Q&A Session for Programming Languages Lecture 1

Session Number: 1204954699
Date: 2020-9-1
Starting time: 14:19

ANON - 14:36
Q: Why has Submitty been so slow? Will it be resolved soon?
Priority: N/A
Konstantin Kuzmin - 15:34
A: Yes, Submitty admins are already working on improving Submitty performance.

ANON - 14:37
Q: Will we be notified when in-class quizzes will occur on Submitty?
Priority: N/A
Konstantin Kuzmin - 15:36
A: All quizzes will be at the same time (the beginning of the lecture) on fixed dates which are listed in the schedule: https://www.cs.rpi.edu/~milanova/csci4430/schedule.html

---

ANON - 14:37
Q: Where is the syllabus posted? Will the lecture be recorded and posted for students who is not in the US?
Priority: N/A
Konstantin Kuzmin - 15:36
A: Syllabus is here: https://www.cs.rpi.edu/~milanova/csci4430/syllabus.htm
Konstantin Kuzmin - 15:37
A: All lectures are pre-recorded. All live sessions will also be recorded and made available to the students.

ANON - 14:38
Q: will the lecture videos be available after the fact?
Priority: N/A
Konstantin Kuzmin - 15:37
A: Yes.

ANON - 14:38
Q: How will the attendance to office hour work? Do we just jump in an office hour queue and say hi once a week?
Priority: N/A

Saksham Chawla - 15:48
A: Essentially, yes. You jump in any office hour queue in the week and "check-in". It's okay if you don't have any questions, and there's no limit on how many times you can visit us during office hours.

ANON - 14:42
Q: where is this slide show found?
Priority: N/A
Konstantin Kuzmin - 16:12
A: https://www.cs.rpi.edu/~milanova/csci4430/Lecture1.pdf

ANON - 14:44
Q: nevermind
Priority: N/A

ANON - 14:46
Q: How do you submit a time zone?
Priority: N/A
Konstantin Kuzmin - 15:39
A: https://submitty.cs.rpi.edu/user_profile

ANON - 14:47
Q: Assuming homeworks won't be due until Submitty is functional again?
Priority: N/A
Konstantin Kuzmin - 15:39
A: It appears to be back online.

ANON - 14:56
Q: will the pre-recorded lectures always be available on mediasite?
Priority: N/A
Konstantin Kuzmin - 15:38
A: Yes.

ANON - 15:04
Q: The syllabus says the HW is due at 2pm on the date it is due. I just want to confirm this is correct and it should not be 2am?
Priority: N/A
Ana Milanova - ?
A: Correct. HW is due at 2pm Eastern Time NOT 2am.

ANON - 15:13
Q: Is something wrong with my Webex? I can only see myself as a attendee, and only my own questions in the Q&A box
Priority: N/A

Steven Haussmann - 15:36

A: Questions in the Q&A box should appear for everyone once they're answered.

ANON - 15:15
Q: Should we be using the chat or should we be using the Q&A section?
Priority: N/A

Steven Haussmann - 15:38

A: Go ahead and ask questions in the Q&A box. This will ensure everyone can see answered questions, and keep Q's and A's together.

ANON - 15:33
Q: Do functional programming languages have better performance than imperative ones like C even if both are compiled?
Priority: N/A

Steven Haussmann - 15:42

A: This is quite a broad question -- performance is often worse due to overhead (immutability can be expensive!), but it's often possible to approach or equal the performance of things like C.

Ana Milanova - ??

A: To add to Steven's answer, generally, functional languages have worse performance. They add a level of abstraction (in their inherent features) and abstraction always comes at performance cost. As Steven says, one essential feature of functional languages is being "side-effect free". This prevents bugs due to unintended mutation, but it comes at significant cost --- e.g., we cannot change one position of an array, we need to create a new array with a different value at that one position. Another feature is the automatic memory management, i.e., garbage collection, which also comes at significant cost.

ANON - 15:40
Q: Do functional languages have significant differences in how their compilers/ interpreters work compared with imperative languages?
Priority: N/A

Ana Milanova - ??:??

A: This is a broad question and I might be misinterpreting. Generally, yes, there are significant differences in how functional vs. imperative languages are implemented/ translated. Functional languages are typically interpreted (but not always, e.g, Haskell) as their features lend them more suitable to interpretation. Imperative languages are often compiled (but again, not always, e.g., Python) as their features lend them more suitable to compilation. Then there would be significant differences in different parts of

the translation, different kinds of semantic analysis that takes place at different levels. But there are also commonalities --- e.g., both functional and imperative languages employ syntax analysis based on formal languages.

ANON - 15:40
Q: We just asking questions in here?
Priority: N/A
Ana Milanova - ??:??
A: Yes. Use the Q&A box from now on!

ANON - 15:41
Q: What's the difference between imperative and declarative language?
Priority: N/A
Ana Milanova - ??:??
A: Generally (and I'd add that modern languages blur the distinction), imperative languages are lower-level as they give significant control to the programmer on memory access and algorithmic steps. Declarative languages are higher-level as they provide an "interface" that allows the programmer to specify the problem, but hide memory access and algorithmic steps, i.e., "how" the problem is solved; there is an inference engine that solves the problem.

