"Vasile Alecsandri" University of Bacău
Faculty of Sciences
Scientific Studies and Research
Series Mathematics and Informatics
Vol. 19 (2009), No. 2, 375 - 392

THE RODIN MODULAR LANGUAGE V. 21 AUG 2009 – USER MANUAL AND REPORT

DAN POPA

Abstract. The scope of this paper is to provide news concerning The Rodin Project (http://www.haskell.org/haskellwiki/Rodin) – a national specific modular didactic language actually used as a helping tool in teaching base of computer science in high school and universities. The problem of producing enough programmers is actual and is a necessary step in order to assure the future development of the IT industry, services and software infrastructure. Rodin is dedicated to the teaching of C-like language's concepts, a wide used set of languages. The Rodin Language is specific designed to cross the language barrier which appears when students without any knowledge of English Language are supposed to quickly learn structured programming. The Rodin Language was release in aug. 2008. Teachers and students are encouraged and invited to contribute in order to build a corpus of Rodin Programs, based on the model of Free Software Groups. The sources written using Rodin are actually available free of charges from its website [5]. Rodin is used by The Faculty of Mathematics of Bacău University and also by some high schools from Bacău and Iasi area. The papers contains information concerning several aspects of the project, visible at users level: syntax, examples, differences, notes .

This community project dedicated to teachers – The Rodin Language – and, of course, the Rodin Language itselfis are presented below.

Keywords and phrases: domain small language, DSL, open source; community; Rodin, C-like languages, structured programming.
(2000) Mathematics Subject Classification: 68-01, 68N15

1. WHY CAN PROGRAMS WRITTEN USING RODIN BE CLASSIFIED AS AN OPEN SOURCE INITIATIVE

- the free distribution of such programs via website [5], everybody being encouraged to use Rodin and produce good quality teaching materials based on it. The Rodin licence is not published yet, but, till this version, Rodin is covered in fact by a sort of BSD licence.
- the source of the Rodin programs are distributed. Rodin being actually implemented as an interpreter like in [1], it ru ns sources directly and supports and encourage the study of the sources by reprinting them – on screen – as part of the running process.
- there are no limitations for derived works, till now, excepting the obligation of mentioning the Rodin web page and author and also the author of the previous works. Other legal obligations applies too if any.
- every teacher, student, may benefit of the latest published versions of Rodin programs and the released sources.
- By its design, The Rodin Language is made for teaching computer science, but we don't forbid any other utilisations of it. If next versions will be good for example for game design or for applications – well – why not?
- Rodin did not have a specific IDE now and various editors and IDE can be used: Total Edit, Ultra Edit etc. Therefore we did not forbid the integration of the Rodin Interpreter itself and the Rodin Programs with or in any other tools.
- Various versions of Rodin were built using various platforms: Linux, Wine, Windows (Tm). So there is no restriction concerning the O.S. Nobody requested Mac OS versions of Rodin but we are ready to produce a Rodin 4 Mac if needed.
- Rodin is build as a modular interpreter in Haskell [2], [3] – also a free software project. If anybody wants to rebuild Rodin using old technologies like TPLex and TPYacc for example, it is not forbidden, but such person is warned that modularity will be lost, probably gaining speed instead of it. Because we intend to develop Rodin by adding language modules we are not recommending to rebuild Rodin on other technologies, but it is not forbidden.
- Rodin syntactic specification can also can be used instead Tiger or While language in The Compiler Construction Course. But, being of the level of the third academic year, it's highly improbable that students did not understand English.

- Rodin module's structure will – probably - be available for those Haskell programmers interested in building any needed language plug-in. It's not the case yet. The theories are also published in the Haskell Community [3], also a Open Source Free Software – BSD Style – Community.

