

Processed IOI Syllabus Feedback

ISC

Overview

The ISC received feedback to various aspects of the Syllabus from the following sources:

- Paul Ashton (New Zealand)
- Thomas Barnet-Lamb, Richard Forster, Brian Dean (GBR/USA)
- Giorgio Casadei (Italy)
- Sergejs Kozlovics and Martins Opmanis (Latvia)
- Pavel Pankov (Kyrgyzstan)
- Margot Phillipps (New Zealand)
- Peter Taylor (Australia)
- Velin K. Tzanov (Bulgaria)
- Troy Vasiga (Canada)

The body of this document contains detailed responses of the ISC to each of these materials. These responses are mostly technical and they are primarily addressed to the person(s) that submitted the feedback.

Appendix A contains a discussion on the content of the Syllabus.

Appendix B contains a discussion on the status of the Syllabus, and on the process of its integration. These issues require the attention of the IC.

1 New Zealand: Margot Phillipps, Paul Ashton

Feedback summary

1. Both submitters stress the importance of having a Syllabus for countries that are new to the IOI or plan to join it.

Responses

1. 😊

2 Italy: Giorgio Casadei

Feedback summary

1. Broad support of the Syllabus.
2. Request to include specific notes on programming languages.
3. Request to add new languages (Java, Prolog).

Responses

1. 😊
2. We are not really sure what kind or depth of notes was intended. We require the contestants to be able to turn an idea of an algorithm into a working program. This implies a good grasp of all the basic programming concepts – conditions, cycles, functions, etc. Nothing else is required or expected, even though knowledge of standard libraries (esp. STL) may help.

However, note that some of the points raised by Kozlovics and Opmanis may be related to this request. We address these points in the response to their feedback.

3. The possible addition of new languages is the responsibility of the ITWG. There's been an ongoing research of how to include Java, but including new languages brings lots of technical difficulties. The ISC will support new languages only after it can guarantee a reasonably fair evaluation of the competition.

3 Canada: Troy Vasiga

Feedback summary

1. Clarification request: is circle really “not needed”?
2. Clarification request: understanding quantifiers.
3. Observation that pointers and references seem to be required.
4. Expression of support.

Responses

1. A similar concern was raised by Kozlovics and Opmanis, and it is addressed in the Appendix A.
2. We agree that understanding quantifiers is a tricky and hard-to-define concept. Understanding single quantifiers, and their English equivalents (“for all”, “for none”, “exists”, “for some”, etc.) is surely a reasonable request.

However, empirical evidence shows that even at the IOI level contestants may have problems to grasp definitions that require more than one quantifier type (or similarly, use both max and min).

Such definitions have been used in past problem statements, but to make them more approachable they were usually split into several steps. It may be helpful to include an explaining footnote in this sense.

3. As far as pointers and references are concerned, we believe that the corresponding footnote in the Syllabus correctly explains the situation. (“The inessential advantage of scalable memory efficiency is outweighed by the increased complexity in reasoning. Static memory implementations should suffice to solve IOI tasks.”)

Even tree-like structures can be implemented using statically allocated arrays only, without using pointer and reference data types. Therefore we believe that pointers and references are classified correctly.

4. 😊

4 Kyrgyzstan: Pavel Pankov

Feedback summary

1. Proposal not to exclude tasks that require arbitrary-precision integer arithmetics.
2. Note that even if a problem has a solution that uses standard integer data types only, contestants may discover a solution where intermediate results overflow.
3. Proposal to change “Graphics and Visual Computing” from Excluded to partially Included.

Responses

1. A similar concern was raised by Kozlovics and Opmanis, and it is addressed in the Appendix A.
2. We would like to stress that the current classification “Not needed” does in fact exactly cover the situation from the note above: In the case mentioned in the feedback arbitrary-precision integer arithmetics is not required to write a correct solution.
3. Concerning the request for Graphics and Visual Computing:

We agree that visualisation is important in order to promote the IOI to the general public. However, implementing the changes proposed by Pavel Pankov would lead to several negative issues:

We would be putting more demands on the competitors’ knowledge. The proposal does claim that all IOI contestants do know points, segments, rectangles, circles and colors. While we don’t argue this, we would like to note that graphics programming is one of the most platform-dependent areas of programming, and contestants with different backgrounds might have experience with completely different graphics systems. If we decided to have the contestants actually produce graphical output, we would be forcing them to become familiar with the actual graphics system used in the competition. This important issue is completely neglected in the submitted proposal.

