The Four Major Programming Paradigms
Topic Paper #17

Alex Laird
CS-3210-01
4/3/09
Survey of Programming Languages
Spring 2009
Computer Science, (319) 360-8771
alexdlaird@cedarville.edu

Grading Rubric		Max	Earn
	On Time/Format	1	
	Correct	5	
	Clear	2	
	Concise	2	
	Total	10 pts	

Peer Reviewer

ABSTRACT
This paper discusses the four major programming paradigms and how they have progressed from 1950 to 2009.

Keywords
Imperative, Functional, Logic, Object-Oriented, Programming, Paradigm

1. INTRODUCTION
Since programming languages were first beginning to be developed in the mid-1900s, programming paradigms have been in existence. A programming paradigm is a syntactical and structural method of writing a program. As langauges have developed and become more sophisticated, so have the paradigms that are used to write in them.
Though there are dozens of programming paradigms, the four major paradigms in existence today are imperative, functional, logic, and object-oriented programming.

2. IMPERATIVE PROGRAMMING
The earliest imperative languages were machine languages, so imperative programming was the earilest well-known programming paradigm. FORTRAN, the very first non-machine-code programming language, was developed in 1954 and was the first imperative programming language [2].

Imperative programming is the base of almost all hardware implementation, since it is straight forward and most closely resembles machine language. In imperative programming, statements are instructions at the native machine-level, and they contain states and variables that point right to the memory. Each step is an instruction, and all ideas are implemented directly on the hardware [2].

3. FUNCTIONAL PROGRAMMING
In the late 1950s, LISP was developed, a programming language generally regarded as the first programming language to contain functional concepts.

Funcational languages contain concepts that, on a pure level, are foreign to other paradigms, but all of the most useful languages use more than one programming paradigm, so the existence of functional concepts is in numerous programming languages. The main ideas of functional programmign are high-order and pure functions, recursion, evaluation of equations, and pattern matching, just to name a few [3].

Alex Laird, Cedarville University, Cedarville, Ohio, 45314

4. LOGIC PROGRAMMING
Logic programming is one that is more used *in* other paradigms than having purely logic-based languages. For instance, Prolog is one of the very few purely logical programming languages.

In logic programming, each step of the program is processed by analyzing a set of facts or rules, most commonly referred to as clauses. Storage is manipulated through the conditions held within the clauses. Prolog is the most widely known logic programming language [4].

5. OBJECT-ORIENTED PROGRAMMING
Object-oriented programming is the most recent paradigm to come into existence, and is the current giant among programmign experts. All the top programming languages support object-oriented design. Object-oriented programming is significant because it borrows concepts from the other top programming paradigms, making it one of the most versatile.

6. CONCLUSIONS
All four of the main programming paradigms are useful in their own way, but pure programmg languages of only one paradigm are known to be slightly more limiting. Object-oriented design is currently the most versatile and widely used programming paradigm.

7. REFERENCES
[1] Sebesta, Robert W., 2008. Concepts of Programming Languages. Addison-Wesley, Boston. ISBN: 978-0-321-49362-0

[2] Reddy, U. S. 1996. Imperative functional programming. *ACM Comput. Surv.* 28, 2 (Jun. 1996), 312-314. DOI= http://doi.acm.org/10.1145/234528.234736

[3] Hudak, P. 1989. Conception, evolution, and application of functional programming languages. *ACM Comput. Surv.* 21, 3 (Sep. 1989), 359-411. DOI= http://doi.acm.org/10.1145/72551.72554

[4] Dantsin, E., Eiter, T., Gottlob, G., and Voronkov, A. 2001. Complexity and expressive power of logic programming. *ACM Comput. Surv.* 33, 3 (Sep. 2001), 374-425. DOI= http://doi.acm.org/10.1145/502807.502810

[5] Müller, B. 1993. Is object-oriented programming structured programming?. *SIGPLAN Not.* 28, 9 (Sep. 1993), 57-66. DOI= http://doi.acm.org/10.1145/165364.165385