Foundations of Programming Languages received 3 comprehensive reviews in 2023. On the \( \beta \)-version of the KA. All reviews were handled in the same manner:

1. The reviewers comments were sent, along with the current evolving \( \chi \)-version of the KA, to all members of the subcommittee.
2. Subcommittee members were asked to independently consider each comment and the chair collated responses for consideration during a meeting.
3. Over the course of 2 meetings, all sections of the KA draft were revisited and discussed.
4. Discussion led to several bullet points in knowledge units being removed, combined, or moved to another KU to ensure that the KU’s were:
   a. Appropriate for undergraduate students
   b. At the appropriate level of exposure (CS-core, KA-core, or elective)
   c. Self-consistent
   d. Did not overlap excessively or unnecessarily.

**Feedback comment:**

Reviewers 1 and 4 suggested removing Logic Programming from the core.

**How incorporated:**

**Why not incorporated:**

No change was made. Logic Programming is a KA core topic, not a CS core topic. (This changed prior to the reviewer’s comments being made available). Hence, it is not necessarily covered by all programs and hence Liberal Arts schools wishing to shrink the core to a minimum are not impacted. The reviewers and subcommittee reached this decision independently.

**Date considered:** June 9, 2023

**Feedback comment:**

Reviewer 4 suggested scripting languages be removed from the CS Core due to overlap with OO languages.

**How incorporated:**

**Why not incorporated:**

Scripting languages was intended to cover unix shell scripts etc, and not interpreted languages which is how the reviewer appears to have interpreted it.

**Date considered:** June 9, 2023
Feedback comment:
Reviewer 4 suggested doubling the number of KA core hours for Program Representation since there was simply too much content.

How incorporated:
KA hours were increased to 4 hours, and some content move to non-core.

Why not incorporated:
Date considered: June 9, 2023

Feedback comment:
Reviewer 4 suggested Abstraction/Data Hiding/Modularity was missing from the OO KU.

How incorporated:
These concepts are already covered:
1. Object-oriented idioms for encapsulation
   a. Privacy, data hiding, and visibility of class members
   b. Interfaces revealing only method signatures
   c. Abstract base classes, traits and mixins

The terms used by the reviewer are simply not used, but the concepts are present.

Why not incorporated:
Date considered: June 9, 2023

Feedback comment:
Reviewer 4 was concerned that all the concepts in the OO KU could fit inside the core hours.

How incorporated:

Why not incorporated:
Students see programming and learn how to do it in other KA’s. This KU is simply discussing the generic concepts in a relatively language independent manner and is leveraging what students have learned elsewhere.

Date considered: June 9, 2023
Feedback comment:
Reviewer 4 noted that lambda expressions and evaluations is the first topic listed in the Functional programming KU. They expressed concern that this may scare some people away.

How incorporated:

Why not incorporated:
The topics are not presented in an order that corresponds to how the material is to be taught. Rather, it focusses on the key topics that students should be aware of. Since no Cs or KA core hours are devoted to any one topic, each program can shift the focus onto topics that make most sense for them and their students.

Date considered: June 9, 2023

Feedback comment:
Reviewer 4 noted that event-driven and reactive programming will have overlap with other KA’s

How incorporated:
This is noted in the KU where shared concepts are identified.

Why not incorporated:

Date considered: June 9, 2023

Feedback comment:
Reviewer 4 commented that there is too much content in the CS Core for Parallel and Distributed Computing.

How incorporated:

Why not incorporated:
No changes were made. The KU has significant overlap with the Parallel and Distributed Computing KA and hence this KU within Foundations of Programming Languages was intended to ensure all students gained an exposure to key concepts. Significantly more detail would come from the PDC KA.

Date considered: June 9, 2023

Feedback comment:
Reviewer 4 suggested more hours to Parallel and Distributed Computing KA., as well as Type Systems, and Language Translation and Execution.

How incorporated:
Why not incorporated:

While it is tempting to add more hours, we decided against it. Instructors have flexibility to go into more detail if they want, but students should minimally get an appreciation for the topic and its related challenges/issues/tools/concepts etc. The reviewer seems to think that significant detail is necessary.

Date considered: June 9, 2023

Feedback comment:
Reviewer 4 suggested some topics be moved around between CS core and KA core in the Type Systems and Language Translation and Execution KA.

How incorporated:

Why not incorporated:

This is just a matter of opinion between the reviewer and the expert subcommittee.

Date considered: June 9, 2023

Feedback comment:
Reviewer 4 suggested making parameter passing mechanisms CS core in the Compiler Semantic Analysis KU.

