

VeHa: A Hybrid National Verification Hackathon for Better Formal Methods Education

Sergey Staroletov^{1,2}, Dmitry Kondratyev³, Vladimir Shelekhov^{3,4}, **Alexander Kogtenkov**⁵,
Nikolay V. Shilov⁶, Natalia Garanina³, Irina Shoshmina⁷, Timofey Cherganov⁸, Vasil Dyadov⁵
serg_soft@mail.ru kwaxer@mail.ru

¹Institute of Automation and Electrometry, Novosibirsk, Russia

²Polzunov Altai State Technical University, Barnaul, Russia

³A.P. Ershov Institute of Informatics Systems, Novosibirsk, Russia

⁴Novosibirsk State University, Novosibirsk, Russia

⁵Kaspersky Lab, Moscow, Russia

⁶Lyceum 22 "Hope of Siberia", Novosibirsk, Russia

⁷Peter the Great St. Petersburg Polytechnic University, St. Petersburg, Russia

⁸RusBITech-Astra LLC, Moscow, Russia

(VeHa Consortium)

FMTea Workshop – May 19, 2026 – Tokyo, Japan

The Gap in Formal Methods Education

- Formal verification is critical for reliable software, but teaching it is hard.
- High entry barrier: students struggle to move from theory to practice.
- Traditional courses (testing, electives) are insufficient for the broader community.
- Need for applied, project-based activities that motivate deep learning.

Solution: A hackathon-style competition – **VeHa** (V**E**rification H**A**ckathon).

Why a Hackathon?

- Hackathons are effective for intensive, team-based problem solving.
- Verification is “programming” in a DSL (specification + proof).
- Competition + tight deadlines + mentoring + training materials = motivating environment.
- Hybrid (in-person + online) format lowers geographic barriers.

“VeHa” also means “a milestone” in Russian (and in Common Slavic).

Evolution of the VeHa Series

Year	Location	Focus	Key outcome
2023	Innopolis (PSSV)	SPIN/Promela, C-lightVer	15 teams; model checking difficult
2024	Innopolis (PSSV)	Frama-C, Coq, TLA+, C-lightVer	More industrial tasks, mixed success
2025	Novosibirsk (Pottosin Contest)	6 tasks, academy–industry partnership	Strong participation, measurable growth

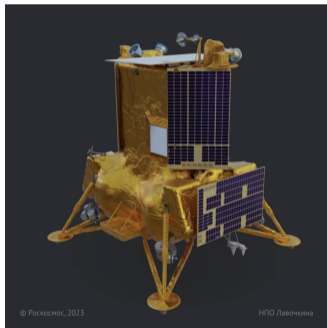
Industry partners: Kaspersky Lab, Astra Group (Astra Linux).

- Following the awards ceremony for the first hackathon, Astra Group employees who participated in it proposed collaboration for the following year.
- After participating in the second hackathon, Kaspersky Lab employees also proposed collaboration for the following year.

- **Location:** Innopolis (satellite event of PSSV workshop)
- **Format:** Hybrid (in-person + online)
- **Tracks:**
 - Deductive verification – C-lightVer (C-light language¹, uses ACL2 as a backend)
 - Model checking – SPIN / Promela
- **Tasks:**
 - Simple: deductive verification of array comparison function
 - Complex: modeling of Luna-25 space station accident (real-world incident)
 - Industrial protocols
- **Participation:** 15 teams, 13 submitted deductive solutions
- **Challenges identified:**
 - Model checking was too difficult – only one team solved Luna-25 task
 - Problems with LTL property formulation, concurrent modeling, counterexample analysis
- **Feedback:** Overall satisfaction but need for clearer problem statements

¹Kondratyev, D.A., Nepomniaschy, V.A.: Automation of C program deductive verification without using loop invariants. *Program. Comput. Softw.* 48(5), 331–346 (2022).

Model checking track: simulate the behavior of the space station control system leading to an accident based on official press releases.



The Roscosmos corporation published preliminary investigation results. They state that Luna-25 crashed into Earth's natural satellite due to a malfunction in the BIUS-L angular velocity measurement unit. It turns out that too many commands were received by the instrument, and the BIUS-L was unable to select the highest-priority ones. "Consequently, the onboard control system received zero signals from the BIUS-L accelerometers," Roscosmos explained. The lack of accelerometer data "prevented the corrective impulse from detecting the moment the required velocity was reached and from shutting down the spacecraft's propulsion system in a timely manner." Luna-25's engine burned for 127 seconds instead of the planned 84, and the station entered an unplanned, open orbit and collided with the lunar surface.

Source: <https://www.ixbt.com/news/2023/10/03/roskosmos-nazval-osnovnuju-prichinu-krushenija-luny25-k-avarii-privel-sboj-v-bloke-biusl.html>

- **Location:** Innopolis (again with PSSV workshop)
- **New tools introduced:** Frama-C, Coq, TLA+
- **Tasks (five total):**
 - Verification of access rights functions (Frama-C, Coq)
 - GPU computational pipeline – model checking with optimal parameter search
 - 2-SAT deductive verification (C-lightVer)
 - IBFT consensus protocol – modeling and verification
- **Outcomes:**
 - Increased registrations vs. 2023
 - More practical, industry-oriented problems
 - However: no submissions for the consensus protocol task
- **Winners:** 16 winners/laureates across categories
- **Feedback highlights:**
 - Submission system needs improvement (public GitHub PRs led to visibility of solutions)
 - Strong interest in industrial applications of formal methods

Related Work – Verification Competitions

- **VerifyThis** (since 2011). Focus on solving complex software verification problems, depth over speed.
- **RERS** series. Initially tool performance comparison, later synthetic benchmarks for education.
- **TOOLympics** (2019, 2023). Merger of VerifyThis, RERS, and others – closest to VeHa, but not annual.
- **SpecifyThis**. Bridge between specification paradigms, case-based learning.

