

# 쉬운 전문용어

수집·제안·편집 이 광근  
서울대학교 컴퓨터공학부  
홈페이지: kwangkeunyi.snu.ac.kr

## 배경

쉽게 사그라들지 모른다. 전문지식이 전문 학자들에게만 머문다면 그 분야는 그렇게 될 것이다. 저변이 좁아지고, 이는 깊은 공부에 필요한 뒷심 부족으로 연결되고, 깊은 공부를 달성하는 인구는 그만큼 쪼그라들기 때문이다.

반대로 온 전문지식이 보다 많은 사람들에게 널리 퍼진다면, 그래서 더욱 발전할 힘을 얻는 활기찬 선순환이 만들어진다면. 이렇게 되면 그 분야를 밀어올리는 힘은 나날이 커질 것이다. 전문지식이 모두의 상식이 되고 모두의 지력은 그만큼 강해질 것이다. 그래서 모두는 쉽게 더 깊은 수준의 지력을 만드는데 공헌할 수 있고 전문가는 궁리해낸 지식을 널리 다양하게 점점받을 수 있다.

이를 위해선 쉬운 전문용어가 필요하다. 전문개념의 핵심을 쉽게 전달하도록, 최대한 쉬운 일상어로 쓰는 전문용어. 학술은 학술의 언어를 -우리로서는 소리로만 읽을 원어나 한문을- 사용해야만 정확하고 정밀하고 경제적인 걸까? 그렇지 않다. 아무리 정교한 전문지식이라도 쉬운 일상어로 짧고 정밀하게 전달될 수 있다. 시에서 평범한 언어로 밀도 있게 전달되는 정밀한 느낌을 겪으며 짐작되는 바이다.

든든한 토대가 이것이다. 활발한 실험이 진행되는 쉬운 전문용어의 생태계. 이것이 전세계가 울타리없이 경쟁하는 소용돌이에서 우리가 선두로 치고나갈 도타운 바탕이다.

## 원칙

쉬운 전문용어를 만들때의 원칙은 다음과 같다.

- 전문용어의 의미를 정확히 이해하도록 한다.
- 그 의미가 정확히 전달되는 쉬운말을 찾는다.
- 이때, 어깨에 힘을 뺀다. 지레 겁먹게하는 용어(불필요한 한문)을 피하고, 가능하면 쉬운말을 찾는다.
- 전문용어 하나에 쉬운 한글용어 하나가 일대일 대응일 필요가 없이, 상황에 따라서 다양하게 풀어쓸 수 있다. 중요한 것은 의미의 명확한 전개.
- 영문 전문용어는 해당 우리말 다음에 괄호안에 항상 따라붙인다.
- 도저히 우리말을 찾을 수 없을 땐, 소리나는대로 쓰고 괄호안에 따라붙인다.
- 기존의 권위에 얽매이지 않는다. 기존 용어사전이나 이미 널리퍼진 용어지만 쉽지않은 전문용어라면, 보다 쉬운 전문용어를 찾고 그렇게 쓴다. 우리 분야의 전문가인 우리가 주도한다.
- 쉬운말은 순수 우리말을 뜻하지 않는다. 널리 퍼진 외래어라도 쉽다면 문제되지않는다.