2. THE RODIN TEACHER'S COMMUNITY

- Actually there exists one point where interested teachers may go in order to find Rodin resources: Programs, Teaching Examples/Samples, Syntax specification, News, Advices etc. It is the Rodin Website. Http://www.haskell.org/haskellwiki/Rodin [5] Separated pages are made for: News, Downloads, Questions, Programs etc. The infrastructure is in fact a wiki but the access is possible only using a password, the same password provided for [3].
- Teachers are encouraged to build their own sets of Demo Programs in Rodin in order to show the concepts of Structured Programming to their students. We, at Bacău University, have also developed some chapters of course for the future teachers in Mathematics and basics of Comp. Sci. as part of a Course called: Fundamentals of Programming Languages. Rodin was used during the academic year 2008-2009 and was welcome by our students. Such students are and will be the first generations of members of the community.
- As a books writer I had the idea of placing some of manuals of mine under free licences as Open Documents. They will help us in producing Rodin Language Manuals.
- The starter kit of Rodin consists in a binary interpreter and open/public sources of some programs, showing the main Structured Programming Concepts in Romanian. Students and even kids are able to read such sources, without any knowledge of English. A program like:

  ```
  { citeste x; scrie x }
  ```

  will be easy understand by a native as

  ```
  { read x; write x}
  ```

  without any problems.

  The starter kit is provided as an archive containing the Rodin
Language
  Interpreter and some directories containing sources and papers.
- Help is provided via Pidgin (an other Open Source Project) using the Yahoo Mail accounts of the users from the community. We even give

advices by phone, for the system administrators interested in installing Rodin in School's Laboratories.

This set of aspects is supposing to give you a basic idea concerning the Rodin Teachers Community.

3. THE RODIN LANGUAGE ITSELF, CHARACTERISTICS OF THE VERSION 21/08/2009

Being considered a teaching tool, this version is having some distinctive characteristics:

– during the summer of 2009 the main source of Rodin itself was "sliced" in many modules, as part of a research work, also, in order to help the development and the revision. The built of a modular language itself is actually an open research area, but it will be subject of a technical paper.
– a difference: the syntax of the sequences was changed, being now closed to a mixture of Pascal and C. A sequence did not require the semicolon after the last statement. The begin and the end are marked with { } as C-like languages does.
– every module of the parser was triple checked. Teachers can count on a better parser and clear error messages.
– the operators: >=, <=, ==, != are included. Also: ! - the negation, && (SI), || (SAU)
– the "text" command was improved: Special characters like: + - _ ( ) . , ? ! : = helps user in order to formulate clear messages. Also the @ sign was included in text's specification. The user can program messages containing e-mail addresses.
– better error reporting messages, missing keywords are correctly and completely announced now
– C-like logic: 0 meaning False and other integers meaning true.
– the "let" statement called "fie" remains in place but it's restricted to simple identifiers – on the left side.
– the name of the running program is also sent to the console output.
– if really needed, the sequences of statements may be separated by « , » too, not only by semicolon.
– commands and expressions are know separated syntactic categories
– vectors, indexed variables
– the "let" statement called "fie" where the left side is an indexed

variable
- the "for" statement called "pentru" has a slight different syntax:

```
pentru (<com> ; <exp>;  <com>)
    <com>
```

Removed characteristics: Don't count yet again on concepts like:
- records – was not implemented at all, in any version
- files – also not implemented
- anonymous 1-parameter functions expressed by abstractions
- the apply invisible/unwritten operator
- the null sequence {}
- real numbers

4. The Rodin Language itself, syntax of the version 21/08/2009, codename: ExperimentExp11

4.1 The **I/O operations** are,yet, console based. There exists a sort of "read" called "citeste", a sort of "write" called "scrie", and also a sort of "writeStr" called "text". Examples:

```
citeste x

scrie  y

text "dati valoarea lui x:"
```

The strings may contain letters, digits and some extra characters, which are very helpful in order to make simple sentences: ! ? , . = @ _ - or to speak about e-mail addresses.

4.2 Assignments: Values are assigned to variables using a "let" statement as in Basic. Its syntax is: let <var> = <exp> where the expression may contains any kind of operators: +, -, * , /, > , < , >=, <=, !=, ==, ! .

```
fie x=1;
fie y=x+1;
fie z=(x+1)*(y+2);
fie logic=(z<=10);
fie negat=! (z<0);
```

Nota: && si OR nu sunt implementati in aceasta versiune.