Main difference: VeHa combines **model checking** (SPIN, TLA+) and **deductive verification** (Isabelle, Coq, Why3, C-lightVer) in a single annual hybrid hackathon.

- ICPC² (popular in Russia): 5 hours, 8–12 problems, tested against hidden test suites.
- VeHa 2025: 3 days, six problems, verification via formal proofs, not black-box testing.
- We think that Integrating formal verification into popular programming contests (ICPC, Codeforces) is a pressing need to improve judging quality and educational level.

²<https://icpc.global>

Core principles:

- Learning-by-doing
- Teamwork + mentorship
- Industry–academia bridge
- Two-track hybrid: **model checking** & **deductive verification**
- Online/offline hybrid.

<https://sites.google.com/view/veha2025>

Time: from 05.11.2024 to 08.11.2025

III VeHa Contest

(Nice (Novosibirsk) / Online)

The event is organized by a group of enthusiasts for formal methods (SI SR BAAL, IAC 06 and SIAC 06, Informatics from University, and ITI) in cooperation with IMIS, SCS and Security Lab.

Analytical information about the competition - in this article:
Competitions on formal verification of Veri 2024: experience and prospects accumulated over two years (article about last year's competition)
Competitions for formal verification of Veri 2021: experience (article about the first competition)

Models verification and deductive verification competitions with tutorials in the Facebook forum.

Our researchers will present interesting tasks for inquisitive students/developers to solve. Formal methods, logic and formal languages will be useful in the solution!

Fill out the feedback form

Awarding record at GFRAS Open

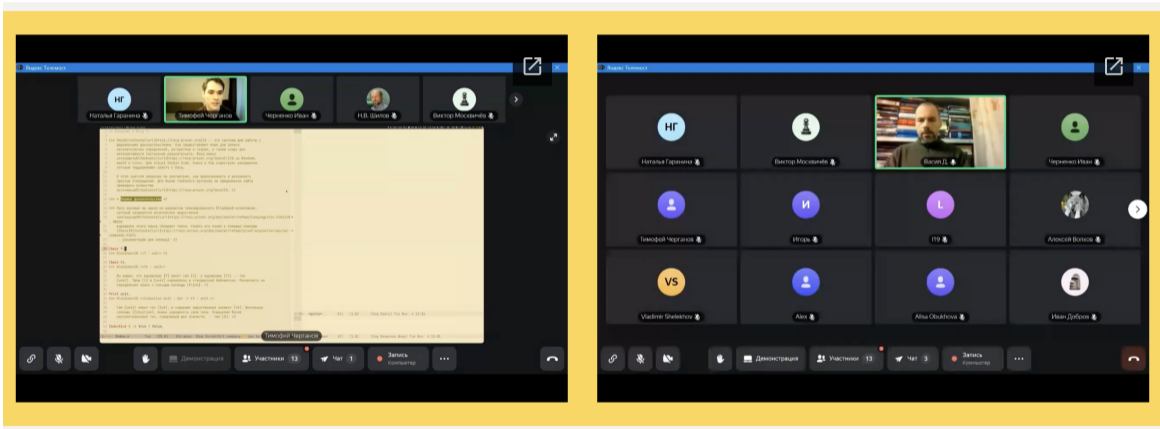
CONFERENCER

Preparatory Phase

- Public website with mini-manuals, video tutorials for each tool.
- Online tutorials (Yandex.Telemost):
 - C-lightVer (deductive verification)
 - TLA+ (model checking of reactive/distributed systems)
 - Rocq (Coq) – from inductive types to program verification.
- Invited lecture by **Alexei Lisitsa** (Univ. Liverpool): “Finite Countermodel Finding for Infinite State and Parametrized Verification”.



Tutoroals



Topic: Formal verification of distributed protocols (network consensus).

Pedagogical techniques:

- **Conceptual layering.** Start with state variables, then transition predicates, finally temporal operators (always, eventually).
- **Interactive experimentation.** The presenter along with participants modify specs and run TLC model checker live.
- **Counterexample analysis.** Show how to read error traces, localise bugs, and strengthen invariants.
- **Abstraction via nondeterminism.** Replace concrete values with sets – reduce state space.
- **Typing through invariants:** Use invariants to enforce type constraints (no separate type system).

Topic: Interactive theorem proving for functional correctness.

Pedagogical sequence (gradual complexity):

- 1 **Basic definitions.** Inductive types (bool, nat), pattern-matching functions.
- 2 **Online feedback.** Check and Compute commands – learn by doing.
- 3 **Tactics introduction:** `simpl`, `rewrite`, `induction`, `apply` – prove simple properties.
- 4 **Abstraction.** Type classes and instances (monoids, polymorphic functions) – link to official docs and Software Foundations.
- 5 **Move to the task.** Define imperative language syntax, operational semantics, state; prove Hoare-like assertions.

Model Checking	Deductive Verification
SPIN / Promela	Isabelle/HOL
TLA+	Coq (Rocq)
	Why3
	C-lightVer (ACL2 based)

Participants could choose one or more problems; solutions submitted via an ICPC-like testing system (NSU-TS). Manual checking by the juri.

- 1 **Model checking: Wenzhou train collision (CTCS)** – Promela, LTL.
- 2 **Distributed version control** – TLA+ specification.
- 3 **Functional lists in Isabelle/HOL** – folds, prefix operators.
- 4 **Static analysis in Rocq/Coq** – abstract interpretation.
- 5 **Chakravala algorithm (Pell's equation)** – C-lightVer, ACL2 invariants.
- 6 **Longest decimal substring** – deductive verification (Why3, Coq, Lean).