Also, scoring the graphical output by hand is bound to make the scoring process longer, more subjective and more error prone. It is not

clear whether the benefits from the proposal would really outweigh these negatives.

In our opinion, there are better ways how to involve visualisation – for example applets that let the contestant simulate the task and solve it by hand, or automated visualisation of the programs’ computations. These ways don’t put additional strain on the contestants, on the contrary, they may even be helpful. And in addition we may expect that visualisation prepared by the organizers well before the competition will be much more visually appealing.

5 Bulgaria: Velin K. Tzanov

Feedback summary

1. Stressing the importance for having clear rules how the Syllabus is changed, and the importance of actually changing it to reflect the evolvments in Computer Science.
2. Note that the General Assembly might be biased when voting on Syllabus changes, and that this may lead to lowering the quality of the IOI competition.
3. Proposal to give the authority to update the Syllabus to the ISC.
4. Alternate proposal: allow the ISC to make temporary changes, GA vote required to make them permanent after two years.

Responses

1. Here we can only agree.
2. With all due respect to the GA, this is a valid concern, and we feel that it is important to keep it in mind.

Note that this concern only applies if the Syllabus is adopted as a more-or-less binding set of guidelines. In other words, this concern matters if the Syllabus can be used by the GA to reject tasks the host SC and ISC consider to be suitable.

The following two proposals by Velin Tzanov, and the proposal by Barnet-Lamb, Forster and Dean all aim to address this situation. Given the importance of this issue we will discuss the introduction of the Syllabus as a separate issue.

3. Discussed separately in Appendix B.
4. Discussed separately in Appendix B.

6 Latvia: Sergejs Kozlovics, Martins Opmanis

Feedback summary

1. Stressing the importance for having clear rules how the Syllabus is changed, and the importance of actually changing it to reflect the evolvments in Computer Science.
2. Note that for some areas mentioned in the Syllabus the contestants who use C++ need only to know how to use a library function (as opposed to actually implementing it).
3. Note that the Syllabus may contain sketches about topics for later studying at high school.
4. Suggestion that in each Syllabus category each topic should have its own identifier. This would make referencing the syllabus easier, and it can even be used e.g. to assign those identifiers to textbooks that cover the respective areas.
5. Difference between languages: High-precision arithmetics.
6. Difference between languages: Random generators.
7. Difference between languages: Sorting and binary search.
8. Difference between languages: Data structures.
9. Difference between languages: Input and output.
10. Suggestions how to mitigate some of the differences: Providing a helper library? Explaining some algorithm in the problem statement?
11. Recommended inclusions into the Syllabus.

Responses

1. Here we can only agree.

2. There is no doubt that the concern is true. However, we would like to note that in practice using library functions without having a clue about their implementation often leads to poor results. (Common examples: Concatenating a lot of C++ `strings`, inserting an element into the middle of a C++ `vector`.) Thus we think that the contestants who use C++ should understand the concepts hidden in the STL.

Moreover, the points of the Syllabus that correspond to the library functions should still remain valid. E.g., the knowledge of basic sorting algorithms is still required, as some problems may require a modification of an existing algorithm. (As an example, consider adapting MergeSort to count the number of inversions.)

3. This note goes beyond the current scope of the Syllabus. In the future, once we have enough experience with using the current Syllabus at the IOI, this seems to be a good extension. However, to keep the Syllabus as concise as possible, it may be preferable to publish such materials separately.
4. The suggested way of labeling (e.g. “SORTING- $O(N^2)$ ”, “CONVEX-HULL”, “BFS”) seems too random to be practically usable. As for referencing the contents of the Syllabus, a simple yet helpful change would be to use ordered lists within categories. E.g., “Brute-force algorithms” would then be referenced as “AL2, item 2”.
5. This issue is addressed in the Appendix A.
6. We don’t really feel that there is any difference here. Generating a random integer in some range is documented e.g. in the `srand` manual page. Initializing the random generator in C/C++ can be done in a simpler way, using `srand(time(NULL))`;

The concern about waiting one second between test cases is invalid. The programs are tested on different test cases, and thus it doesn’t matter whether the generated random numbers seed is equal. Also note that model solutions are (usually) deterministic, and that for a correct randomized solution (i.e., one where the possibility of an error is sufficiently small) the choice of the random seed should not matter.