4.3. The **"begin... end"** block statement is replaced by "{ ...... }", where single statements can be separated using ";" and also "," (not recommended but also possible).

```
{citeste x;
  scrie x }
```

Notati:Nu este permis spatiul de dupa "cand".

```
{text "dati valoarea lui x:";
  citeste x;
  scrie x }
```

Notati:Nu este permis spatiul de dupa "cand".

Some programs using the translated version of the "begin ... end" sequence, inspired by C-like languages.

4.4. The "if ... then ...else..." becomes "daca ... atunci ... altfel". A simplified version: The "if...then..." become "daca...atunci..." and it is also usable.

```
{ daca (1==1) atunci scrie 10 altfel scrie 0 }

-- daca1.txt
-- Comparatii: egalitatea scrisa cu doua semne egal
```

```
{ citeste x;
  citeste y;
  daca (x==y) atunci scrie 10 altfel scrie 0 }
```

-- daca2.txt
Se pot compara si variabilele, si expresiile... Orice expresie intreaga poate fi conditie.

```
{ citeste x;
  citeste y;
  daca (y!=0) atunci scrie x/y altfel scrie 0 }
```

-- daca3.txt
Comparatia "diferit" scrisa in stil C.
Impartirea intreaga /.
Se pot compara si variabilele, si expresiile...
Orice expresie intreaga poate fi conditie.

```
{text "Dati urmatorul y ";
citeste y;
text "Dati urmatorul xm ";
```

```
citeste xm;
executa {
   {daca (y>xm)
    atunci fie xm=y    };
atat cat (y!=0);
}
```

```
{text "Start program: dati x, ENTER, y si ENTER";
 citeste x;
 citeste y;
 daca (x>y) atunci text "x mai mare ca y"
 altfel text "x mai mic sau egal cu y";
 text " apasa 0 si Enter";
 citeste z
 }
```

Modular Language written by Dan V Popa, Ro/Haskell Group.
8/aug/2009 - Rodin - Codename:ExperimentExp8
limitare :{ <com> ; <com> ... <com> } fara ; final.

Some Programs using the alternative (i.e. Conditional) statement.

4.5. The usual "while" keyword is replaced by two keywords: "cat timp". Spaces are allowed between the two keywords. The space between the second keyword and the block of statements, theoretically accepted is not allowed in the actual implementation.

```
{citeste x;
 cat timp(x>0)
   { fie x = x /2;
     scrie x }
}
```

Un numar este impartit repetat la 2.
Rodin Modular / 8.08.2009/ ExperimentExp8
Atentie, aceasta versiune de while nu mai are "executa".
Notati:Nu este permis spatiul de dupa "timp".

```
{ fie x=100;
  cat timp(x>10)
    fie x=x-1;
  scrie x;
  text "Salut!"
}
```

Nu puneti spatiu dupa "timp".
Nu-l va accepta.
Revizuiti sursele vechi.

```
{ text "Calculul lui n! pentru n= ...";
  citeste n;
  fie x=1;
  fie nr=1;
  cat timp(nr<n)
    { fie nr=nr+1;
      fie x=x*nr
    };
  scrie x
}
```

Modular Language written by Dan V Popa, Ro/Haskell Group.
8/aug/2009 - Rodin - Codename:ExperimentExp8
limitare :{ <com> ; <com> ... <com> } fara ; final.
Programul:RodinV08-Factorial-Ro.txt

```
{ fie y=2;
  fie x=100;
  cat timp(x>10) {
    fie x=x-1;
    scrie x
  };
  scrie y;
  text "Salut!"
}
```

Numaratoarea descendenta:
Bucla cu test initial cu
mai multe instructiuni in bucla.

```
{ citeste n;
  fie f1=0;
  fie f2=1;
  scrie f1;
  scrie f2;
  cat timp(f2<n)
    { fie f1p=f2;
      fie f2p=f1+f2;
      fie f1=f1p;
      fie f2=f2p;
      scrie f1
    }
}

-- 7 aug 2009. Fibo.
-- Refacut cu ocazia Exp 07
-- fara spatiu dupa timp(
-- fara ; dupa ultima instructiune
```

Some Programs using the romanian version of the "while" loop.