Task 1: Wenzhou Train Collision (Model Checking)

- **Tool:** SPIN / Promela
- **Type:** Research
- **Goal:** Model the CTCS high-speed train control system crash scenario.
- **Requirements:**
 - Formalize equipment fault tolerance, automatic blocking, dispatch-onboard interactions, time constraints.
 - Formulate LTL safety properties whose violation leads to collision.
 - Propose architectural changes to guarantee safety under multiple failures.
- **Reading:** Ouyang et al. (STAMP-based analysis)
- **Outcome:** Only one team submitted a partial solution (special diploma) + one good solution came after the contest.

Task 1: Wenzhou Train Collision (Model Checking)



- Just like in the first content, the task was to simulate an accident.
- This time, it was about Chinese high-speed trains, as there are plenty of natural language descriptions online (English and Chinese Wikipedia entries on collisions, train control system components, and automatic blocking).
- The goal was also to simulate the control system, show the state of the crash and propose a fix, verifiable by a verifier.
- However, many teams postponed the task due to more interest in industrial challenges and more clearly defined problems, and ultimately did not submit solutions.

Task 2: Distributed Version Control (Industrial, Kaspersky)

- **Tool:** TLA+
- **Type:** Industrial (proposed by Vasil Dyadov, Ilya Shchepetkov)
- **Goal:** Specify a distributed version control system with multiple agents editing files in parallel, synchronizing via a central server, and handling conflicts (Push, Pull, Merge).
- **Main aspects:**
 - Linear change log
 - Eventual consistency under diverging histories
 - Stepwise verification of invariants and absence of data loss using TLC model checker

Task 3: Functional Lists in Isabelle/HOL (Industrial, Kaspersky)

- **Tool:** Isabelle/HOL
- **Type:** Industrial (proposed by Alexander Kogtenkov)
- **Goal:** Prove properties of functional programs on recursive data structures.
- **Tasks:**
 - Prove lemmas about list folds
 - Equivalence of different prefix operator implementations
 - Properties of functions on lists of lists
- **Methodological constraint:** Use only basic tactics (`simp`, `blast`, `cases`, `induction`).

Task 4: Static Analysis in Rocq/Coq (Industrial, Astra Group)

- **Tool:** Rocq (Coq)
- **Type:** Industrial (proposed by Timofey Cherganov, Ivan Smirnov)
- **Goal:** Build and prove correctness of an abstract interpreter for a simple imperative language.
- **Requirements:**
 - Define abstract domains (flat constant lattice, interval analysis)
 - Formalize lattice structure, prove monotonicity and correctness of abstract operations
 - Extend analysis with branch and loop conditions
 - Prove preservation of safety properties (e.g., no division by zero)
- **Feedback:** Received three “5.0” ratings from participants.

Task 5: Chakravala Algorithm (Research)

- **Tool:** C-lightVer + ACL2
- **Type:** Research (proposed by Dmitry Kondratyev)
- **Goal:** Deductive verification of the historical Chakravala method for Pell's equation $x^2 - ny^2 = 1$.
- **Challenge:** Identify a loop invariant reflecting the arithmetic essence, using recurrent relations from Ayyangar's paper.
- **Automation:** ACL2 automatically applies domain theory lemmas to prove verification conditions.

Task 5: Chakravala Algorithm (Research)