Since this question came up a lot, here is a concrete example. Suppose we are trying to find whether there is a path from node X to node Y in a graph. In C, we'll encode the graph, using one of the many data structures we have for that, then we'll code the steps of one of the many graph search algorithms. In Prolog (declarative), we'll specify the graph, in the limited way provided by Prolog, then we'll write a set of logical rules that specify what does it mean to have a path from A to B: (1) there is a path from node A to node A for each A, and (2) there is a path from node A to node B, if there is an edge from A to some node C, and a path from C to B. Then we'll ask the question "Is there a path from X to Y and Prolog will answer with Yes or No. The actual search happens in the Prolog inference engine, behind the scene and we don't have to know how Prolog searches for a path.

ANON - 15:41
Q: What category do languages that are Just in Time (JIT) compiled fall into?
Priority: N/A
Steven Haussmann - 15:47
A: I would not say that being JIT-able implies anything -- JITing is just the process of performing compilation during execution.
Ana Milanova - ??:??
A: To add to Steven's answer, in theory, implementation (i.e., compilation, including JIT compilation, or interpretation) is orthogonal to the language. In theory, we can build an interpreter for every language and we can build a compiler (including JIT) for every

language. That said, some languages are more suitable to interpretation and other languages are more suitable to compilation. E.g., Python and JavaScript which are very dynamic, are naturally interpreted while C/C++ and Fortran, which are more static, are naturally compiled. Much more on dynamic vs. static, and compilability will come later in the class.

ANON - 15:42
Q: In an earlier slide, professor milanova said imperative languages are generally lower level languagaes, but then when describing FORTRAN, she said it was the 1st high level programming language....I don
Priority: N/A

Steven Haussmann - 15:50

A: FORTRAN is high-level because it's not a direct description of machine instructions. It's still less abstract than many other languages, however, so we say it is "lower-level" compared to them.

Ana Milanova - ??:??

A: Yes. "High-level" vs. "low-level" here refers to the level of abstration a language employs. "High-level" adds abstraction, i.e., it hides underlying architecture or algorithmic details. FORTRAN is "high-level" compared to machine code which is "low-level". Before FORTRAN and the FORTRAN compiler, programming was done essentially in machine code, which was very difficult and error prone (imagine writing x86!). FORTRAN raised the level of abstraction, i.e., we have variables with meaningful names and operations on those variables, etc., and the COMPILER did the translation from high-level to low-level machine instructions. This made programming much more accessible to computational scientists. In the same time, FORTRAN is "low-level" compared to declarative languages which add even more levels of abstraction, i.e., hide even more of the underlying architecture and algorithms from the programmer.

ANON - 15:42
Q: oops -- I don't understand the distinction
Priority: N/A

Ana Milanova - ??:??

A: I hope the above makes the distinction clearer. This will become more concrete when we work with an actual declarative language.

ANON - 15:42
Q: For compilation, could you just explain what the target machine is?
Priority: N/A

Steven Haussmann - 15:43

A: The "target machine" is whatever is actually going to run the program.

Ana Milanova - ??:??

A: Yes, the target machine instruction set. E.g., x86 64, ARM. E.g., on an x86

processor the compiler compiles to program into an x86 binary.

ANON - 15:42
Q: I'm still a little confused about imperative and declarative. This is a difference between how code is designed, not between different languages?
Priority: N/A

Steven Haussmann - 15:46

A: They're ways in which you can use a language to describe a program. A language can fall into several paradigms -- e.g., Java is object-oriented and imperative.

Ana Milanova - ??:??

A: Yes, these terms characterize the programming languages. The difference is in the languages. Different languages have made different design choices and these choices affect how the programmers can use the language to do things. You can think of imperative vs. declarative as design choices.

ANON - 15:42
Q: If only the host can see the full list of attendees, how do you suggest we find group mates for studying and collaboration if we don't know who is in the class?
Priority: N/A

Ana Milanova - ??:??

A: You can try the Submitty forum if you don't already know students in the class.

ANON - 15:43
Q: What forms will the quizzes take?
Priority: N/A

Ana Milanova - ??:??

A: Plan is for Submitty notebooks and a mix of multiple choice and short answer.

ANON - 15:43
Q: is it possible to get the slides in a powerpoint file so we can adjust the way the print rather than 6 slides per page like in the pdf file?
Priority: N/A

Ana Milanova - ??:??

A: Yes, will do starting from Lecture 2.

ANON - 15:43
Q: I was also having a lot of weird issues with playing the video, would vote for Events if possible
Priority: N/A

Ana Milanova - ??:??

A: Thanks.

ANON - 15:45
Q: When is the abstract syntax tree used and when is the intermediate form used?
Priority: N/A

Steven Haussmann - 15:56

A: The AST is a representation of parsed code. It's how the parser interpreted its input. An intermediate form is something like Java's bytecode -- it is not machine code, but it *has* been produced from something else.

