or the extremity of a line.

The description of each of ringings of the program of integration is characterised by a matching counted slice of time, parametres of an exactitude and regimes of representation of the information on a calculation course. For each program of integration the list of output variables operatively represented during calculation from the list defined earlier in subsections # OUTPUT can be set. It is considered, that each subsequent program of integration continues calculation from that instant on which it has been discontinued by the previous program of integration. In complex *PradiSLang* there is a possibility online to operate end of the program of integration. In case of interactive interruption the agreement on calculation prolongation by the subsequent programs of integration remains in force.

Saving of a current condition of calculation happens to the temporary pitch set by the user and, automatically, upon termination of integration.

Now the complex composition includes programs of integration of system of the differential equations of II th order an implicit method of Stormer and method Ньюмарка. The description of a ringing of the program of integration looks as follows:

[~идентификатор] ' ~имя (имя$_1$ = number [имя$_2$ = number [... имя$_n$ = number]];
идентификатор$_1$ [(Number)] [= ([параметр11], [параметр12]...)]
[идентификатор$_2$ [(number)] [= ([параметр21], [параметр22]...)]
[идентификатор$_n$ [(number)] [= ([параметрn1], [параметрn2]...)]]]])

~идентификатор The identifier of the program of integration.

Name Name of the program of integration ("SHTERM", "NEWMARK").

имя$_j$ Name of j th key parametre from the list of the key parametres defined in the certificate of the program of integration.

Number The real number setting value of key parametre.

идентификатор$_i$ The full identifier i-й operatively represented variable from the list of the variables presented in a subtitle #OUTPUT. The full identifier of operatively represented variable should contain a full path from an upper level fragment to the characteristic identifier of a variable in subsection #OUTPUT a powered fragment. This path consists of sequence of identifiers of the fragments disjointed by a numeral "/". The sequence of enumeration of fragments matches to an order of connection of these fragments each other, thus in the sequence beginning fragments of higher level are.

Number Component number (for multicomponent represented variables). If component number is absent, it is supposed, that representation of the first component is set.

параметр$_{ij}$ j-й The positional parametre characterising i-ю operatively represented variable. Assignment and amount of these parametres is defined in the description of the program of integration. In case in an argument list j-й the parametre is let pass, its lack is noted by a comma. Lack of positional parametre means, that the integration program should take advantage of value of this parametre by default.

If it is necessary to switch on all list of variables defined by a population of all subsections OUTPUT of the program the ringing of the program of integration does not contain the list of separate represented variables in composition of operatively represented variables and looks like:

[~идентификатор] **' SHTERM** *(имя1 = number [имя2 = number [... имяn = Number]])*

Note Values of the key parametres which are not meeting in the description a ringing of the program of integration, are accepted by default equal to the values of these parametres specified in the certificate of the program of integration.

Instance. Descriptions of ringings of the program of integration can look as follows:

```
Calculation of oscillations of a pendulum ' SHTERM (END=1, OUT=1.e-3,
                                          SAVE=0.1)
  {Calculation prolongation}
                                ' SHTERM (END=1.5)
```

REPRESENTATION OF OUTCOMES - SECTION $ PRINT

The section begins title:

```
$ $ PRINT :
```

This section in the program can be only one. If the title $ PRINT is first in the program text, it is supposed, that is required to realise representation of outcomes for spent before calculation. If the title $ PRINT follows after title $ RUN representation of outcomes for current calculation is spent.

Presence of this section in the program is obligatory.

After title on the next lines there is a description of ringings of programs of representation. The description of separate ringings is disjointed by a numeral ";" (semicolon) or the extremity of a line.

The description of a ringing of the program of representation looks as follows:

[~идентификатор] ' ~имя (имя$_1$ = number [имя$_2$ = number [... имя$_n$ = number]];
идентификатор$_1$ [(Number)] [= ([параметр11], [параметр12]...)]
[идентификатор$_2$ [(number)] [= ([параметр21], [параметр22]...)]
[идентификаторп [(number)] [= ([параметрп1], [параметрп2]...)]]]])

