ICON

Ben Taylor
Computer Science
University of Arizona
Tucson, AZ 85719
`bentaylor012@email.arizona.edu`

Lize Chen
Computer Science
University of Arizona
Tucson, AZ 85721
`lizechen@email.arizona.edu`

April 6, 2020

ABSTRACT

This language is as cool as us!

1 Introduction

2 History

Icon is a high-level, general-purpose programming language with novel features including string scanning and goal-directed evaluation. It was chiefly designed by Ralph Griswold in 1977 at the University of Arizona. The design philosophy of Icon is to provide a “critical mass” of types and operations, free the programmer from worrying about details and put the burden of efficiency on the language implementation. With Icon, the designer said he could write programs he didn’t have the time to write in C or C++.

There is no official Icon users’ group, but The Icon Project maintains a moderated "Icon-group" electronic mailing list.

3 Control Structures

Icon has a few control structures that are quite useful. They can be broken up into a couple categories, loops and conditional structures. Loops in icon are not to far gone from the loops that we are used to in java and c. The one most people are familiar with from other languages is the while loop. Like other languages it is formatted as "while (condition) do ...". The other loop that Icon uses is the until loop. The formatting is the same but instead of looping while the condition is still met, until loops until it reaches the condition statement. This means that technically "while not (condition) do ..." is the same as "until (condition) do ...".

The other type of control structures are if/then/else statements as well as switch cases. The format of the if/then/else goes...

if (condition) then (something) else (something)

This isn’t too different from what we are used to seeing in other languages. The "then" signifies what will happen if the condition is met for the if statement. Switches are useful for thing that may need multiple if statements. An example of this is if there are multiple values that x could be and each one does something else.
case x of
0 :write("zero")
1 :write("zero")
2 :write("zero")

4 Data Types

5 Subprograms

6 Summary

References

[1] Laurence Tratt. Experiences with an Icon-like expression evaluation system. In *Proc. 6th symposium on Dynamic languages, 2010*, pages 73-80. IEEE, 2010.

[2] Ralph E. Griswold and Madge T. Griswold. The Icon Programming Language, Third Edition. Peer-to-Peer Communications, 1996, ISBN 1-57398-001-3.