4.6. The **"do... while ..." statement** is replaced by "executa.... atat cat ". Spaces are allowed between the two keywords. The space between the second keyword and the expression, theoretically accepted are not allowed in the actual implementation.

```
 { text "  Maximul elementelor unui sir de numere ";
  text "pozitive distincte terminat cu numarul zero. ";
  fie xmax = 0;
  text "dati y ";
  citeste y;
  executa {
    {daca (y>xmax)
     atunci fie xmax=y
    };
    text "dati urmatorul y ";
    citeste y }
  atat cat (y!=0);
  text "maximul este ";
  scrie xmax
}

--Rev 9 aug 2009 pt ExperimentExp8
--Instructiunea
```

```
                        executa ... atat cat ...
 Este echivalentul lui do... ..........while ... din C.
 Primul loc: o instructiune (compusa eventual)
 Al doilea: conditia

 -- Instructiunea daca ... atunci...
 fara alternativa:altfel
```

A Program using the translated version of the "do... while ...." loop, which is specific for the C-like languages.

4.7. The **"for"** keyword is replaced by "pentru". Dual and multiple counters loops are allowed.

```
 {pentru (fie x=1;  x<10;  fie x=x+1)
    scrie x
 }

 --Rodin, 8 aug 2009, Exp8

 {pentru (
        {fie x=1,fie y=2};
        x*x<100;
        {fie x=x+1,fie y=y*2}
        )
          {text "x=";
           scrie x;
           text "y=";
           scrie y;
           text " "}
 }

   Modular Language written by Dan V Popa, Ro/Haskell Group.
 8/aug/2009  -       Rodin        - Codename:ExperimentExp8

   Programul:bucladubla.txt.
 La instructiunea for este nevoie de acolade la comenzile
 c1, c3, c4 din
   for ( c1 ; e2 ; c3 ) c4
 Se pot scrie si acele ciudate bucle cu doua contoare.
```

Some programs using the translated version of the "for" loop.

4.8. The **"repeat... until ....."** statement is replaced by "repeta ...pana cand.....".

```
{citeste x;
 repeta
   { fie x = x /2;
     scrie x }
 pina cand(x==0)
}
```

Rodin Modular / 8.08.2009/ ExperimentExp8
Notati:Nu este permis spatiul de dupa "cand".

```
{ text "Calculul divizorului comun";
  text "dati numarul a ";
  citeste a;
  text "dati numarul b ";
  citeste b;
  fie undeimp=a;
  fie unimp=b;
  repeta
    { fie unrest=undeimp%unimp;
      fie undeimp=unimp;
      fie unimp=unrest
    }
  pina cand (unimp==0);
  text "Dati divizorul comun:";
  scrie undeimp
};
```

Program 4:cmmdc

```
==================
Modular Language written by Dan V Popa, Ro/Haskell Group.
8/aug/2009  -       Rodin       - Codename:ExperimentExp8
```

Some programs using the translated version of the "for" loop.

5. VECTORS (NO MATRIX ALLOWED IN THIS VERSION: EXPERIMENTEXP11)

Rodin (Codename: ExperimentExp11) is using simple indexed Integer variables. Some things may be noted:

- there is no need to declare vectors
- but every element should be initialized , for example using the 0 value
- such an indexed variable can be placed as the left part of an assignment, an other kind of assignment being defined
- vectors may be, somehow, partially defined, for some values of the index the locations exists, but for other, no. They are usually called rare vectors (similarly with rare matrix)
- vectors being undeclared, the domain of the index is composed by the set of integers
- the values are also integers, no matter how long
- being introduced as a version of factors, (i.e. being embedded in the expression's evaluator, elements of a vector can be used inside every expression, even compared if needed. As a consequence, such values of indexed variables may be written by the usual statement.

```
{fie x[1]=1001;
 scrie x[1]
}
```