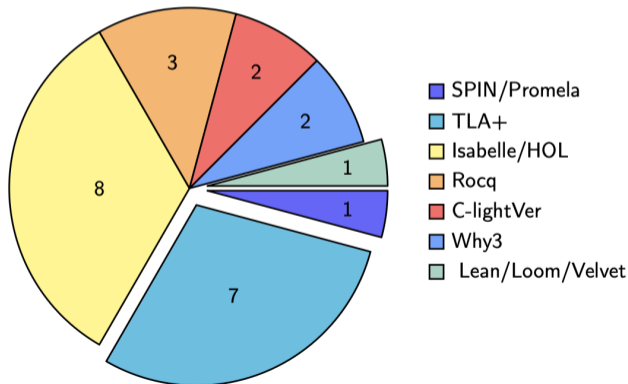
```
1 /*
2 (and
3 (integer n)
4 (= 0 n)
5 )
6 */
7 void chakravala(int n, int variables[])
8 {
9     int a = 1;
10    int b = 0;
11    int m = 0;
12    int k = 1;
13    int i = 0;
14    int j = 0;
15    int c = 1;
16    int d = 0;
17    int e = 0;
18    int f = 0;
19    int g = 0;
20    int h = 0;
21    int l = 0;
22    int o = 0;
23    int p = 0;
24    int q = 0;
25    int r = 0;
26    int s = 0;
27    int t = 0;
28    int u = 0;
29    int v = 0;
30    int w = 0;
31    int x = 0;
32    int y = 0;
33    int z = 0;
34
35    /*
36
37
38    */
39    while (k != 1 || f != 0)
40    {
41        if (k != 0 && c != 0)
42        {
43            f = n + 1;
44            a = r / k;
45            y = k * a;
46            q = f - y;
47
48            if (q == 0)
49            {
50                o = k*i;
51                e = o-m;
52                if (o != 0)
53                {
54                    if (o > 0)
55                    {
56                        s = 0;
57                        if (q == 1)
58                        {
59                            t = pep;
60                            l = n - 1;
61                            if (l < 1)
62                            {
63                                n = 1;
64                            }
65                            else
66                            {
67                                m = p;
68                            }
69                        }
70                    }
71                }
72            }
73            else
74            {
75                k = k - n;
76                g = 0;
77                u = -1;
78                f = 1;
79                c = 1;
80                k = e + m;
81                k = k - n;
82            }
83            else
84            {
85                if (f == 1)
86                {
87                    j = a;
88                    g = 0;
89                    h = a;
90                    u = v;
91                    v = b;
92                    w = m + j;
93                    c = k;
94                    z = w / c;
95                    a = h*w;
96                    b = w*w;
97                    g = a - 0;
98                    d = b - 0;
99                    k = z*w;
100                   d = 2*w;
101                   d = d*w;
102                   k = k - d;
103                   t = j*a;
104                   t = t - n;
105                   t = t / c;
106                   k = k + t;
107                }
108                else
109                {
110                    m = 1;
111                    b = 0;
112                    c = 1;
113                    f = 1;
114                }
115            }
116        }
117        else
118        {
119            a = 0;
120            i = 0;
121        }
122        else
123        {
124            o = 1;
125            p = 1;
126            l = i + 1;
127        }
128        else
129        {
130            m = 1;
131            b = 0;
132            c = 0;
133            k = 1;
134        }
135    }
136    else
137    {
138        i = i + 1;
139    }
140    }
141    variables[0] = a;
142    variables[1] = b;
143    /*
144    (
145    (
146    (
147    (
148    (
149    (
150    (
151    (
152    (
153    (
154    (
155    (
156    (
157    (
158    (
159    (
160    (
161    (
162    (
163    (
164    (
165    (
166    (
167    (
168    (
169    (
170    (
171    (
172    (
173    (
174    (
175    (
176    (
177    (
178    (
179    (
180    (
181    (
182    (
183    (
184    (
185    (
186    (
187    (
188    (
189    (
190    (
191    (
192    (
193    (
194    (
195    (
196    (
197    (
198    (
199    (
200    (
201    (
202    (
203    (
204    (
205    (
206    (
207    (
208    (
209    (
210    (
211    (
212    (
213    (
214    (
215    (
216    (
217    (
218    (
219    (
220    (
221    (
222    (
223    (
224    (
225    (
226    (
227    (
228    (
229    (
230    (
231    (
232    (
233    (
234    (
235    (
236    (
237    (
238    (
239    (
240    (
241    (
242    (
243    (
244    (
245    (
246    (
247    (
248    (
249    (
250    (
251    (
252    (
253    (
254    (
255    (
256    (
257    (
258    (
259    (
260    (
261    (
262    (
263    (
264    (
265    (
266    (
267    (
268    (
269    (
270    (
271    (
272    (
273    (
274    (
275    (
276    (
277    (
278    (
279    (
280    (
281    (
282    (
283    (
284    (
285    (
286    (
287    (
288    (
289    (
290    (
291    (
292    (
293    (
294    (
295    (
296    (
297    (
298    (
299    (
300    (
301    (
302    (
303    (
304    (
305    (
306    (
307    (
308    (
309    (
310    (
311    (
312    (
313    (
314    (
315    (
316    (
317    (
318    (
319    (
320    (
321    (
322    (
323    (
324    (
325    (
326    (
327    (
328    (
329    (
330    (
331    (
332    (
333    (
334    (
335    (
336    (
337    (
338    (
339    (
340    (
341    (
342    (
343    (
344    (
345    (
346    (
347    (
348    (
349    (
350    (
351    (
352    (
353    (
354    (
355    (
356    (
357    (
358    (
359    (
360    (
361    (
362    (
363    (
364    (
365    (
366    (
367    (
368    (
369    (
370    (
371    (
372    (
373    (
374    (
375    (
376    (
377    (
378    (
379    (
380    (
381    (
382    (
383    (
384    (
385    (
386    (
387    (
388    (
389    (
390    (
391    (
392    (
393    (
394    (
395    (
396    (
397    (
398    (
399    (
400    (
401    (
402    (
403    (
404    (
405    (
406    (
407    (
408    (
409    (
410    (
411    (
412    (
413    (
414    (
415    (
416    (
417    (
418    (
419    (
420    (
421    (
422    (
423    (
424    (
425    (
426    (
427    (
428    (
429    (
430    (
431    (
432    (
433    (
434    (
435    (
436    (
437    (
438    (
439    (
440    (
441    (
442    (
443    (
444    (
445    (
446    (
447    (
448    (
449    (
450    (
451    (
452    (
453    (
454    (
455    (
456    (
457    (
458    (
459    (
460    (
461    (
462    (
463    (
464    (
465    (
466    (
467    (
468    (
469    (
470    (
471    (
472    (
473    (
474    (
475    (
476    (
477    (
478    (
479    (
480    (
481    (
482    (
483    (
484    (
485    (
486    (
487    (
488    (
489    (
490    (
491    (
492    (
493    (
494    (
495    (
496    (
497    (
498    (
499    (
500    (
501    (
502    (
503    (
504    (
505    (
506    (
507    (
508    (
509    (
510    (