How incorporated:

Why not incorporated:

Parameter passing mechanisms are covered in The OO and functional programming, and Program Representation KU’s. The Compiler Semantic Analysis KU is focused on how these concepts may be implemented by a compiler.

Date considered: June 9, 2023

Feedback comment:
Reviewer 4 wanted to see proof by induction explicitly listed as a Math requirement, and a greater emphasis on formal proofs.

How incorporated:

Why not incorporated:

Discrete Math is listed as a prerequisite knowledge, and it includes “proof techniques”. It was decided that this included proof by induction, proof by contradiction etc, and that these did not need to be listed explicitly as the list of topics would otherwise grow to be very large and prescriptive.
Date considered: June 9, 2023

Feedback comment:
Reviewer 4 interpreted the course packaging suggestion for an introductory course on Foundations of Programming Languages to be a course aimed at novices.

How incorporated:
The first course, focused on CS Core, was renamed to “Introduction to Programming Language Concepts” to try and convey it was way to package this KA’s content, and was not intended to describe an introduction to programming course.

Why not incorporated:

Date considered: June 9, 2023

Feedback comment:
The reviewer recommended cutting back on CS Core and KA core hours.

How incorporated:

Why not incorporated:

While this could be a matter for ongoing debate, the expert subcommittee felt that what was listed was important and necessary,

Date considered: June 9, 2023

Feedback comment:
Reviewer 3 felt that the KA could not be named Foundations of Programming Languages, preferring Fundamentals of Programming Languages instead. The reviewer felt the current title implied a first course in learning how to program. In addition, the reviewer thought that the content described a comparative programming languages course.

How incorporated:

Why not incorporated:

Either Foundations of Fundamentals in the title could work. The committee decided to keep the current name. The content describes the foundational/fundamental material germane to programming languages. While the first fire KU’s could be covered in a comparative programming course, the later material might better fit a compilers course, or a formal proofs course.

Date considered: June 9, 2023.
Feedback comment:
Reviewer 3 felt 8 hours on functional programming was excessive, and wanted an increased emphasis on scripting.

How incorporated:

Why not incorporated:
There are only 4 CS Core hours in functional programming. The subcommittee felt this was appropriate. The KA core hours was not something every CS student would necessarily be expose to. The number of hours on scripting was left unchanged since it is somewhat dependent on the computing platform used within a program. Windows, Mac and Linus all have different tools. The hours suggested seem to be appropriate. However it is clear that the pathway to a programing languages course or discussion is dependent on the pathway the student took to reach that point, i.e., introductory course is OOP of FP, pr procedural.

Date considered: June 9, 2023.

Feedback comment:
Reviewer 3 suggested that students should have experience with at least 2 OO languages.

How incorporated:

Why not incorporated:
The KU identifies topics for discussion and does not attempt to prescribe how the content is delivered and what the best pedagogical approach for its delivery is.

Date considered: June 9, 2023.

Feedback comment:
Reviewer 3 suggested that students should have experience with at least 2 functional languages.

How incorporated:

Why not incorporated:
The KU identifies topics for discussion and does not attempt to prescribe how the content is delivered and what the best pedagogical approach for its delivery is.

Date considered: June 9, 2023
Reviewer 3 commented on the 3 KA core hours for logic programming, compared to the 2 CS Core hours for scripting. The argument was that scripting is more important than logic programming.

**How incorporated:**

**Why not incorporated:**

Scripting is CS core – all Cs students should see it – logic programming is KA core – not all CS students will necessarily see it unless an institution decides to focus on this KU within the curriculum. The subcommittee decided to leave the hours unchanged.

**Date considered:** June 9, 2023

---

**Feedback comment:**

Reviewer 3 wanted more detail in the list of topics in the scripting KU.

**How incorporated:**

**Why not incorporated:**

This was not altered since that are tools in some OS systems to help automate this, and there are differences between Windows, MacOS and unix (and linux variants). The major common topics are listed and OS centric tools etc are deliberately avoided.

**Date considered:** June 9, 2023

---

**Feedback comment:**

Reviewer 3 suggested the removal of “Use of sophisticated (complex) type systems, e.g., Rust” in the Type Systems KU.

**How incorporated:**

Agreed. Line deleted.

**Why not incorporated:**

**Date considered:** June 9, 2003.

---

**Feedback comment:**

Reviewer 3 felt that the Code Generation KU was too reflective of procedural programming.

**How incorporated:**

**Why not incorporated:**
While this is somewhat true, it provides the foundation for more advanced conversations related to implanting compilers for OO languages, functional programming languages, and logic programming languages etc. the subcommittee wanted to cover the basics, and not go into excessive details that are biased toward one paradigm such as OO. What is listed in the KU is necessary to support all of the paradigms.