~идентификатор — The identifier of the program of representation.

Name — Name of the program of representation from the program list of the representation, switched on in complex composition.

имя$_j$ — Name of j th key parametre from the list of the key parametres defined in the documentation under the program of representation.

Number — The real number setting new value of key parametre.

идентификатор$_i$ — The full identifier i-й a represented variable from the list of the variables presented in subtitle OUTPUT. The full identifier of a represented variable should contain a full path from an upper level fragment to the characteristic identifier of a variable in subsection OUTPUT of a powered fragment. This path consists of sequence of identifiers of the fragments disjointed by a numeral "/". The sequence of enumeration of fragments matches to an order of connection of these fragments each other, thus in the sequence beginning fragments of higher level are.

Number — Component number (for multicomponent represented variables). If component number is absent, it is supposed, that representation of the first component is set.

параметр$_{ij}$ — j-й The positional parametre characterising i-ю the represented variable. Assignment and amount of these parametres is defined in the certificate of the program of representation. In case in an argument list j-й

the parametre is let pass, its lack is noted by a comma. Lack of positional parametre means, that the representation program should define its value automatically.

If it is necessary to switch on all list of variables defined by a population of all subsections #OUTPUT of the program the ringing of the program of representation does not contain the list of separate represented variables in composition of represented variables and looks like:

[~идентификатор] ' ~имя (имя$_1$ = number [имя$_2$ = number [... имя$_n$ = number]])

Instance. Descriptions of ringings of programs of representation can look as follows:

```
The basic performances of process ' DISP (;
           Force on a spring = (0, 1.e6),
           Migration on a X-axis = (10, Amplitude),
           Velocity on a X-axis,
           Leg migration = (0.001))
All parametres of process ' DISP (END=1.)
The table of separate components ' TABL (OUT=1.e-1, START=0.5;
           Spring/deformation (2) = (0,),
           Index травмируемости (3))
```

In reduced instances it is supposed, that earlier in sections of the description of data the argument list "Amplitude" is defined, in library of programs of representation there are programs DISP, TABL, in subsection OUTPUT of a fragment c identifier "Spring", the upper level switched on immediately in a fragment, the leading-out of a multicomponent variable "Strain" is defined, in subsection OUTPUT of a fragment of an upper level the leading-out of variables "Force on a spring", "Migration on a X-axis", "the Velocity on a X-axis", "Migration to a leg", "the Index травмируемости" is defined.

POSSIBLE SEQUENCE OF SECTIONS AND SUBSECTIONS IN THE PROGRAM IN LANGUAGE PradiSLang

As follows from the present deed, the majority of sections and subsections in the program in language *PradiSLang* can appear in strictly certain order. Here it is represented pertinent to result a possible sequence of sections and subsections.

It is possible to gate out three types of programs differing from each other in language *PradiSLang*. It, at first, the program providing shaping of new model of installation with the subsequent task execution. Secondly, it is the program providing execution of calculation for already generated model. And, thirdly, it is the program providing representation of outcomes gained earlier for model.

In the list of enumeration of sections and subsections numerals (***) mark out obligatory sections and subsections. Sections and the subsections noted by numerals (**) should appear in the program at least once, noted by numerals (*) are not obligatory, but their lack involves delivery of the notifying message.

Sequence of sections and subsections in case of shaping of new model of installation:

1. **$ DATA** indeterminate (global)
2. **$ DATA** named and **$ FRAGMENT** the named.

The concrete data unit necessarily precedes matching fragment.

Sequence of subsections in section $FRAGMENT:

(*) 2.a **# BASE**
(***) 2.b **# STRUCTURE**
(**) 2.c **# OUTPUT**
2.d **# MAP** and **# EXTERNAL**

For # MAP and # EXTERNAL the strict order is not defined, but the subsection # STRUCTURE necessarily should precede them.