511    (
512    (
513    (
514    (
515    (
516    (
517    (
518    (
519    (
520    (
521    (
522    (
523    (
524    (
525    (
526    (
527    (
528    (
529    (
530    (
531    (
532    (
533    (
534    (
535    (
536    (
537    (
538    (
539    (
540    (
541    (
542    (
543    (
544    (
545    (
546    (
547    (
548    (
549    (
550    (
551    (
552    (
553    (
554    (
555    (
556    (
557    (
558    (
559    (
560    (
561    (
562    (
563    (
564    (
565    (
566    (
567    (
568    (
569    (
570    (
571    (
572    (
573    (
574    (
575    (
576    (
577    (
578    (
579    (
580    (
581    (
582    (
583    (
584    (
585    (
586    (
587    (
588    (
589    (
590    (
591    (
592    (
593    (
594    (
595    (
596    (
597    (
598    (
599    (
600    (
601    (
602    (
603    (
604    (
605    (
606    (
607    (
608    (
609    (
610    (
611    (
612    (
613    (
614    (
615    (
616    (
617    (
618    (
619    (
620    (
621    (
622    (
623    (
624    (
625    (
626    (
627    (
628    (
629    (
630    (
631    (
632    (
633    (
634    (
635    (
636    (
637    (
638    (
639    (
640    (
641    (
642    (
643    (
644    (
645    (
646    (
647    (
648    (
649    (
650    (
651    (
652    (
653    (
654    (
655    (
656    (
657    (
658    (
659    (
660    (
661    (
662    (
663    (
664    (
665    (
666    (
667    (
668    (
669    (
670    (
671    (
672    (
673    (
674    (
675    (
676    (
677    (
678    (
679    (
680    (
681    (
682    (
683    (
684    (
685    (
686    (
687    (
688    (
689    (
690    (
691    (
692    (
693    (
694    (
695    (
696    (
697    (
698    (
699    (
700    (
701    (
702    (
703    (
704    (
705    (
706    (
707    (
708    (
709    (
710    (
711    (
712    (
713    (
714    (
715    (
716    (
717    (
718    (
719    (
720    (
721    (
722    (
723    (
724    (
725    (
726    (
727    (
728    (
729    (
730    (
731    (
732    (
733    (
734    (
735    (
736    (
737    (
738    (
739    (
740    (
741    (
742    (
743    (
744    (
745    (
746    (
747    (
748    (
749    (
750    (
751    (
752    (
753    (
754    (
755    (
756    (
757    (
758    (
759    (
760    (
761    (
762    (
763    (
764    (
765    (
766    (
767    (
768    (
769    (
770    (
771    (
772    (
773    (
774    (
775    (
776    (
777    (
778    (
779    (
780    (
781    (
782    (
783    (
784    (
785    (
786    (
787    (
788    (
789    (
790    (
791    (
792    (
793    (
794    (
795    (
796    (
797    (
798    (
799    (
800    (
801    (
802    (
803    (
804    (
805    (
806    (
807    (
808    (
809    (
810    (
811    (
812    (
813    (
814    (
815    (
816    (
817    (
818    (
819    (
820    (
821    (
822    (
823    (
824    (
825    (
826    (
827    (
828    (
829    (
830    (
831    (
832    (
833    (
834    (
835    (
836    (
837    (
838    (
839    (
840    (
841    (
842    (
843    (
844    (
845    (
846    (
847    (
848    (
849    (
850    (
851    (
852    (
853    (
854    (
855    (
856    (
857    (
858    (
859    (
860    (
861    (
862    (
863    (
864    (
865    (
866    (
867    (
868    (
869    (
870    (
871    (
872    (
873    (
874    (
875    (
876    (
877    (
878    (
879    (
880    (
881    (
882    (
883    (
884    (
885    (
886    (
887    (
888    (
889    (
890    (
891    (
892    (
893    (
894    (
895    (
896    (
897    (
898    (
899    (
900    (
901    (
902    (
903    (
904    (
905    (
906    (
907    (
908    (
909    (
910    (
911    (
912    (
913    (
914    (
915    (
916    (
917    (
918    (
919    (
920    (
921    (
922    (
923    (
924    (
925    (
926    (
927    (
928    (
929    (
930    (
931    (
932    (
933    (
934    (
935    (
936    (
937    (
938    (
939    (
940    (
941    (
942    (
943    (
944    (
945    (
946    (
947    (
948    (
949    (
950    (
951    (
952    (
953    (
954    (
955    (
956    (
957    (
958    (
959    (
960    (
961    (
962    (
963    (
964    (
965    (
966    (
967    (
968    (
969    (
970    (
971    (
972    (
973    (
974    (
975    (
976    (
977    (
978    (
979    (
980    (
981    (
982    (
983    (
984    (
985    (
986    (
987    (
988    (
989    (
990    (
991    (
992    (
993    (
994    (
995    (
996    (
997    (
998    (
999    (
1000    (
1001    (
1002    (
1003    (
1004    (
1005    (
1006    (
1007    (
1008    (
1009    (
1010    (
1011    (
1012    (
1013    (
1014    (
1015    (
1016    (
1017    (
1018    (
1019    (
1020    (
1021    (
1022    (
1023    (
1024    (
1025    (
1026    (
1027    (
1028    (
1029    (
1030    (
1031    (
1032    (
1033    (
1034    (
1035    (
1036    (
1037    (
1038    (
1039    (
1040    (
1041    (
1042    (
1043    (
1044    (
1045    (
1046    (
1047    (
1048    (
1049    (
1050    (
1051    (
1052    (
1053    (
1054    (
1055    (
1056    (
1057    (
1058    (
1059    (
1060    (
1061    (
1062    (
1063    (
1064    (
1065    (
1066    (
1067    (
1068    (
1069    (
1070    (
1071    (
1072    (
1073    (
1074    (
1075    (
1076    (
1077    (
1078    (
1079    (
1080    (
1081    (
1082    (
1083    (
1084    (
1085    (
1086    (
1087    (
1088    (
1089    (
1090    (
1091    (
1092    (
1093    (
1094    (
1095    (
1096    (
1097    (
1098    (
1099    (
1100    (
1101    (
1102    (
1103    (
1104    (
1105    (
1106    (
1107    (
1108    (
1109    (
1110    (
1111    (
1112    (
1113    (
1114    (
1115    (
1116    (
1117    (
1118    (
1119    (
1120    (
1121    (
1122    (
1123    (
1124    (
1125    (
1126    (
1127    (
1128    (
1129    (
1130    (
1131    (
1132    (
1133    (
1134    (
1135    (
1136    (
1137    (
1138    (
1139    (
1140    (
1141    (
1142    (
1143    (
1144    (
1145    (
1146    (
1147    (
1148    (
1149    (
1150    (
1151    (
1152    (
1153    (
1154    (
1155    (
1156    (
1157    (
1158    (
1159    (
1160    (
1161    (
1162    (
1163    (
1164    (
1165    (
1166    (
1167    (
1168    (
1169    (
1170    (
1171    (
1172    (
1173    (
1174    (
1175    (
1176    (
1177    (
1178    (
1179    (
1180    (
1181    (
1182    (
1183    (
1184    (
1185    (
1186    (
1187    (
1188    (
1189    (
1190    (
1191    (
1192    (
1193    (
1194    (
1195    (
1196    (
1197    (
1198    (
1199    (
1200    (
1201    (
1202    (
1203    (
1204    (
1205    (
1206    (
1207    (