**Date considered:** June 9, 2023.

**Feedback comment:**

Reviewer 3 noted that “An important modern topic is input i.e. how do we enter a program? Specifically, these days, young children use phones and select keywords from blocks (see MakeCode). That affects the program design and the translation. If not here, then the topic should be picked up somewhere.” With respect to the Language Pragmatics KU.

**How incorporated:**

**Why not incorporated:**

The KU is focused on the language, not how the program is written. However, the KU does already state, “Effect of technology needs and software requirements on programming language development and evolution” as a non-core topic which opens the door for a program to address the topic the reviewer raises.

**Date considered:** June 9, 2023

**Feedback comment:**

Reviewer 2 identified a couple of spelling errors.

**How incorporated:**

These were corrected.

**Why not incorporated:**

**Date considered:** June 12, 2023

**Feedback comment:**

Reviewer 2 suggested that more details should be given regarding pipelining in the KU Functional Programming.

**How incorporated:**

**Why not incorporated:**

The reviewer did not provide specifics and the subcommittee felt that the term was sufficiently clear.
**Date considered:** June 12, 2023

---

**Feedback comment:**

Reviewer 2 asked if the intent of the KU Program Representation was to have students write their own compiler, in which case the competency level would be develop rather than explain.

**How incorporated:**

Students could develop their own compiler, design their own language, or the instructor may choose to explain how a compiler works. All are feasible and dependent on the program’s objectives. The level of competency was left at “explain” which would permit all 3 scenarios. A program is always welcome to go into more detail. However, discussion amongst the expert subcommittee decided to move some content to a new KU called Systems Programming to make it clear that it dealt with the underlying representation and rename Program Representation to Program Abstraction and Representation.

**Why not incorporated:**

---

**Feedback comment:**

Reviewer 1 and 2 asked if there was a definition of mobile coding in the KU Program Representation, and suggested the topic be dropped

**How incorporated:**

The committee could see how this phrase may cause confusion. The topic was dropped.

**Why not incorporated:**

---

**Feedback comment:**

Reviewer 1 felt that mobile coding (KU Program Representation) was a hot topic in the 2000’s but questions its relevance in the CS2023 document.

**How incorporated:**

**Why not incorporated:**

No change made. The subcommittee felt it was still relevant and was a topic students should have an awareness of. It is relevant to cloud computing, and is still relevant to virus propagation.

**Date considered:** June 12, 2023.
Feedback comment:
Reviewer 1 identified a statement in the KA’s preamble that suggested that Logic Programming was now part of the CS Core.

How incorporated:
The paragraph was deleted to be consistent with the current document which as logic programming as KA core, not CS core.

Why not incorporated:

Date considered: June 12, 2023

Feedback comment:
Reviewer 1 commented on the use of the terms “traits and mixins” in the KU Object Oriented Programming suggesting that they are not taught in most OOP courses.

How incorporated:

Why not incorporated:
The terms were left unchanged. The subcommittee felt that the concepts were important for students to be aware of them.

Date considered: June 12, 2023

Feedback comment:
Reviewer 1 suggested that subtyping should be CS Core, not KA core, in the KU Object Oriented Programming.

How incorporated:
We agree; the topic is now CS Core.

Why not incorporated:

Date considered: June 12, 2023

Feedback comment:
Reviewer 1 asked what the difference is between higher-order functions (listed as CS core) in the KU Functional Programming, and passing functions as parameters (listed as KA core).

How incorporated: There is none, the reference in the KA core was deleted. The Number of KA core hours required was adjusted downward accordingly.
Why not incorporated:

Date considered: June 12, 2023.

Feedback comment:
Reviewer 1 referred to the mention of SECD machine (KU Functional Programming) as 20\textsuperscript{th} century.

How incorporated:

Why not incorporated:

SECD machines are influential abstract machines for functional programming. The subcommittee felt that the concept was relevant and helps students to understand how functional programming languages might be implemented.

Date considered: June 12, 2023

Feedback comment:
Reviewer 1 comments on learning objective 11 in the KU Functional Programming pointing out that it is not a purely FP concept.

How incorporated:

The point of the learning outcome was so that students would understand the pros and cons of various paradigms with respect to concurrency. A cross reference to FPL-A was added.

Why not incorporated:

Date considered: June 12, 2023.