Pt Rodin ExperimentExp10
din 12/8/2009
Program: vector1.txt
====================

```
{ citeste y;
  fie x[0]=y;
  fie x[1]=y;
  citeste x[1];
  scrie x[1]
}
```

Rodin Exp10 12/09/2009

Vector2.txt
======================

Nu uitati sa initializati elementele vectorului inainte de folosire.

```
{fie x[1]=0;
 citeste x[1]
}
```

Rodin Exp10 12/08/2009

Vector3.txt

```
========================================
Elementele vectorului trebuie initializate inainte de a fi folosite,
chiar daca e vorba de o citire.

{text "dati lungimea vectorului:";
 citeste x;
 pentru (fie i=0; i<x ;fie i=i+1)
   {fie v[i]=0;
    citeste v[i];
    text "v[i]=";
    scrie v[i]
   }
}
```

Rodin Exp10 12/08/2009

Vector4.txt

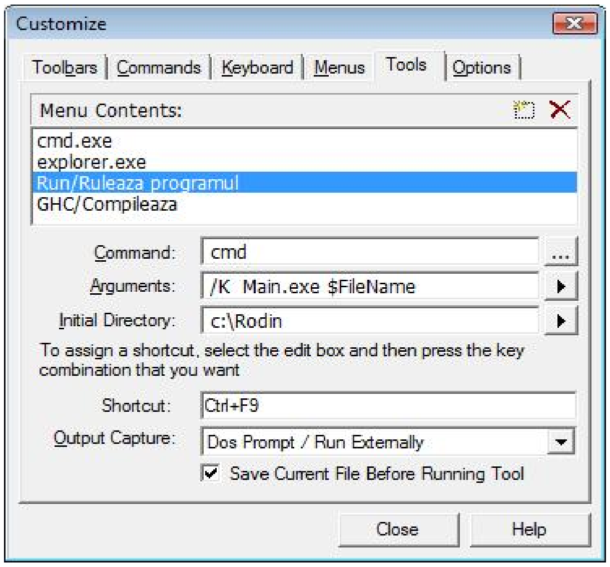
```
==========================================
Initializarea se poate realiza si din mers.
```

The examples (Vector1.txt – Vector4.txt) underline how ca vectors be used in Rodin (Experiment 10 and Experiment 11).

6. Running programs using Total Edit on Windows platforms

The Rodin programs are stored in common texts files and can be edited with any editor supporting the txt file format. Simply run the Main Rodin Binary from a console or from the menu of the editor, where can be easily added (Ultra Edit, Total Edit, X Emacs etc...).The name of the program is given as a single parameter. The same procedure is used on various operating systems: Window, Linux etc.

In order to use Rodin and Total Edit together, there is a need to define a new command in the Tools menu.



Menu Contents:
cmd.exe
explorer.exe
Run/Ruleaza programul
GHC/Compileaza

Command: cmd
Arguments: /K Main.exe $FileName
Initial Directory: c:\Rodin
To assign a shortcut, select the edit box and then press the key combination that you want
Shortcut: Ctrl+F9
Output Capture: Dos Prompt / Run Externally
[✓] Save Current File Before Running Tool

The user must define a new entry in the Tools Menu, calling it as he wish (for example: Run/Ruleaza programul). When used, it will start a command window running the main binary of Rodin (Main.exe) using the current edited file as an input file.

In this configuration the default directory and the place where Main.exe can be found is a directory called Rodin, on drive c: .