1208    (
1209    (
1210    (
1211    (
1212    (
1213    (
1214    (
1215    (
1216    (
1217    (
1218    (
1219    (
1220    (
1221    (
1222    (
1223    (
1224    (
1225    (
1226    (
1227    (
1228    (
1229    (
1230    (
1231    (
1232    (
1233    (
1234    (
1235    (
1236    (
1237    (
1238    (
1239    (
1240    (
1241    (
1242    (
1243    (
1244    (
1245    (
1246    (
1247    (
1248    (
1249    (
1250    (
1251    (
1252    (
1253    (
1254    (
1255    (
1256    (
1257    (
1258    (
1259    (
1260    (
1261    (
1262    (
1263    (
1264    (
1265    (
1266    (
1267    (
1268    (
1269    (
1270    (
1271    (
1272    (
1273    (
1274    (
1275    (
1276    (
1277    (
1278    (
1279    (
1280    (
1281    (
1282    (
1283    (
1284    (
1285    (
1286    (
1287    (
1288    (
1289    (
1290    (
1291    (
1292    (
1293    (
1294    (
1295    (
1296    (
1297    (
1298    (
1299    (
1300    (
1301    (
1302    (
1303    (
1304    (
1305    (
1306    (
1307    (
1308    (
1309    (
1310    (
1311    (
1312    (
1313    (
1314    (
1315    (
1316    (
1317    (
1318    (
1319    (
1320    (
1321    (
1322    (
1323    (
1324    (
1325    (
1326    (
1327    (
1328    (
1329    (
1330    (
1331    (
1332    (
1333    (
1334    (
1335    (
1336    (
1337    (
1338    (
1339    (
1340    (
1341    (
1342    (
1343    (
1344    (
1345    (
1346    (
1347    (
1348    (
1349    (
1350    (
1351    (
1352    (
1353    (
1354    (
1355    (
1356    (
1357    (
1358    (
1359    (
1360    (
1361    (
1362    (
1363    (
1364    (
1365    (
1366    (
1367    (
1368    (
1369    (
1370    (
1371    (
1372    (
1373    (
1374    (
1375    (
1376    (
1377    (
1378    (
1379    (
1380    (
1381    (
1382    (
1383    (
1384    (
1385    (
1386    (
1387    (
1388    (
1389    (
1390    (
1391    (
1392    (
1393    (
1394    (
1395    (
1396    (
1397    (
1398    (
1399    (
1400    (
1401    (
1402    (
1403    (
1404    (
1405    (
1406    (
1407    (
1408    (
1409    (
1410    (
1411    (
1412    (
1413    (
1414    (
1415    (
1416    (
1417    (
1418    (
1419    (
1420    (
1421    (
1422    (
1423    (
1424    (
1425    (
1426    (
1427    (
1428    (
1429    (
1430    (
1431    (
1432    (
1433    (
1434    (
1435    (
1436    (
1437    (
1438    (
1439    (
1440    (
1441    (
1442    (
1443    (
1444    (
1445    (
1446    (
1447    (
1448    (
1449    (
1450    (
1451    (
1452    (
1453    (
1454    (
1455    (
1456    (
1457    (
1458    (
1459    (
1460    (
1461    (
1462    (
1463    (
1464    (
1465    (
1466    (
1467    (
1468    (
1469    (
1470    (
1471    (
1472    (
1473    (
1474    (
1475    (
1476    (
1477    (
1478    (
1479    (
1480    (
1481    (
1482    (
1483    (
1484    (
1485    (
1486    (
1487    (
1488    (
1489    (
1490    (
1491    (
1492    (
1493    (
1494    (
1495    (
1496    (
1497    (
1498    (
1499    (
1500    (
1501    (
1502    (
1503    (
1504    (
1505    (
1506    (
1507    (
1508    (
1509    (
1510    (
1511    (
1512    (
1513    (
1514    (
1515    (
1516    (
1517    (
1518    (
1519    (
1520    (
1521    (
1522    (
1523    (
1524    (
1525    (
1526    (
1527    (
1528    (
1529    (
1530    (
1531    (
1532    (
1533    (
1534    (
1535    (
1536    (
1537    (
1538    (
1539    (
1540    (
1541    (
1542    (
1543    (
1544    (
1545    (
1546    (
1547    (
1548    (
1549    (
1550    (
1551    (
1552    (
1553    (
1554    (
1555    (
1556    (
1557    (
1558    (
1559    (
1560    (
1561    (
1562    (
1563    (
1564    (
1565    (
1566    (
1567    (
1568    (
1569    (
1570    (
1571    (
1572    (
1573    (
1574    (
1575    (
1576    (
1577    (
1578    (
1579    (
1580    (
1581    (
1582    (
1583    (
1584    (
1585    (
1586    (
1587    (
1588    (
1589    (
1590    (
1591    (
1592    (
1593    (
1594    (
1595    (
1596    (
1597    (
1598    (
1599    (
1600    (
1601    (
1602    (
1603    (
1604    (
1605    (
1606    (
1607    (
1608    (
1609    (
1610    (
1611    (
1612    (
1613    (
1614    (
1615    (
1616    (
1617    (
1618    (
1619    (
1620    (
1621    (
1622    (
1623    (
1624    (
1625    (
1626    (
1627    (
1628    (
1629    (
1630    (
1631    (
1632    (
1633    (
1634    (
1635    (
1636    (
1637    (
1638    (
1639    (
1640    (
1641    (
1642    (
1643    (
1644    (
1645    (
1646    (
1647    (
1648    (
1649    (
1650    (
1651    (
1652    (
1653    (
1654    (
1655    (
1656    (
1657    (
1658    (
1659    (
1660    (
1661    (
1662    (
1663    (
1664    (
1665    (
1666    (
1667    (
1668    (
1669    (
1670    (
1671    (
1672    (
1673    (
1674    (
1675    (
1676    (
1677    (
1678    (
1679    (
1680    (
1681    (
1682    (
1683    (
1684    (
1685    (
1686    (
1687    (
1688    (
1689    (
1690    (
1691    (
1692    (
1693    (
1694    (
1695    (
1696    (
1697    (
1698    (
1699    (
1700    (
1701    (
1702    (
1703    (
1704    (
1705    (
1706    (
1707    (
1708    (
1709    (
1710    (
1711    (
1712    (
1713    (
1714    (
1715    (
1716    (
1717    (
1718    (
1719    (
1720    (
1721    (
1722    (
1723    (
1724    (
1725    (
1726    (
1727    (
1728    (
1729    (
1730    (
1731    (
1732    (
1733    (
1734    (
1735    (
1736    (
1737    (
1738    (
1739    (
1740    (
1741    (
1742    (
1743    (
1744    (
1745    (
1746    (
1747    (
1748    (
1749    (
1750    (
1751    (
1752    (
1753    (
1754    (
1755    (
1756    (
1757    (
1758    (
1759    (
1760    (
1761    (
1762    (
1763    (
1764    (
1765    (
1766    (
1767    (
1768    (
1769    (
1770    (
1771    (
1772    (
1773    (
1774    (
1775    (
1776    (
1777    (
1778    (
1779    (
1780    (
1781    (
1782    (
1783    (
1784    (
1785    (
1786    (
1787    (
1788    (
1789    (
1790    (
1791    (
1792    (
1793    (
1794    (
1795    (
1796    (
1797    (
1798    (
1799    (
1800    (
1801    (
1802    (
1803    (
1804    (
1805    (
1806    (
1807    (
1808    (
1809    (
1810    (
1811    (
1812    (
1813    (
1814    (
1815    (
1816    (
1817    (
1818    (
1819    (
1820    (
1821    (
1822    (
1823    (
1824    (
1825    (
1826    (
1827    (
1828    (
1829    (
1830    (
1831    (
1832    (
1833    (
1834    (
1835    (
1836    (
1837    (
1838    (
1839    (
1840    (
1841    (
1842    (
1843    (
1844    (
1845    (
1846    (
1847    (
1848    (
1849    (
1850    (
1851    (
1852    (
1853    (
1854    (
1855   
```