Ana Milanova - ??:??

A: To add to Steven's answer. More on the AST later in class. AST is used in the semantic analysis front-end phase (to do various analysis such as type checking, and to translate into intermediate form). Intermediate form is used later in the back end to generate target machine code.

ANON - 15:46
Q: What is the difference between Imperative vs declarative?
Priority: N/A

Ana Milanova - ??:??

A: Addressed already.

ANON - 15:50
Q: Are we allowed to use the discussion forum to make study groups since we aren't allowed to see the class list?
Priority: N/A

Ana Milanova - ??:??

A: Yes

ANON - 15:50
Q: On the compilation workflow slide, why is the "machine independent code improvement" step part of the compiler backend/machine dependent slide?
Priority: N/A

Ana Milanova - ??:??

A: This is somewhat imprecise you are right. In some books it's part of the back-end, and in other books, it has its own "end", a middle-end. The front-end is the specific language-dependent stuff: each language has its own unique syntax rules, semantic rules, type system, etc. So these language-dependent stuff goes into the front-end. The back-end is the machine-specific stuff: target machine instruction set, register allocation optimizations, etc. So these are strictly machine-dependent. The "machine independent

code improvement" is kind of in the middle because it's neither language-dependent nor machine-dependent. It can go at either end and for simplicity and symmetry I put it in the back-end.

ANON - 15:51
Q: There is some lecture 8 video on mediasite. is that just a mistake and we won't do that one until Lecture 8
Priority: N/A
Ana Milanova - ??:??
A: Removed. That was a mistake.

ANON - 15:54
Q: ^^ it is slide 31 btw
Priority: N/A

ANON - 15:54
Q: Will this course use LMS at all?
Priority: N/A
Steven Haussmann - 16:20
A: Not that I'm aware of. Homework will be collected via Submitty.

Jiarui Ruan(ruanj4@rpi.edu) - 15:54
Q: If it is possible to change from 6 slides each page to 1 or 2 slides each page of lecture note PDF which is posted on Website in the future?
Priority: N/A
Ana Milanova - ??:??
A: Addressed already, YES.

ANON - 15:54
Q: I'm not sure if this was already asked, but are the Pre-Recorded Lectures released before Lecture, or just the Lecture Slides?
Priority: N/A
Steven Haussmann - 16:00
A: The lectures were put online before the lecture for today. It looks like they aren't currently linked on the course website, however. We'll check on that.
Ana Milanova - ??:??
A: Will add a link to Proglang Mediasite from course website.

ANON - 15:55

Q: Is there a quiz next class?
Priority: N/A
Ana Milanova - ??:??
A: No.

ANON - 16:02
Q: The syllabus says groups of 5-6 students for quizzes, but Professor Milanova said 6-7. Do we go with 6-7 or should we go with 5-6 for quiz groups?
Priority: N/A
Ana Milanova - ??:??
A: Sorry, take the max: 7.

ANON - 16:03
Q: Is it possible to post the pre recorded video earlier? Or post part of the video earlier so that we can bring question to class? I just find the pace is a little too fast for me to take notes.
Priority: N/A
Ana Milanova - ??:??
A: We will address in staff meeting what is a good time ahead of class to post the videos. Shouldn't be a problem to post earlier.

ANON - 16:04
Q: Regarding the previously answered question, why is declarative considered what is happening (the logic) and imperative how it is happening? This is in reference to the previous answered question. I do not understand.
Priority: N/A
Steven Haussmann - 16:06
A: A great example of the declarative style is SQL. You don't tell the database how, exactly, the query should be performed; you just tell it what action it should perform.
Ana Milanova - ??:??
A: Steven's example is great. I added another example earlier in the Q&A.

ANON - 16:05
Q: wait so once a week we have to go to office hours to at least say hi?
Priority: N/A
Saksham Chawla - 16:08
A: Yes, starting week 3.

ANON - 16:12
Q: The compilation workflow at slide 33 shows both an abstract syntax tree AND an intermeidtae form. Is an intermediate form always expressed for all languages? Or only for hydrib interpreted and compiled languages?
Priority: N/A

Steven Haussmann - 16:19

A: The AST is how your program was parsed. The intermediate form (or intermediate forms) are produced by the compiler after it interprets your program.

ANON - 16:15
Q: Should we email our Group Members to Professor Milanova?
Priority: N/A

Ana Milanova - ??:??

A: No.

ANON - 16:16
Q: so are office hours virtual, and if so, how do we join them?
Priority: N/A

Steven Haussmann - 16:18

A: You'll use Submitty to join a queue. I believe that the actual meeting will occur via personal webex rooms.

ANON - 16:20
Q: What is the difference between the compiler front end and the compiler back end? How is the distinction made for which compilation steps are classified in which category?
Priority: N/A

Ana Milanova - ??:??

A: Front-end phases are language dependent: what is the syntax of the language, what kind of type checking it does, etc. Back-end is machine-dependent: whether we are producing an x86-32 or x86-64, or ARM, or another instruction set binary.