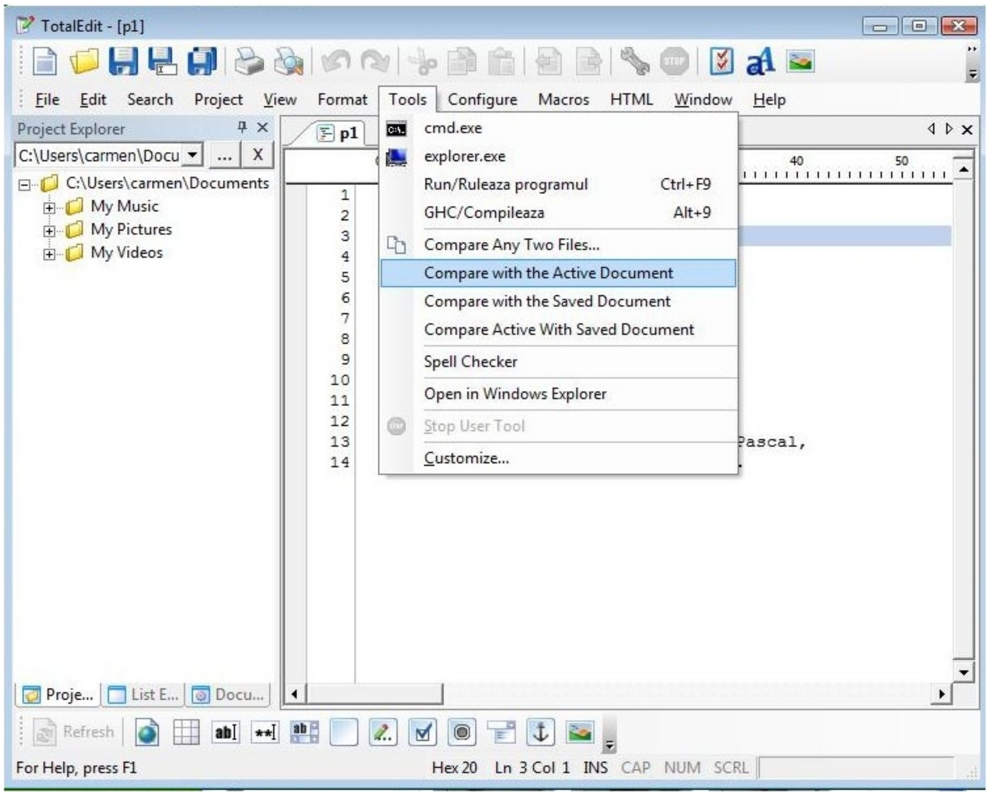
The user should be careful to save the files using the specified directory.

A shortcut can also be defined, for example CTRL+F9 by pressing the keys.

The "cmd" program has to be used from the "Dos Prompt or Externally" and the file should be saved automatically in order to reflect the latest changes of its content.

As a result, the user can Run Rodin programs with a CTRL-F9 combination of keystrokes.

The resulting windows have to be closed after the program had run. I.e. It did not automatically close itself.


TotalEdit - [p1]
File Edit Search Project View Format Tools Configure Macros HTML Window Help
Project Explorer
C:\Users\carmen\Docu
C:\Users\carmen\Documents
My Music
My Pictures
My Videos
p1
cmd.exe
explorer.exe
Run/Ruleaza programul Ctrl+F9
GHC/Compileaza Alt+9
Compare Any Two Files...
Compare with the Active Document
Compare with the Saved Document
Compare Active With Saved Document
Spell Checker
Open in Windows Explorer
Stop User Tool
Customize...
Pascal,
Proje... List E... Docu...
Refresh
For Help, press F1
Hex 20 Ln 3 Col 1 INS CAP NUM SCRL

Total Edit and Rodin together. The user defined menu helps running Rodin Programs.

7. Running an usual program

Because Rodin inherits its long arithmetic from its development language Haskell [2] and the integers implementation was made via the data declaration [2], Cap 3 , pp. 57-66 using Integers (the type of long integers available in Haskell), Rodin can also manipulate long arithmetic.