## 전문용어

abduction

애펙, 2

원인 짐작하기, 2

abstract interpretation

요약해석, 2

abstract semantics

요약 의미, 2

요약된 의미구조, 2

abstract syntax

핵심 문법구조, 2

abstract type

구현된 속사정이 감추어진 타입, 2

속내용이 감추어진 타입, 2

추상적인 타입, 2

abstraction

속내용 감추기, 2

요약, 2

핵심 드러내기, 2

abstraction hierarchy

속내용 감추며 차곡차곡 쌓기, 2

allocation

메모리 할당, 2

application

호출, 2

applicative language

값 중심의 언어, 2

assignment

값 기록하기, 2

기록하기, 2

메모리에 쓰기, 2

associativity

결합법칙, 2

방향성, 2

attribute grammar

속성 문법, 2

할일이 팔려있는 문법, 2

axiomatic thoery

엄밀한 논리 시스템, 2

- binary
  - 두개의, 2
- bind
  - 묶다, 2
  - 이름짓다, 2
  - 정의하다, 2
- binding
  - 명명하기, 2
  - 이름짓기, 2
  - 정의하기, 2
- Boolean expression
  - 부울식, 2
- bottom
  - 바닥, 2
- bounded probabilistic polynomial
  - 오류율을 잡아둘 수 있는 확률형 다항, 2
- built-in
  - 불박이, 2
  - 이미 있는, 2
- calculus
  - 계산법, 2
  - 셈법, 2
- call by name
  - 식전달 호출, 2
- call by reference
  - 주소전달 호출, 2
- call by value
  - 값전달 호출, 2
- Cartesian product
  - 데카르트 곱, 2
  - 완전곱, 2
- case expression
  - 선택식, 2
- closure conversion
  - 함수 변환, 2
  - 함수가 인자를 통해서만 외부와 소통하게 하는 변환, 2
  - 함수의 자유변수를 없애주는 변환, 2
- compilation
  - 언어의 기계어 변환, 2

- 프로그램 번역, 2
- compilation unit
  - 번역 단위, 2
  - 컴파일 단위, 2
- complete
  - 빠뜨리게 없는, 2
  - 빠뜨림이 없는, 2
  - 완전한, 2
- complete partial-order set
  - 완전한 부분순서 집합, 2
- completeness
  - 빠뜨림없음, 2
  - 완전성, 2
  - 완전함, 2
- computation
  - 계산, 2
- computation strategy
  - 계산 방식, 2
  - 계산 전략, 2
- computational complexity
  - 계산 복잡도, 2
- computational learning theory
  - 계산 학습 이론, 2
- concrete syntax
  - 구체적 문법 구조, 2
- conjunction
  - 그리고-식, 2
- conjunctive normal form
  - 그리고-조합 바른 식, 2
  - 그리고-조합 표준형, 2
- consistency
  - 일관성, 2
- constant
  - 상수, 2
- constraint
  - 제약, 2
- constraint expression
  - 제약식, 2
- constructor
  - 데이타 구성자, 2

- constructor bind
  - 데이터 구성자 정의, 2
- constructor description
  - 데이터 구성자 접속방안, 2
- context
  - 문맥, 2
  - 환경, 2
- continuation
  - 앞으로 할 계산, 2
  - 앞으로 할 일, 2
- continuation passing style
  - 계산과정을 전달하는, 2
  - 앞으로 할 일을 전달하는, 2
  - 앞으로 할 일을 함수로 정리해서 전달하는, 2
- continuation passing style transform
  - 앞으로 할 일 전달 변환, 2
- control flow analysis
  - 실행 흐름 분석, 2
  - 함수 흐름 분석, 2
- control structure
  - 실행 흐름, 2
- convergent
  - 수렴하는, 2
- convex programming
  - 볼록 프로그래밍, 2
- correctness
  - 맞음, 2
  - 올바름, 2
- curried application
  - 커리형 함수 적용, 2
  - 함수를 야금야금 적용하기, 2
- curried function
  - 야금야금 함수, 2
  - 여러인자를 야금야금 받을 수 있는 함수, 2
  - 커리형 함수, 2
- dangling pointer
  - 대상이 사라진 포인터, 2
  - 오리알 포인터, 2
  - 오염된 메모리, 2