Task 6: Longest Decimal Substring (Research)

- **Tool:** Any (Why3, Coq, Lean, etc.)
- **Type:** Research (proposed by Vladimir Shelekhov)
- **Source:** 1980s Pascal programming textbook.
- **Goal:** Select longest substring of decimal digits that is closest to the left.
- **Requirements:**
 - Write efficient program
 - Build formal specification (pre/post, loop invariants)
 - Fully verify using chosen tool
 - Expert assessment of effectiveness (additional goal)
- **Observation:** List representation proved much harder than array representation; Why3 with arrays succeeded automatically.

Participation Metrics

- 35 teams registered (47 participants, 8 cities).
- 19 teams (30 participants) actively solved problems (4 cities).
- Solutions by tool (total 23 submitted solutions):



Competencies Acquired by the Participants

- **Train collision (SPIN)** – abstract domain modeling, LTL safety properties, fault injection.
- **Distributed VC (TLA+)** – compositional design, invariants, model checking with TLC.
- **Isabelle/HOL lists** – structural induction, equivalence proofs, modular reasoning.
- **Coq static analysis** – abstract interpretation lattice, concretization, monotonicity.
- **Chakravala** – loop invariants from recurrence relations, automated verification.
- **Longest substring** – formal specification, WhyML, automation vs. interactive proof.

Awards by city:

- St. Petersburg, Russia: 14 awards (SPbPU, ITMO)
- Novosibirsk, Russia: 7 awards (NSU, SB RAS)
- Moscow, Russia: 5 awards (BMSTU, MIPT, ISP RAS)
- International: University of Naples (Cyprus) – 1st in task 6.



<https://youtu.be/zMIQGuX1YzI>

Participant Feedback (responses from 12 teams)

- 67% gave overall organisation the highest rating (5/5).
- 100%: task level “okay”, wording “fairly adequate”.
- 42% wanted a longer event.
- Preferred format: 67% hybrid, 33% fully online.
- Improvement requests: segmented time blocks, better submission system (closed repo), advance evaluation criteria.
- Task 4 (Rocq static analysis) received three 5/5 ratings.