Because on space constraints we did not logged a big list of prime numbers below, but they can be found and stored, using vectors and long arithmetic.

```
[dan@localhost ExperimentExp11]$ ./Main Eratostene.txt
Modular Language written by Dan V Popa, Ro/Haskell Group.
21/aug/2009 -      Rodin      - Codename:ExperimentExp11
```

Noutate:Operatorii logici SI &&, SAU || sunt implementati.
 Limitare: nu scrieti serii lungi de SAU si nici de SI cum scrieti la adunare serii lungi de plus: 1+1+1+1+1 ori 1+2+7.
Programul:Eratostene.txt

```
{ text "Ciurul lui Eratostene";
  fie max=20;
  pentru (fie i=1; i <= max; fie i=i+1)
     fie c[i]= i
     ;
  text "Ciurul este plin de numere";
  fie nrcrt=2;
  repeta {
     text "Sterg multiplii numarului;";
     scrie nrcrt;
     pentru (fie j=nrcrt; j <= max ; fie j = j + nrcrt)
        fie c[j]=0
        ;
     text "Au trecut prin sita.";
     text "trec la numarul urmator";
     repeta
        fie nrcrt=nrcrt+1
     pina cand (c[nrcrt] != 0 SAU nrcrt == 101);
     scrie nrcrt
     }
  pina cand (nrcrt >= max);
  text "Am terminat de trecut prin sita numerele";
  pentru (fie p=2; p <= max; fie p=p+1)
    {  scrie p;
       text " , "
    }
}
```

```
"Ciurul lui Eratostene"
"Ciurul este plin de numere"
"Sterg multiplii numarului;"
2
"Au trecut prin sita."
"trec la numarul urmator"
3
```

```
"Sterg multiplii numarului;"
3
"Au trecut prin sita."
"trec la numarul urmator"
5
"Sterg multiplii numarului;"
5
"Au trecut prin sita."
"trec la numarul urmator"
7
"Sterg multiplii numarului;"
7
"Au trecut prin sita."
"trec la numarul urmator"
11
"Sterg multiplii numarului;"
11
"Au trecut prin sita."
"trec la numarul urmator"
13
"Sterg multiplii numarului;"
13
"Au trecut prin sita."
"trec la numarul urmator"
17
"Sterg multiplii numarului;"
17
"Au trecut prin sita."
"trec la numarul urmator"
19
"Sterg multiplii numarului;"
19
"Au trecut prin sita."
"trec la numarul urmator"

Programul a rulat !
Modular Language written by Dan V Popa, Ro/Haskell Group.
http://www.haskell.org/haskellwiki/Rodin
21/aug/2009  -           - Codename:ExperimentExp11
```

8. Conclusions

This article is dedicated to The Rodin Community, a community of teachers dedicated to make C-like languages affordable by Romanian Students. The article focuses on the latest stage of development of the Rodin Language, which had been revised during this summer of 2009. The Rodin version of the moment (after one year from its first release in 2008) is a bit different – being modularly sliced and verified module by module – and then rebuild using various operating systems.

Nowadays, Rodin ExperimentExp11, the latest version of the moment is available for various platforms: Windows, Linux- 386, Linux-X86-64.

The main theoretic aspects of Rodin as those presented on Anglo Haskell 2008 web page from [3] by myself will probably be the subject of an other paper or on other book like [1].

References

[1] Dan Popa, **Practica Interpretării Monadice**, MatrixRom, Bucureşti, 2008, ISBN 978-973-755-417-8
[2] Dan Popa, **Introducere în Haskell 98 prin exemple**, Edusoft, Bacău, 2007, ISBN 978-973-8934-48-1
[3] The Haskell Community – www.haskell.org
[4] The Ro/Haskell Community – www.haskell.org/haskellwiki/Ro/Haskell
[5] The Rodin Community – www.haskell.org/haskellwiki/Rodin
[6] Pidgin – http://www.pidgin.im

Dan POPA
Department of Mathematics and Informatics
Faculty of Sciences
"Vasile Alecsandri" University of Bacău, Romania,
Spiru Haret 8, 600114 Bacău, Romania
danvpopa@ub.ro