- 재활용된 메모리, 2
- 헛 포인터, 2
- data constructor
  - 데이타 구성자, 2
  - 자료 구성자, 2
- data description
  - 데이타 타입 접속방안, 2
- data structure
  - 데이타 구조, 2
  - 자료 구조, 2
- de-sugar
  - 설탕 구조를 풀다, 2
  - 설탕구조를 녹이다, 2
- dead code
  - 무용지물 코드, 2
  - 쓸데없는 코드, 2
- decision problem
  - 예-아니오 문제, 2
- decision procedure
  - 예-아니오 자동 판정기, 2
  - 예-아니오 판정 알고리즘, 2
  - 예-아니오 판정 프로그램, 2
- declaration
  - 선언, 2
- deduction
  - 디덕, 2
  - 반드시 이끌기, 2
- deep neural net
  - 깊은 신경망, 2
  - 딥뉴럴넷, 2
- delayed evaluation
  - 최대한 미루어 계산하기, 2
  - 최대한 미루어 계산하는 방법, 2
- denotational semantics
  - 고정점 방식의 의미구조, 2
  - 궁극의 의미하는 바를 표현하는 의미구조, 2
  - 조립식 의미구조, 2
- destructive
  - 메모리값을 변동시키는, 2
  - 저장값을 변동시키는, 2

- deterministic
  - 모든게 정해진, 2
  - 한가지로 정하는, 2
  - 확실한 연산만 있는, 2
- digit
  - 숫자, 2
- disjunction
  - 또는-식, 2
- disjunctive normal form
  - 또는-조합 바른 식, 2
  - 또는-조합 표준형, 2
- dynamic scoping
  - 실행중에 드러나는 이름의 실체, 2
  - 이름의 유효범위가 실행 중에 결정되는, 2
- dynamic semantics
  - 동적 의미구조, 2
  - 프로그램의 실행, 2
  - 프로그램의 실행 의미구조, 2
- eager evaluation
  - 적극적인 계산법, 2
- environment
  - 이름의 실체를 보여주는 목록, 2
  - 이름표 목록, 2
  - 환경, 2
- environment enrichment
  - 기획 환경의 적응, 2
- environment function
  - 환경 함수, 2
- environment unroll
  - 실행환경 펼치기, 2
- equational reasoning
  - 같은것들로 따져가기, 2
  - 같은것을 따지기, 2
- error
  - 오류, 2
- evaluation
  - 값계산, 2
  - 계산, 2
  - 실행, 2



- evaluation by value
  - 적극적인 계산법, 2
- evaluation strategy
  - 계산방식, 2
  - 계산법, 2
- exception
  - 예외상황, 2
- exception bind
  - 예외상황 정의, 2
- exception description
  - 예외상황 접속방안, 2
- expression
  - 계산식, 2
  - 식, 2
  - 프로그램식, 2
- first-order equational logic
  - 단순 등식 논리, 2
- foreign language interface
  - 다른 언어로 짜여진 프로그램과 연결하는 방법, 2
  - 외부 언어와 연결하는 방법, 2
- free identifier
  - 무이지 않은 이름, 2
  - 실체가 없는 이름, 2
  - 자유로운 이름, 2
- free type name
  - 무이지 않은 타입 이름, 2
- free variable
  - 무이지 않은 변수, 2
  - 자유로운 변수, 2
  - 자유로운 이름, 2
  - 자유변수, 2
- function
  - 함수, 2
- function abstraction
  - 함수, 2
  - 함수로 만들기, 2
  - 함수로 속내용 감추기, 2
- function application
  - 계산, 2

- 함수 사용, 2
- 함수 사용하기, 2
- 함수 호출, 2
- function argument
  - 함수의 인자, 2
- function expression
  - 함수식, 2
- functional
  - 함수, 2
- functional language
  - 값 중심의 언어, 2
  - 함수 중심의 언어, 2
  - 함수형 언어, 2
- functional programming
  - 값 중심의 프로그래밍, 2
  - 함수형 프로그래밍, 2
- functional style
  - 값 중심 스타일, 2
  - 함수 중심 스타일, 2
  - 함수형 스타일, 2
- functor
  - 모듈 만드는 함수, 2
  - 모듈함수, 2
- functor signature instantiation
  - 모듈함수 타입의 실현, 2
- fuzzing
  - 마구잡이 sw깨기, 2
  - 마구잡이 깨기, 2
- garbage collection
  - 메모리 재활용, 2
- grammar
  - 문법, 2
- halting problem
  - 멈춤문제, 2
- heap profiler
  - 메모리 계측기, 2
- hierarchy
  - 계층구조, 2