Kaspersky Lab

KasperskyOS (microkernel OS)

Company:

- Attract people
- Demonstrate used tools
- Show FM use in the company
- Reuse materials in internal educational programs

Kaspersky Lab KasperskyOS (microkernel OS)

Organizers:

- New tasks
- Tooling diversity
- Concise introduction
- Evaluation of results
- Prizes

Company:

- Attract people
- Demonstrate used tools
- Show FM use in the company
- Reuse materials in internal educational programs

Organizers:

- New tasks
- Tooling diversity
- Concise introduction
- Evaluation of results
- Prizes

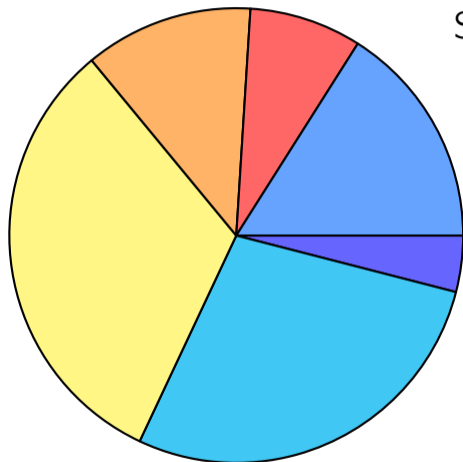


Company:

- Attract people
- Demonstrate used tools
- Show FM use in the company
- Reuse materials in internal educational programs

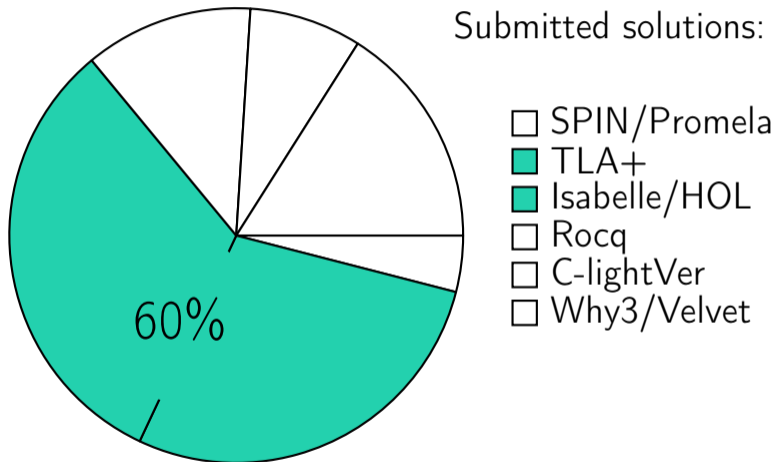
Goals	Methods	Tools
Strong functional correctness	Hoare logic	Frama-C + Why3 + Isabelle/HOL
Behavioral correctness	Model checking	TLA+/TLC/Pluscal
Security model guarantees	Refinement	Event-B
Language safety	Formal semantics, Run-time validation	Isabelle/HOL
...

Submitted solutions:



- SPIN/Promela
- TLA+
- Isabelle/HOL
- Rocq
- C-lightVer
- Why3/Velvet

Submitted solutions:



Factors:

- Detailed preparation procedure
- Self-contained introductory material
- Simplicity of tasks
- Gradual complexity of sub-tasks
- Transparent grades

Tasks characteristics

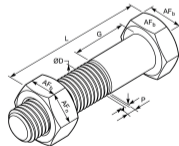
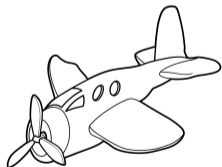
Characteristic	TLA+	Isabelle/HOL
Real	✗	✓✓
Interesting	✓	✗
Easy	✗✓	✓
Creative	✓	✗
Automation-resistant	✓	✓✓

Tasks characteristics

Characteristic	TLA+	Isabelle/HOL
Real	✗	✓✓
Interesting	✓	✗
Easy	✗✓	✓
Creative	✓	✗
Automation-resistant	✓	✓✓

Tasks characteristics

Characteristic	TLA+	Isabelle/HOL
Real	✗	✓✓
Interesting	✓	✗



Tasks characteristics

Characteristic	TLA+	Isabelle/HOL
Real	X	✓✓
Interesting		X
Easy		✓
Creative		X
Automation-resistant	✓	✓✓

Understanding

Tasks characteristics

Expectation:



Automation-resistant



Tasks characteristics

Reality:



Automation-resistant



Efficient navigation:

- Quick introduction to a new domain
- Core knowledge in **all** approaches
- Estimation: time, expertise, comp. resources
- Systematization

good



bad

Hybrid and heterogeneous proofs of correctness:

- Different methods in one project
- Integration of the tools
- Multi-level guarantees: tools, programs, data

- VeHa-2025 confirmed hackathon format as **effective for teaching formal methods**.
- **Combining model checking and deductive verification** in one event provides a comprehensive view.
- **Industry** involvement ensures relevance and **motivates** participants.
- The community is growing; **strong centers** in St. Petersburg, Novosibirsk, Moscow.

- **Balance** task difficulty – multi-level tasks work better.
- Improve submission **infrastructure** (closed repositories, better integration with website).
- Attract more industrial participants (**not only researchers**).
- Build a **benchmark** database from submitted solutions (for local verification tools).
- Strengthen **international** collaboration and increase outreach.

- Ivannikov Institute for System Programming (**ISP RAS**) (award ceremonies, support for publishing analytics papers).
- **V.A. Nepomniaschy**³ Memorial Fund (for support to go there).

Thank you!

<https://github.com/VeHaContest> – Collection of VeHa Solutions
Questions?

³<https://dblp.org/pid/n/VANepomniaschy.html>