Feedback comment:
Reviewer 1 felt that the KU Scripting had a confusing name (bash or python). They also felt the content should be distributed across other KUs

How incorporated:

Why not incorporated:

The KU Scripting covers both bash and python etc. It is about the ability to write scripts to manipulate the file system, operating system commands, redirecting output etc. The subcommittee considered the list of topics and reworked them to be more cohesively presented.

Date considered: June 12, 2023.
Feedback comment:
Reviewer 1 suggested dropping model, view controller from the KA-core of the KU Event-Driven and Reactive Programming, because they have found it difficult to teach coherently.

How incorporated:

Why not incorporated:
This was left. Unchanged. It is part of the KA-core and hence not every student needs to see the content, but it is an important topic related to event-driven and reactive programming.

Date considered: June 12, 2023

Feedback comment:
Reviewer 1 was confused about a topic in the CS-Core in KU Type Systems which spoke about implementing compound types from more basic types.

How incorporated:

Why not incorporated:
The subcommittee left the topic unchanged.

Date considered: June 12, 2023

Feedback comment:
Reviewer 1 suggested adding subtyping to the list of subtopics related to generic types.

How incorporated:
Agreed. Subtyping was added.

Why not incorporated:

Date considered: June 12, 2023

Feedback comment:
Reviewer 1 identified Hindley-Miller and Girard-Reynolds type systems as advanced.

How incorporated:
The type systems were deleted form the KU Type Systems.

Why not incorporated:

Date considered: June 12, 2023
Feedback comment:
Reviewer 1 identified an incorrect word in the KU FPL and SEP

How incorporated:
Epistemology was replaced by Etymology

Why not incorporated:

Date considered: June 12, 2023

Feedback comment:
Reviewer 1 commented on a proposed course packaging suggestion where a topic they felt should bit be included and more time spent on other topics

How incorporated:

Why not incorporated:
This is entirely why the KA suggested a possible packaging of content and did not prescribe a syllabus. Faculty are free to package content in ways that make sense for their program.

Date considered: June 12, 2023

Feedback comment: Core hour survey: Object-oriented programming

How incorporated:

Why not incorporated:
No changes needed. The survey results were very strongly in support of the topics identified.

Date considered: July 3, 2023

Feedback comment: Core hour survey: Functional programming

How incorporated:

Why not incorporated:
The survey results were strongly in support of the topics identified.

Date considered: July 3, 2023
Feedback comment: Core hour survey: Scripting

How incorporated:

There was reasonable support for this topic and the majority of the topics within the KU. Code objects was deleted since it was not supported by eth majority of the survey respondent’s. Other topics were discussed and changes made to make the subcommittees intent clearly and

Why not incorporated:

Date considered: July 3, 2023

Feedback comment: Core hour survey: Event-driven and reactive programming

How incorporated:

There was agreement (100%) that the topic should be required for all Cs majors, but components of the KU received various levels of support. The subcommittee discussed the topic at length and decided that to provide graduates with an understanding of the topic then the components were necessary. These components included: components of a reactive system, behavior model of event-based programming, canonical uses such as GUI’s, mobile devices, robots, etc. Reactive programming as a state transition was dropped as a topic.

Why not incorporated:

Date considered: July 3, 2023

Feedback comment: Core hour survey: Parallel and distributed programming

There was strong support for this KU. The only topics questioned were programming models and semantics.

How incorporated:

Why not incorporated:

The subcommittee discussed this at length, and in conjunction with the chair of the parallel and distributed computing KA. It was decided to leave these topics in place within the KU as they provide important foundational knowledge for graduates.

Date considered: July 3, 2023

Feedback comment: Core hour survey: Type systems

The KU type systems received extremely strong support with the exception of the topic “Goals and limitations of static and dynamic typing”.

How incorporated:
It appears that the survey results indicate that graduates should be able to explain the differences in a relatively superficial manner, while the subcommittee felt that a slightly deeper understanding was necessary to properly prepare students for the length of their working life. However, the topic structural vs name equivalence was dropped as CS Core and moved to KA core since it is likely covered in the topics related to type safety.

**Why not incorporated:**

**Date considered:** July 3, 2023

---

**Feedback comment:** Core hour survey: Language translation and execution

There was fairly strong support for most of the topics within the KU Language Translation and Execution. Some topics were viewed less favorably.

**How incorporated:**

The subcommittee reviewed the feedback and decided that the systems related content (memory management etc) which had strong support should be pulled out into a new KU called Systems Programming. Of the content that remained, some was removed in response to the survey and the topics within the KU were deemphasized. The hours remaining in the KU were designated as KA-core rather than CS-core.

**Why not incorporated:**

**Date considered:** July 3, 2023

---