- 계층구조 형성하기, 2
- high-order function
  - 고차 함수, 2
  - 함수를 주고 받는 함수, 2
- homomorphic
  - 동형의, 2
  - 생긴구조가 같은, 2
- identifier
  - 이름, 2
- identity function
  - 일없는 함수, 2
- imperative language
  - 기계중심의 언어, 2
  - 메모리 중심의 언어, 2
  - 명령형 언어, 2
  - 행동지침형 언어, 2
- incomplete
  - 불완전한, 2
  - 빠뜨리게 있는, 2
  - 완전하지않은, 2
- incompleteness theorem
  - 불완전성 정리, 2
- induction
  - 인덕, 2
  - 짐작해서 이끌기, 2
- infix
  - 사이끼기, 2
- insertion sort
  - 삽입 정렬, 2
- interface
  - 사용법, 2
  - 접속 방안, 2
  - 접속 형태, 2
- interpretation
  - 실행, 2
- interpreter
  - 실행기, 2
- invariant
  - 불변성질, 2

- isomorphic
  - 똑같은, 2
- iteration
  - 반복, 2
- lattice
  - 래티스, 2
- lazy evaluation
  - 값 계산을 최대한 미루는, 2
  - 소극적 계산법, 2
  - 제때 계산법, 2
  - 지연 계산법, 2
  - 필요한 때만 값을 계산하는, 2
- leaf
  - 말단노드, 2
- lexical conventions
  - 어휘 만드는 방법, 2
- lexical scope
  - 생김새로 결정되는 유효 범위, 2
- lexicographic order
  - 사전적 순서, 2
- linear function
  - 직선 함수, 2
- list
  - 리스트, 2
- local definition
  - 그동네 정의, 2
  - 우물안 정의, 2
- logical relation
  - 논리 관계, 2
- machine learning
  - 기계 학습, 2
- match
  - 어울리기, 2
  - 패턴에 맞추기, 2
- memory leak
  - 메모리 누수, 2
  - 메모리 출혈, 2
  - 재활용 놓치는 메모리, 2

- metalanguage
  - 언어를 설명하는 언어, 2
- model checking
  - 맞나 확인, 2
  - 모델 검증, 2
  - 모델 체킹, 2
- module
  - 모듈, 2
- mono-variant analysis
  - 다대일 분석, 2
  - 다수의 프로그램 흐름을 하나로 요약하는 분석, 2
- mutual recursive
  - 서로 맞물려서 호출하는, 2
  - 서로 호출하는, 2
- negation
  - 뒤집기, 2
- network
  - 네트워크, 2
- node
  - 노드, 2
- non-deterministic
  - 모든가지를 한꺼번에 다하는, 2
  - 운에 기대는, 2
  - 한가지로 정하지 않는, 2
- non-deterministic polynomial
  - 모든가지를 한꺼번에 다할때 다항시간에 풀리는, 2
  - 운에 기대면 다항시간 안에 풀리는, 2
- non-expansive
  - 새 메모리를 소모하지않는, 2
- normal form
  - 바른꼴, 2
  - 표준형, 2
- object
  - 물건, 2
- object-oriented language
  - 물건 중심의 언어, 2
- operational semantics
  - 계산과정을 드러내는 의미구조, 2

- 실행과정을 드러내는 의미구조, 2
- operator
  - 연산자, 2
- or-pattern
  - 무더기 패턴, 2
- ordered relation
  - 순서 관계, 2
- overflow
  - 넘침, 2
- parameter
  - 인자, 2
- parameterized module
  - 일반화된 모듈, 2
- parity function
  - 홀짝 함수, 2
- partial function
  - 일부만 정의된 함수, 2
- pattern
  - 패턴, 2
- pattern match
  - 패턴에 대보기, 2
  - 패턴에 맞추기, 2
- pattern row
  - 레코드 패턴, 2
- poly-variant analysis
  - 다대다 분석, 2
  - 다수의 프로그램 흐름을 하나이상으로 요약하는 분석, 2
  - 다형성을 가지는 분석, 2
- polymorphic
  - 다형의, 2
  - 모양이 다양한, 2
  - 여러 모양의, 2
  - 여러 타입을 가지는, 2
- polymorphic function
  - 다형 함수, 2
  - 인자 타입에 상관없는 함수, 2
- polymorphism
  - 다형성, 2
- postfix

