Q&A Session for Programming Languages Lecture 25

Session Number: 1204254990
Date: 2020-12-8
Starting time: 14:21

ANON - 14:33
Q: Functions in C cannot have function arguments or function returns, right? only function pointer arguments and returns are allowed?
Priority: N/A
	Steven Haussmann - 14:35
	A: Yes, they can't accept functions or return functions directly. They also cannot create new functions at runtime. They can, however, accept and return pointers to functions.

ANON - 14:35
Q: hi. would you also be able to explain the order in which we are able to apply the different types. I think I was sent something called spiral method or something
Priority: N/A
	Steven Haussmann - 14:35
	A: The spiral rule, yes. You start at the identifier and work outward in a clockwise direction. If you search it, you should find a page describing the specifics.

ANON - 14:38
Q: what happens to the paranthesis around *v?
Priority: N/A
	Steven Haussmann - 14:39
	A: I believe those were just enforcing order of operations.

ANON - 14:39
Q: what is tau?
Priority: N/A
	Steven Haussmann - 14:39
	A: τ is a type variable.

ANON - 14:40
Q: yes could we review a concrete example
Priority: N/A

ANON - 14:41
Q: When we write out a type, should we write it in one sentence or use

intermediate types tau1, tau2, ... for clarity?
Priority: N/A
        Steven Haussmann - 14:40
        A: Writing it out as a series of type variables makes it a lot easier to look at, so I would suggest it. You can also write it out as a single (possibly heavily nested) sentence to tie it all together, though

---

ANON - 14:42
Q: what prevents you from applying the reference * before the ( ) and [ ] (why is the order of operations to the right of *v)?
Priority: N/A
        Steven Haussmann - 14:42
        A: * has the lowest precedence of any operator

ANON - 14:44
Q: How would the description of the chunkfun type declaration go again? I missed when she said it, I only have the type tree right now
Priority: N/A
        Ana Milanova - ?
        A: A pointer to a function that takes () and returns a pointer to struct _chunk.

ANON - 14:45
Q: so when the homework asks us to (in question 4) to explain the meaning in addition to drawing the trees, we can explain it as a series of type variables?
Priority: N/A
        Steven Haussmann - 14:47
        A: That's ideal, yes.

ANON - 14:46
Q: if it's * [] a () [] which would you do first: [] a or a ()
Priority: N/A
        Steven Haussmann - 14:57
        A: This is not syntactically valid -- you can't put [] on the left side of the identifier like that

---

ANON - 14:48
Q: I think I zoned out for a second...the english interpretation of this example would be "chunkfun is a pointer to a function that takes no arguments and returns a pointer to struct _chunk"?
Priority: N/A
        Steven Haussmann - 14:49
        A: Yes, that's right

ANON - 14:48
Q: in C, is a pointer always returned?
Priority: N/A
Steven Haussmann - 14:48
A: Not quite sure what you mean here -- pointers can be returned from functions, but so can other values (e.g. an int)

ANON - 14:50
Q: If *a[] === *(a[]) why did prof. milanova write (*a)[]?
Priority: N/A
Steven Haussmann - 14:50
A: It can be useful to make things more explicit -- but yes, it's not necessary in this case
Ana L. Milanova - 14:59
A: If you write *a[] that means an array of pointer. But if you want a pointer to array, then you need (*a)[]. Parens force * to take precedence over [].

ANON - 14:51
Q: int *(* v () []) would mean that v is a Function that takes () and returns an Array of Pointers to Pointers to int?
Priority: N/A
Steven Haussmann - 14:52
A: Yes, that's correct.

ANON - 14:53
Q: As a follow up, can you return an int in C or does it have to be a pointer to an int?
Priority: N/A
Steven Haussmann - 14:54
A: Yes -- consider how pointers are just long integers, after all!

ANON - 14:55
Q: when type declarations in the homeowork have like [n] with the n as the size of the array I believe, is that something that needs to be stated in the type declaration or can we just call it an array?
Priority: N/A
Steven Haussmann - 14:55
A: Go ahead and include it in the description.