(***) 3. **$ FRAGMENT** concrete or indeterminate (global).
4. **$ SHOW**
(***) 5. **$ RUN :**
(***) 6. **$ PRINT :**
(***) 7. **$ END**

Sequence of sections in case of execution of calculation for already generated model:

1. **$ DATA** indeterminate (global)
3. **$ REPLACE**
(*) 4. **$ RESTORE :**
(***) 5. **$ RUN :**
(***) 6. **$ PRINT :**
(***) 7. **$ END**

Sequence of sections in case of a task execution on representation of outcomes:

1. **$ DATA** indeterminate (global)
(***) 3. **$ PRINT :**
(***) 4. **$ END**

USE OF THE INSTRUCTION OF THE PREPROCESSOR $ INCLUDE

For turning on in the text of the current job of the text of other file, the preprocessor instruction $INCLUDE is used.

Instruction format:

`$ INCLUDE:` *имя_файла*

In a line containing $INCLUDE, there should not be other sentences of an input language.

All contents of the specified file are switched on in that place of the program where this instruction is had. If the switched on file itself contains instructions $INCLUDE before turning on of this file in the text of the current job instructions $INCLUDE for the text in a switched on file are carried out. The enclosure of instructions $INCLUDE is not restricted. Instructions $INCLUDE can be used in any place of the program. With their help any fragment of the program can be switched on in the treated job on input language *PradiSLang*.

The instruction $INCLUDE is executed at a stage препроцессорной job machining then text parse follows. In case specified for turning on in the text of the current job the file will not be detected by a preprocessor, the matching message and machining of the current job is given out stops.

RUSSIAN-SPEAKING SYNONYMS OF TITLES AND SUBTITLES OF LANGUAGE PradiSLang

For an applications programming there can be useful Russian-speaking synonyms of titles and input language subtitles:

$ DATA:	$ DATA:
$ FRAGMENT:	$ FRAGMENT:
# # BASE :	# BASELINE:
# STRUCTURE:	# STRUCTURE:
# # OUTPUT :	# LEADING-OUT:
# EXTERNAL:	# THE EXTERIOR:
$ SHOW:	$ SCANNING:
$ $ RESTORE :	$ TO CONTINUE:
$ REPLACE:	$ TO SUBSTITUTE:
$ $ RUN :	$ TO EXECUTE:
$ $ PRINT :	$ TO REPRESENT:
$ $ END	$ THE EXTREMITY

Rules of use of Russian-speaking synonyms completely are equivalent to rules of use of matching basic constructions of an input language.

INSTANCES OF PROGRAMS IN LANGUAGE PradiSLang

THE PROGRAM CONTAINING THE DESCRIPTION OF INSTALLATION AND THE DESCRIPTION OF THE JOB FOR CALCULATION AND REPRESENTATION OF OUTCOMES

