iRunner: an educational platform of Belarusian State University

Sergei Sobol (sobols@bsu.by)

iRunner ('Insight Runner') is a multifunctional web platform that has been developed at the Faculty of Applied Mathematics and Computer Science of Belarusian State University. It is an online judge, a course and contest management system.

We use iRunner to teach programming and computer science. There are 340 academic problems along with 6.5K test cases that cover different topics of algorithm design and implementation ('Binary Search Trees', 'Dynamic Programming', etc.). Solutions are judged automatically; students get instant feedback.

iRunner can also handle programming competitions of different kinds. Western Subregional Collegiate Programming Contest (a stage of ICPC organized in Minsk) is annually held on iRunner. Moreover, the platform is used for training secondary school students and serving local Olympiads in informatics.

Our product has a rich history since 2004. Now it stores about 250K solutions and 100+ GB of data.

Let us name some key features of iRunner that distinguish it from similar packages.

Storing full output. When executing a solution, we put all its output (files, text written to *stdout*/*stderr*) to the storage. If the solution is not accepted, the outputs help a teacher to diagnose a bug and to give a piece of advice to the student. iRunner uses smart content-based hashing and does not save the same data multiple times.

Sandboxing. We try to run the solutions in safe environment. Main executors operate on Windows and rely on its built-in security features. Linux executor uses Docker containers.

Rejudges. When test cases are added or modified, it is possible to retest a set of affected solutions. iRunner stores a history of all runs for each solution. If something goes wrong during a rejudge (e.g. the new-added test is malformed), you can easily rollback your changes.

Statements in TeX. We use custom TeX-like language for writing problem statements. The engine renders math and highlights code snippets. All the problems in our database look uniform and neat when viewed as HTML.

Test case validation. It is possible to write validators for problems using *testlib.h* library. A validator is a program that checks input data correctness.

Challenges. This technique is useful when adding new test cases. Imagine you already have a set of accepted solutions for a problem. You submit new input data, and iRunner launches each of the solutions over this input. Then you can compare the outputs and find solutions that have failed the test.

Arbitrary problems. We are developing the ability to create custom problems using *pytest* library. Many real-life tasks do not fit well into the format of a traditional competitive programming contest problem: you may want to create multiple files or directories, to run programs with different command-line arguments, etc. iRunner lets you create highly customized problems.

Plagiarism detection. Students sometimes try to deceive teachers and submit somebody else's solutions. iRunner compares a new solution to the previous solutions stored in the database and calculates the similarity score. Best-matched solution pairs are then manually reviewed. This approach helps us to avoid cheating.

Quizzes. It is another way to check the student's knowledge. A quiz consists of several questions. Students are asked to choose the correct items from the given options or to type their own answers in. We invented a framework for creating quiz questions in bulk. For example, given an adjacency matrix, the student has to count the number of connected components in the graph. We create plenty of such questions using a special generator, so each student receives his own matrix and has no possibility to cheat.

Electronic queue. Classroom lessons are usually limited in time. Each student needs to tell his solution to a teacher and then get a confirmation that his algorithm is correct. iRunner establishes the order the students talk to the teacher during the class.