ANON - 14:55
Q: Is there a specific reson the reading on slide 1 is optional? Does

that mean dynamic languages won't be on the final?
Priority: N/A
        Ana L. Milanova - 15:44
        A: The chapter is long and I decided to make it optional reading. Generally, all material in lecture 25 (except for Wat I guess) may be on the final.

ANON - 14:56
Q: Does the same logic apply to more complex types like doubles? can a function return a double in C or does it have to be a pointer to a double?
Priority: N/A
        Steven Haussmann - 14:55
        A: Yes -- and, in general, I believe you can return things like structs, too

---

ANON - 14:58
Q: what are components?
Priority: N/A
        Steven Haussmann - 14:58
        A: Any bits of software.

ANON - 15:01
Q: is programming langauges research very different than computer science research?
Priority: N/A
        Steven Haussmann - 15:01
        A: It's a category of computer science research.

ANON - 15:05
Q: how does the rise of dynamic langauges development reflect the changing needs of the computer science community?
Priority: N/A
        Ana L. Milanova - 15:15
        A: The rise was driven mostly by the rise of the web, I believe. However, those languages have taken hold in many different domains, e.g., statistics.

ANON - 15:13
Q: What does the last eval say?
Priority: N/A
        Ana L. Milanova - 15:15
        A: I believe you are referring to eval(v1+"="+v2). I don't think this is correct syntax though. A correct one will be eval("v1="+v2). Interestingly, this is interpreted as eval("v1=v2").

Wat?

ANON - 15:14
Q: what is eval?
Priority: N/A
Steven Haussmann - 15:14
A: "eval" takes a string and interprets it as code. It can be useful for metaprogramming -- getting variables by name, for example

---

ANON - 15:14
Q: Was it eval(v1.f "=" v2.f)?
Priority: N/A
Ana L. Milanova - 15:26
A: I believe this will be a syntax error (but I'm not sure about anything in JavaScript). But if you write eval("v1.f=v2.f") that will be interpreted as the assignment v1.f = v2.f.

ANON - 15:18
Q: don't static languages have data structures as well?
Priority: N/A
Ana L. Milanova - 15:47
A: They often do. Languages often become popular because of the availability of high-level data structures, libraries, and modules that support development.

ANON - 15:19
Q: how is building a static langauge different than a dynamic langauge?
Priority: N/A
Steven Haussmann - 15:20
A: Static languages generally do more typechecking and general analysis at compile time (and dynamic languages might not even be compiled in the first place)

ANON - 15:20
Q: what's the different between the Java interpreter and Java?
Priority: N/A
Ana L. Milanova - 15:29
A: What that slide referred to was interpreters for the dynamic language that were written in Java. I.e., the interpreter is written in Java.

ANON - 15:28
Q: So f.y makes an attribute/variable y of the function which is why

f(f.y) = 91?
Priority: N/A
        Ana L. Milanova - 15:48
        A: Yes, that is correct.

ANON - 15:28
Q: should we be familiar with JavaScript? I haven't used it before
Priority: N/A
        Ana L. Milanova - 15:49
        A: No. There won't be any JavaScript on the exam. But you should be familiar with Python --- scoping, parameter passing, functions as first class values, etc.

ANON - 15:28
Q: the video seems stuck for me?
Priority: N/A
        Steven Haussmann - 15:29
        A: It hung for a bit for me, but it's alive again. You can always pull it up elsewhere if it's not working on here, though!
        Ana L. Milanova - 15:31
        A: It got stuck on Ruby but went fine for the rest of the presentation. Link: https://www.destroyallsoftware.com/talks/wat
---

ANON - 15:29
Q: Wat
Priority: N/A

ANON - 15:29
Q: Playing a video inside a video is not good for the webex :(
Priority: N/A

ANON - 15:30
Q: I need the comedian's name!
Priority: N/A
        Ana L. Milanova - 15:31
        A: https://www.destroyallsoftware.com/talks/wat
        Ana L. Milanova - 15:51
        A: Gary Bernhardt