Mathematical model of a rod pendulum with an elastic leg and concentrated to the pendulum extremities pointwise inertia elements. With a pendulum axis through the reductor the propeller incorporates to a linear speed-torque characteristic.

```
$ DATA:
    {The description of global data}
  Modulus = 2.E11 {the Pas};
  Scale = 1.; {a scale for inferred magnitudes}
$ DATA: the Drive
    {The description of data for a fragment the Drive}
  Propeller starting torque = 45 {н*м}
  No-load operation velocity = 2 {1/с}
  К.п.д. The reductor = 0.95;
  Reduction ratio = 2
  The nominal moment = 100.;
  Rigidity = 1.E6 {н*м}
  Propeller parametres =
  Starting torque of the propeller,
  No-load operation velocity
  Reductor parametres =
  Reduction ratio,
  К.п.д. The reductor,
  The nominal moment, Rigidity
{Curl and flywheel moments of inertia are set in the description of
structure of the drive.}
$ FRAGMENT: the Drive
     # # BASE : 1 {Base - the first degree of freedom}
     # # STRUCT :
          The propeller ' DVL (2 1; propeller Parameters);
          Curl ' M (2; 0.1);
          The reductor ' REDN (2 3; reductor Parameters);
          Flywheel ' M (3; 1);
     # EXTERNAL: 1, 3
     {A degree of freedom on which the presented fragment
incorporates
       With other fragments of system}
     # # OUTPUT :
        Propeller angular velocity ' V (2; the Scale);
$ DATA: Pendulum
{The description of data for a fragment the Pendulum}
  Point A = 0., 0; Point B = 1,0.; {co-ordinates of points}
  Material = 1, 1, the Modulus; {J, A, E}
  Mass = 10.; the Moment of inertia = 0.1;
  Rigidity of a leg = 1.E4 {н*м}
  Gravity(1); 1)
$ FRAGMENT: Pendulum
     # # BASE : 1
     # # STRUCT :
        Leg X ' K (2 1; Rigidity of a leg); Leg Y ' K (3 1;
                                          Rigidity of a leg)
        The rod ' BALKAD (2 3 4 5 6 7; Point A,
                                       Point B,
```

```
                                          Material)
    {Pointwise inertia elements}
             ' M (2; Mass); ' M (3; Mass);
             ' M (5; Mass); ' M (6; Mass);
                            ' M (7; the Moment of inertia);
    {Turning on presented above a fragment "Drive"}
     The drive ' the Drive (1 4);
    {A gravity}
     Force in a leg ' F (3; the Gravity);
     Force on the free extremity ' F (6; the Gravity);
  # # OUTPUT :
     The moment on the propeller ' X (I:Двигатель; the Scale)
     Migration to a leg on a X-axis ' S (2; the Scale)
     Migration to a leg on a Y-axis ' S (3; the Scale)
     Pendulum angle of rotation ' S (4; the Scale)
     Migration of the free extremity X ' S (5; the Scale)
     Migration of free extremity Y ' S (6; the Scale)
     The moment on the day off to the shaft редуктора'X (I:Редуктор
(2); 1.)
     Response in a leg on a X-axis ' X (I:Опора X; the Scale)
     Response in a leg on a Y-axis ' X (I:Опора Y; the Scale)
$ $ RUN :
  {The job for transient calculation}
     Calculation of behaviour of a pendulum ' SHTERM (END = 1.,
                                        {To consider 1c process}
                                         SAVE = 0.1)
{To spend saving of a current condition of calculation everyone
0.1c
                                    Process}
$ $ PRINT :
Outcomes of calculation of a pendulum ' DISP ()
    {To build graphs for all inferred magnitudes}
$ END
```

THE PROGRAM REALIZING PROLONGATION OF CALCULATION FROM LAST PLACE OF SAVING OF OUTCOMES

```
$ RESTORE {to rebuild a calculation condition}
$ $ RUN :
   Calculation prolongation ' SHTERM (END = 1.5)
$ $ PRINT :
   Outcomes of calculation of a pendulum ' DISP (;
                                    Pendulum angle of rotation
    {To build the graph only this magnitude,
                 Process of scaling automatic})
$ END
```

THE PROGRAM CONTAINING THE JOB FOR REPRESENTATION BEFORE GAINED OUTCOMES OF CALCULATION

```
$ $ PRINT :
     Propeller parametres ' DISP (;
                                      Angle of rotation of a pendulum,
                                      The moment on the propeller,
 {The scale is defined automatically}
                  The drive / an angular velocity of the propeller = (3)
 {The lower boundary line of the graph is defined automatically,
  The overhead is set (= 3).})
Pendulum ' DISP (;
                               Angle of rotation of a pendulum,
                               Response in a leg on a X-axis,
                               Response in a leg on a Y-axis)
$ END
```