- 뒤에 붙는, 2
- precedence
  - 우선순위, 2
- predicate
  - 서술식, 2
  - 조건자, 2
- predicate abstraction
  - 조건식 요약, 2
  - 조건식을 하나의 변수로 요약하기, 2
- predicate logic
  - 모든-어떤 논리, 2
  - 술어 논리, 2
- prefix
  - 앞에 붙는, 2
- primitive
  - 기본, 2
- primitive recursive function
  - 단순한 재귀 함수, 2
  - 원시적인 재귀 함수, 2
- principal type
  - 가장 일반적인 타입, 2
  - 대표 타입, 2
- Probably Approximately Correct, PAC
  - 얼추거의맞기, 2
- programming language
  - 프로그래밍 언어, 2
- ramdomization
  - 무작위, 2
- ramdomized algorithm
  - 무작위 알고리즘, 2
- reasoning
  - 이치따지기, 2
- record
  - 레코드, 2
- recursive
  - 자기자신을 부르는, 2
  - 자기호출, 2
- recursive function
  - 자기자신을 부르는 함수, 2

- 자기호출함수, 2
- 재귀함수, 2
- recursive primitive definition
  - 원시적 자기참조 정의, 2
- reduction
  - 계산, 2
  - 수행, 2
  - 줄이기, 2
- reference
  - 메모리 주소, 2
- reference manual
  - 참고서, 2
- rewrite
  - 다시 쓰기, 2
- rewrite rule
  - 다시쓰기 규칙, 2
- rewrite semantics
  - 다시쓰기로 정의한 의미구조, 2
- scaffolding code
  - 테스터 코드, 2
  - 테스트 발판 코드, 2
- scheme
  - 틀, 2
- scope
  - 유효범위, 2
- semantics
  - 뜻, 2
  - 속내용, 2
  - 의미, 2
  - 의미구조, 2
- separated sum
  - 출신기억 합집합, 2
  - 출신을 기억하는 합집합, 2
- sequence
  - 나열식, 2
- side-effect
  - 따라 일어나는 일, 2
  - 메모리 반응, 2
  - 수반되는 반응, 2



- 함께오는 반응, 2
- signature
  - 모듈타입, 2
- signature bind
  - 모듈타입 정의, 2
- signature instantiation
  - 모듈 타입의 실현, 2
- signature matching
  - 모듈타입에 대보기, 2
  - 모듈타입에 맞추기, 2
- simple type
  - 단순 타입, 2
- skolemization
  - 안전하게 정량자 제거하기, 2
  - 안전한 정량자 제거, 2
- soundness
  - 믿을만함, 2
  - 안전성, 2
  - 안전함, 2
  - 올바름, 2
- sparse data structure
  - 듬성듬성한 데이터 구조, 2
- specification
  - 명세, 2
- static analysis
  - 정적 프로그램 분석, 2
  - 정적분석, 2
- static scope
  - 미리 결정된 이름의 유효 범위, 2
  - 정적인 유효 범위, 2
- static scoping
  - 실행전에 결정되는 이름의 실체, 2
  - 이름의 유효범위가 미리 결정된, 2
- static semantics
  - 정적 의미구조, 2
  - 프로그램의 기획, 2
  - 프로그램의 타입 의미구조, 2
- strict evaluation
  - 일단 값을 계산하고 보는, 2
  - 적극적 계산법, 2

