



Application of Olympiad-Style Code Assessment to Pre-Hire Screening of Programmers

Grzegorz Jakacki Marcin Kubica Tomasz Waleń

IOI 2011

Outline:

- History
- Tasks and Task Preparation
- Codility Web Service
- Evaluation Technology
- Correctness and Soundness
- Scoring
- Conclusions

Exoweb — the beginning:

- Exoweb's demand for new employees (2005)
- Crucial ability to write correct code
- Hires-to-candidates ratio $\approx 1 : 50$
- Average interview time of 100 min.
- 80 h of senior engineer's time per one hire
- Main cause of rejection — inability to write bug-free code

Solution:

- Screening — programming test
- Automated evaluation
Exobench — command-line evaluation system
- 90 % screening accuracy
- 12 h of senior engineer's time per one hire
- Post-screening hires-to-candidates ratio $\approx 1 : 5$
- Codility is born (2008)
- Reimplementation of pre-hire screening as a web service.

Task characteristics:

- Goal of IOI and other programming contests:
select top contestants and fairly distribute points among other contestants
- Goal of pre-hire screening:
eliminate candidates that should not be employed
- Binary outcome of pre-hire screening —
reject or send for interview
- Tasks are much more elementary (compared to IOI)

Task preparation:

- Over a hundred tasks (and growing)
- General task preparation schema is similar to IOI
- Simpler task formulation
- No narrative story
- Only batch tasks are supported

Task preparation:

- Procedural interface (instead of standard I/O)
- Testing procedures instead of predetermined test files
- Tests profile similar to IOI:
 - correctness tests of various sizes,
 - border-cases tests,
 - efficiency tests

Task Preparation for Multiple Programming Languages

Variety of programming languages:

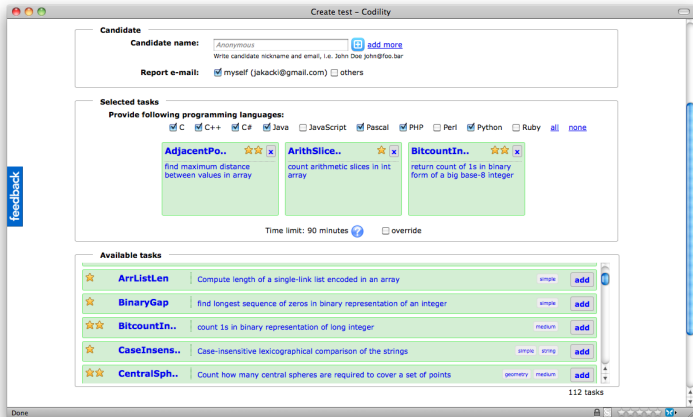
- C, C++, C#, Java, JavaScript, Pascal, Python, Perl, PHP, Ruby
(and still growing)
- Fully compilable, byte-code level compilable and scripting languages
- How to deal with such a variety of programming languages?

Task Preparation for Multiple Programming Languages

Solution:

- Python — front-end programming language
- Automated translation of solutions written in (a subset of) Python into other programming languages
- Evaluation based on translated solutions in Python
Language-specific solutions are also possible
- Automated time-limits calibration — independent of particular programming language and hardware

- Two types of users: candidates and recruiters
- Recruiter's interface:



- Candidate's interface — programmer's micro IDE:

The screenshot shows the Codility web IDE interface. At the top, the title bar says "Codility" and "version: 1.1.20". The top navigation bar includes language options: English, Polski, 中文, and a list of programming languages: Java, C++, C#, C, Javascript, Pascal, Perl, PHP, Python, Ruby. The user's session information is "nick: Demo ticket, passcode: demoEHMFKD-VCN".

