Types and Programming Languages (15-814)
Fall 2018
Assignment 0: The untyped λ-calculus

Contact: 15-814 Course Staff

Due Thursday, September 13, 2018

This assignment is due at the beginning of class on the above date and it must be submitted electronically as a PDF file on Canvas. Please use the attached template to typeset your assignment and make sure to include your full name and Andrew ID. Questions subject to the whiteboard policy are marked by "WB".

15-814 Programming Language Award

The Programming Language Award recognizes a programming language that has had significant influence on computing, as reflected in contributions to academic research, commercial acceptance, or both. The award is presented each fall at the 15-814 Awards Brunch and is accompanied by a prize of one lecture dedicated to the chosen language, plus food expenses at the brunch.

Next Deadline Thursday September 13, 2018, 10:30am EDT.

Selection Criteria Nominations will be reviewed for the evidence they provide of originality, significant conceptual impact, influence on related developments, beauty, and elegance.

Submissions Nominations for the Programming Language Award should be submitted on Canvas. Submitted materials should explain the contribution in terms understandable to a non-specialist. Each nomination involves several components.

- Name and Andrew ID of the nominator.
- Name of the programming language nominated.

- URL with key information about the language, which could reference a paper, home page, or an implementation.
- Suggested citation if the language is selected. This should be a concise statement (maximum of 25 words) describing the language and its benefits that have had impact. Note that the final wording will be at the discretion of the Award Committee.
- Nomination statement 450 to 500 words in length addressing why the language should receive this award. This should draw particular attention to the contributions that merit the award.
- Brief timeline with key dates, individuals responsible for its creation, publications, websites, implementations, etc.
- Names of at least 3 and not more than 5 researchers or practitioners who might, if asked, write a brief letter of endorsement of the language. Hypothetical endorsers should be chosen to represent a range of perspectives and institutions and could provide additional insights or evidence of the language's significance. Do not contact hypothetical endorsers!

For questions on the above, please contact us on Piazza. 15-814's academic integrity guidelines apply to all award nominations.

Task 1 (20 points)**.** Nominate a language of your choice for the Programming Language Award.

Programming with the λ-calculus

In class, we saw how to encode the natural numbers using Church numerals. We will explore various other representations of numerals.

Task 2 (5 points, WB)**.** Define functions pair, projl, and projr such that projl $(\mathsf{pair}\ x\ y) = x$ and projr $(\mathsf{pair}\ x\ y) = y$.

Using pairings, we can now define numerals:

$$\ulcorner 0 \urcorner = \lambda x.x \qquad\qquad \ulcorner n+1 \urcorner = \mathsf{pair}\ \mathbf{F}\ \ulcorner n \urcorner$$

where

$$\mathbf{T} = \lambda x.\lambda y.x \qquad\qquad \mathbf{F} = \lambda x.\lambda y.y$$

are the booleans from class.

Task 3 (5 points, WB)**.** Implement the successor, predecessor, and test for zero functions:

$$\mathbf{S}\ulcorner n\urcorner = \ulcorner n+1\urcorner, \qquad \mathbf{P}\ulcorner n+1\urcorner = \ulcorner n\urcorner,$$
$$\mathbf{Z}\ulcorner 0\urcorner = \mathbf{T}, \qquad \mathbf{Z}\ulcorner n+1\urcorner = \mathbf{F}.$$

This numbering system is equivalent to system of the Church numerals we saw in class, where we defined:

$$\bar{0} = \lambda z.\lambda s.z$$
$$\overline{n+1} = \lambda z.\lambda s.s(\bar{n}zs)$$

The next task asks you to implement a function mapping between our two definitions of natural numbers.

Task 4 (10 points, WB)**.** Implement functions H and H^{-1} such that $H\overline{n} = \ulcorner n\urcorner$ and $H^{-1}\ulcorner n\urcorner = \overline{n}$ for all n. Then implement the predecessor function pred for Church numerals, satisfying $\mathsf{pred}\ \overline{n+1} = \overline{n}$.