ANON - 15:51
Q: How do you know if it will be printed as a Tuple or not?
Priority: N/A
        Ana L. Milanova - 15:53
        A: m,j,k creates a tuple and "return m,j,k" returns that

tuple. print(outer()) prints the tuple.
Steven Haussmann - 15:53
A: whilst, conversely, print(m, j) will just print two numbers

---

ANON - 15:53
Q: If we wanted to import m not from global scope but from the scope of outer, what is the syntax? outer m?
Priority: N/A
Steven Haussmann - 15:54
A: g;pba;
Steven Haussmann - 15:54
A: oops, ignore my gibberish answer (:
Steven Haussmann - 15:57
A: Anyway, yes, there is a keyword for this: nonlocal. I just found out about that!
Ana L. Milanova - 15:58
A: I don't think one can do that in Python. I did check and I will continue checking, but my intuition is that we shouldn't be able to do that. Dynamic languages strive for simple scoping rules, e.g., some only have locals, and other languages only have globals. Python introduced the "local by assignment" rule to make things simpler, so I would expect it won't care for more complex constructs to provide access to intermediate scopes. But I am not sure, I will continue to look to make sure.

ANON - 15:54
Q: What about print (m, j, k) why is that not a Tuple?
Priority: N/A
Steven Haussmann - 15:53
A: This is just calling the print function with three arguments.

ANON - 15:56
Q: If the Middle did not call Inner and just return Inner, would everything still be the same?
Priority: N/A
Ana L. Milanova - 15:59
A: Yes, everything would be the same. The call to inner is not necessary for the point I was trying to make with this example.

ANON - 15:57
Q: Or is the Call to Inner needed for the Closure Bindings to be known?
Priority: N/A

ANON - 15:57
Q: Why do we say no varianble declarations though?
Priority: N/A

ANON - 16:08
Q: So it is not Call By Value at all. It is just Call By Sharing
Priority: N/A

Ana L. Milanova - 16:09

A: Technically it is Call By Value. The parameter copies the value of the argument, which happens to be an address. But the traditional meaning of Call-by-value implies no side effects, so that's why we have Call-by-sharing, to make the distinction.

Steven Haussmann - 16:09

A: Yes -- and this is very similar to what Java winds up doing. You can mutate non-primitive arguments, but you can't completely reassign them

---

ANON - 16:12
Q: Milanova: "If you write *a[] that means an array of pointer. But if you want a pointer to array, then you need (*a)[]. Parens force * to take precedence."
Isn't it the other way around? *a[] is a pointer to a array?
Priority: N/A

Steven Haussmann - 16:15

A: I believe it's the right way around as-is.

Ana L. Milanova - 16:20

A: Yes, the other way around. *a[] is an array of pointer. As another, more meaningful example, *a() is a function that returns a pointer. But if we want a pointer to a function, we'll have to do (*a)().

ANON - 16:14
Q: is the next class a review class?
Priority: N/A

Ana L. Milanova - 16:22

A: Yes, that's right. We'll start with the quiz, then go over a review, including answers to quiz questions.

ANON - 16:14
Q: So if we were asked on the exam whether python is call by value or call by sharing, would you accept both?
Priority: N/A

Ana L. Milanova - 16:23

A: We try to avoid questions that might be ambiguous. But if there is a question on parameter passing in Python, we'll accept both call by value and call by sharing.

ANON - 16:27
Q: In Python, why do we say no varianble declarations though?
Priority: N/A
Ana L. Milanova - 16:33
A: There is no explicit declaration. The declaration is implicit: a variable is local to the block where it is defined. We can think of an assignment, e.g., m = 2, as a declaration of local variable m followed by initialization to 2.

ANON - 16:30
Q: I had another quick question - what was the purpose of the wat video? to make fun of JS?
Priority: N/A
Ana L. Milanova - 16:34
A: To illustrate that there are lots of strange things happening in JavaScript that don't make a lot of sense.