The main content area is split into two panes. The left pane contains the problem description for "Equilibrium index of a sequence". It defines an equilibrium index as an index where the sum of elements at lower indexes equals the sum of elements at higher indexes. It provides an example sequence A with values: $A[0]=-7, A[1]=1, A[2]=5, A[3]=2, A[4]=-4, A[5]=3, A[6]=0$. It states that 3 is an equilibrium index because $A[0]+A[1]+A[2]=A[4]+A[5]+A[6]$, and 6 is also an equilibrium index because $A[0]+A[1]+A[2]+A[3]+A[4]+A[5]=0$. It notes that 7 is not an equilibrium index because it is not a valid index. A precise definition is given: the integer k is an equilibrium index if $A[0]+A[1]+\dots+A[k-1]=A[k]+A[k+1]+\dots+A[n-1]$ for $0 \leq k < n$. An "Example test" button is visible at the bottom of this pane.

The right pane is a code editor showing a Java snippet:

```
1 // you can also use imports, for example:
2 // import java.math.*;
3 int equi ( int[] A ) {http://codility.com/demo/run/
4     // write your code here
5 }
```

Below the code editor is a status bar showing "Position: Ln 5, Ch 2" and "Total: Ln 5, Ch 142". There are buttons for "help", "verify", "submit task", and "quit".

At the bottom of the right pane is a "Compiler output" window with a pink background, showing several error messages:

```
Compiler output:
wrapper.java:22: illegal start of expression
>
^
wrapper.java:26: illegal start of expression
public static void main(String[] args) {
^
wrapper.java:26: ';' expected
public static void main(String[] args) {
^
wrapper.java:26: illegal start of expression
public static void main(String[] args) {
^
wrapper.java:26: ';' expected
public static void main(String[] args) {
```

- Candidate's interface — testing facility:

Example test

WRONG ANSWER (got 7, which is not valid array index)

 **add test case**

write here your own test data

[remove](#)

[1,2,3]

NO RUNTIME ERRORS (returned value: 3)

write here your own test data

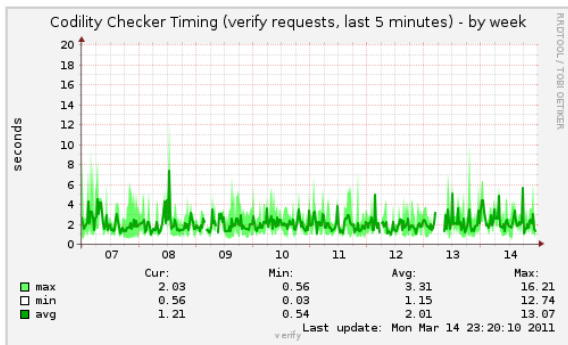
[remove](#)

[-7, 1, 5, 2, -4, 3, 0]

NO RUNTIME ERRORS (returned value: 7)

Evaluation Technology

- Generally, based on IOI and ACM CPC models
- Experimental asymptotic time complexity assessment
- Reactive tests augmentation — hybrid of:
model checking, symbolic interpretation, random and enumerative test generation
- Scalability — real time evaluation
- Grading machines — Amazon EC2 computing cloud



Correctness and Soundness

- Goal: to distinguish good programmers (winners) and bad programmers (losers).
- What does it mean?
- **Correctness**: all winners are good programmers
all bad programmers are losers
- **Soundness**: all good programmers are winners
all losers are poor programmers
- Programming contests — focus on correctness,
occasional lack of soundness is acceptable
- Pre-hire screening — soundness $>$ correctness
Codility's correctness $>$ 90 %
- 50 candidates \rightarrow 5 winners \rightarrow 1 hire

Scoring:

- Similar to IOI model
- 1–6 problems to solve
- Total score is the sum of scores for the tasks

- Successful application of programming contest's technology to pre-hire screening
- Today Codility is a profitable business
- Codility's customers include:
Nokia, Siemens, Barnes & Noble
- Two innovative technologies:
 - automated time complexity assessment, and
 - reactive tests augmentation

Want to try it yourself?

Get certified at:

`http://codility.com/cert/start/`

Thank you for your attention!