7. Remark taken into account, no Syllabus changes required.
8. Remark taken into account, no Syllabus changes required.

9. There were problems that required large input/output files in the past, and the established practice is to advise the contestants on how to process these files. This approach works reasonably well and it seems to be sufficient.
10. These suggestions are really valuable. Still, the ISC will need time to review whether it is feasible to implement them, and if yes, how.
11. The response on concrete inclusions is given in the Appendix A.

7 Australia: Peter Taylor

Feedback summary

1. Expressing concern that having a Syllabus might cause unwanted consequences. Citations:
“I find that exams in a syllabus driven system tend not to select the students who will be able to cope with an unforeseen situation, as they must in real life.”
“The reason I am attracted to Olympiads is that one is free to set problems in a style which might not be possible when a syllabus might prevent it.”

Responses

1. The general notion in the IOI community is that the IOI Syllabus should be accepted. We all want the IOI to be a leading-edge competition that mirrors the advancements in Computer Science, and the Syllabus is introduced as a means to this end, to help the countries prepare their contestants to the required level, thus raising the overall level of the competition. The existence of the Syllabus should in no way prevent a gradual introduction of innovations.
The ISC appreciates these words of advice, and it will take every step possible to prevent the Syllabus from harming the IOI in any way.

8 Great Britain / USA: Thomas Barnet-Lamb, Richard Forster, Brian Dean

Feedback summary

1. Suggestion to adopt the Syllabus gradually, first introducing only a broad overview of the syllabus, and gradually introducing a greater level of detail, and taking into account lessons from its use.
2. Citation: “Moreover, if things were done this way the original syllabus should be short and simple enough and each subsequent year’s changes to it should be small enough to be readily comprehended and carefully considered by the GA.”
3. Expressing concern that the Syllabus will be interpreted too rigidly, thus causing good questions to be rejected.
4. Suggestion that the Syllabus should not be the *source* of the authority on what subjects are suitable for the IOI. Instead, the precedent of the prior IOIs should have this authority, and the Syllabus should just be viewed as a formulation of these precedents, and of the past experience. This would give IOI the guarantee of a possibility to go beyond the current Syllabus.

Responses

The entire feedback by Barnet-Lamb, Forster and Dean addressed the process of introducing the Syllabus. We address this issue in Appendix B.

Appendix A: Concrete proposals on additions and changes

Understanding quantifiers

The corresponding item in DS2 could get a footnote stating that in case a definition contains more than one quantifier it is recommended to split it into parts to increase readability.

Heap data structure

Its status is clearly “Included, not for task description” due to HeapSort having the same status. An explicit mention of the (binary) heap data structure doesn’t seem necessary. The more advanced heap types (binomial, Fibonacci) are beyond the current scope of the IOI.

Array manipulation

Should be included in AL3 as “Simple array manipulation (filling, shifting/rotating, reversal, resizing, prefix sums).”

Interval tree data structure

This data structure has been used in past IOI tasks, and it is simple to implement using static arrays only. It may be included in AL3 as “Interval trees”.

Hash tables

The task mentioned by Kozlovics and Opmanis (IOI2001 Double Crypt) was an open-data task solvable without using hashing. Thus for this problem the current classification “Not needed” was correct, and we propose to keep it this way for now.

Induction/scanning

This request aims to include a specific mention of algorithms that sequentially process the input, while maintaining some invariant.

To incorporate this, the third point in AL3 may be changed to “Sequential processing, sequential and binary search algorithms.”

Scaling

This request aims to include a specific mention of algorithms that use a binary-search like approach to determine the optimal solution.

In our understanding, this is already covered by the inclusion of binary search as such. These algorithms just perform binary search on a conceptual array containing zeroes (no such solution exists) and ones (a solution does exist).

Heuristics

Currently, heuristics are classified as “Excluded”. However, the Syllabus provides no rationale for this. On the other hand, the request didn’t provide any rationale against the current classification.

We believe that the term Heuristics is too broad to be included. There have been past IOI problems that required the contestants to find heuristics, but always in the “approximating algorithms” sense – find a *valid* solution that is as good as possible.

Adding “Discrete approximation algorithms” to AL2 would cover these tasks and probably satisfy the request.

Extended Euclid’s algorithm and Chinese remainder theorem

We feel that inverse elements in finite fields are beyond the current scope of the IOI. Extended Euclid’s algorithm may be used on integers, and we consider it to be covered by the current Syllabus.

Radix conversion, Roman numerals, Factorization

These items should be added among the examples in the first point of AL3.

Floating-point arithmetics

This issue is much too difficult and it needs to be considered in more detail. We are aware of Martins Opmanis’s suggestions how to re-introduce floating point numbers into the competition. For some tasks this approach can indeed be used, but requiring full understanding of floating point arithmetics, precision errors and related issues seems to go beyond the current scope of the IOI.

Arbitrary precision integer arithmetics

The basic algorithms (e.g., addition and multiplication of non-negative integers) are simple enough and it makes sense to include them into the Syllabus.

On the other hand, e.g. the Fast Fourier Transform-based fast multiplication algorithm is clearly out of the current scope of the IOI.

The suggested change is to add “Simple operations on arbitrary precision integers (addition, subtraction, simple multiplication” into AL3.

Circle

Circle is currently marked as “Not needed” in 6.1.1. It should be brought to the same level as “Angle”, i.e., “Included, to be clarified”.

Angles, polar sorting, trigonometric functions

Kozlovics and Opmanis argue that both angles (also, sorting by polar angle) and trigonometric functions (marked as ”Excluded” in 6.1.1) could be included. They note that some simple properties of trigonometric functions are required in tasks concerning physics (e.g., refraction of light).

We believe that the current classification of angles is correct. Polar sorting is an important concept in Computational Geometry, and it can be done without resorting to trigonometric functions. Thus we suggest to add “Polar sorting” into AL10.

High school mathematics usually contains some basic properties of trigonometric functions, but it is not clear whether we want to add them into the Syllabus.

Squared papers and grids, Pick’s theorem

Pick’s theorem is a too narrow topic to be included in the Syllabus. We believe that grids are already implicitly covered both by 2D arrays and by the remark at AL10.

Sweeping line method

There were past IOI problems that used this method. Moreover, the general concept is simple enough. AL10 should include a point “Sweeping line method”.

Orthogonal rectangles

Feedback mentions tasks: finding their common area, common perimeter. Rectangles as such are included in 6.1.1, and both mentioned tasks are covered by the Sweeping line method.

The intersection of two convex polygons

Again, covered by the Sweeping line method.

Graph algorithms

The current state of graph algorithms in the Syllabus is inconsistent. Currently, they are mentioned as a part of AL3. However, given their importance it may be viable to make a separate AL category for these algorithms.

Also, the contents is inconsistent. While an algorithm to find an Euler tour is currently included, other similarly difficult algorithms are missing. Kozlovics and Opmanis suggest the following additions:

- Strongly-connected components
- Biconnected components, bridges and articulation points
- Network Flows
- Maximal matching in bipartite graphs
- Minimum-cost flow

In our opinion, minimum cost flow clearly exceeds the current scope of the IOI. The other algorithms are far too advanced to be specified as required knowledge but they may be used in one of the more difficult problems. The current way the Syllabus is formed doesn't really give a way to express this.

Strings, automata and grammars

It was suggested to transform AL7 into “String algorithms, automata and grammars”. These issues are indeed intimately related.

Kozlovics and Opmanis suggest the following items in this category:

- Anagrams (i.e., sorting the letters in each word)
- Knuth-Morris-Pratt string search algorithm (also the notion of prefix-function)

- Finite automata (is the word in the given language? how to construct an automaton accepting words containing “abracadabra”)
- Pattern matching
- Context-free grammars and arithmetic expression definition (also: prefix/infix/postfix notation and conversion between these notations)
- Properties of strings that arise from cyclically shifting the given string (see a backup task “Binary Codes” from IOI2001)
- Suffix trees
- Archiving (also, RLE-compression and Huffman codes). There was a task about RLE compression at BOI2006.

Our observations:

Pattern matching and KMP are currently covered in AL2.

Directed tree notations are currently covered under DS5, Traversal strategies.

Suffix trees and the Burrows-Wheeler transform are too advanced topics to be specified as prerequisite knowledge.

Gray codes

Too specific to be included as a category.

Integer partitions

Too specific to be included as a category. However, combinatorial algorithms are missing from the current Syllabus, and we need to find a suitable place for them.

Catalan’s numbers

If included, this would belong into DS4. It is not clear whether to include them.

Cryptographic algorithms

There were past olympiad problems related to cryptography, including one IOI tasks. However, these tasks were self-contained and required no previous cryptography knowledge, and currently we don’t want to require any such knowledge.

Normal forms

A proposal states that currently excluded “Normal forms” in DS2 may be useful when constructing a Boolean expression that corresponds to a given truth table.

While true, this at most advocates promoting “Normal forms” to be “Not needed”, and we don’t think that even this promotion is necessary at this moment.

Appendix B: Introducing the Syllabus

TODO