- string
  - 글자실, 2
- structure
  - 모듈, 2
- structure bind
  - 모듈 정의, 2
- structure description
  - 모듈 접속방안, 2
- structure expression
  - 모듈식, 2
- substitution
  - 바꿔치기, 2
- symbol
  - 심벌, 2
- syntactic constraint
  - 문법적인 제약, 2
- syntactic sugar
  - 설탕구조, 2
- syntax
  - 겉모양, 2
  - 문법, 2
  - 문법구조, 2
  - 생김새, 2
- syntax analysis
  - 문법 구조 분석, 2
- tail recursive
  - 끝 재귀호출, 2
  - 마지막에 자기자신을 부르는, 2
  - 자기 호출이 마지막인, 2
- template
  - 거푸집, 2
- term
  - 식, 2
- top declaration
  - 가장 위의 선언, 2
- top-level declaration
  - 가장 위의 선언, 2
- total function
  - 완전히 정의된 함수, 2

- tree
  - 가지구조, 2
  - 나무구조, 2
- tuple
  - 짜, 2
- type
  - 타입, 2
- type abbreviation
  - 타입 줄임말, 2
- type bind
  - 타입 정의, 2
- type construct
  - 타입식, 2
- type constructor
  - 타입 구성자, 2
- type description
  - 타입 접속방안, 2
- type expression
  - 타입식, 2
- type inference
  - 타입 유추, 2
- type instantiation
  - 타입 틀 구체화, 2
  - 타입 틀 적용, 2
- type realization
  - 타입 틀 실현, 2
- type scheme
  - 타입 틀, 2
- type scheme generalization
  - 타입 틀 만들기, 2
  - 타입 틀로 일반화하기, 2
- type scheme instantiation
  - 타입 틀 유추, 2
- type structure
  - 타입 구조, 2
- type variable
  - 타입 변수, 2
- typing rule
  - 타입결정하는 규칙, 2
  - 타입유추하는 규칙, 2

- unary
  - 인자가 하나인, 2
- uncurrying
  - 야금야금 함수를 단번 함수로, 2
  - 언커링, 2
- undecidable
  - 컴퓨터로는 불가능한, 2
  - 컴퓨터로는 할 수 없는, 2
- unification
  - 같이 만들기, 2
  - 동일화, 2
- universal machine
  - 보편만능 기계, 2
- value
  - 값, 2
- value bind
  - 값 정의, 2
- variable
  - 변수, 2
- well-formed
  - 제대로 생긴, 2
- well-founded
  - 바닥이 갖추어진, 2
  - 바닥이 있는, 2
  - 올바르게 기초한, 2
- wild pattern
  - 임의 패턴, 2
- 가장 위의 선언
  - top declaration, 2
  - top-level declaration, 2
- 가장 일반적인 타입
  - principal type, 2
- 가지구조
  - tree, 2
- 값
  - value, 2
- 값 계산을 최대한 미루는
  - lazy evaluation, 2

- 값 기록하기
  - assignment, 2
- 값 정의
  - value bind, 2
- 값 중심 스타일
  - functional style, 2
- 값 중심의 언어
  - applicative language, 2
  - functional language, 2
- 값 중심의 프로그래밍
  - functional programming, 2
- 값계산
  - evaluation, 2
- 값전달 호출
  - call by value, 2
- 같이 만들기
  - unification, 2
- 같은것들로 따져가기
  - equational reasoning, 2
- 같은것을 따지기
  - equational reasoning, 2
- 거푸집
  - template, 2
- 겉모양
  - syntax, 2
- 결합법칙
  - associativity, 2
- 계산
  - computation, 2
  - evaluation, 2
  - function application, 2
  - reduction, 2
- 계산 방식
  - computation strategy, 2
- 계산 복잡도
  - computational complexity, 2
- 계산 전략
  - computation strategy, 2
- 계산 학습 이론
  - computational learning theory, 2

- 계산과정을 드러내는 의미구조
  - operational semantics, 2
- 계산과정을 전달하는
  - continuation passing style, 2
- 계산방식
  - evaluation strategy, 2
- 계산법
  - calculus, 2
  - evaluation strategy, 2
- 계산식
  - expression, 2
- 계층구조
  - hierarchy, 2
- 계층구조 형성하기
  - hierarchy, 2
- 고정점 방식의 의미구조
  - denotational semantics, 2
- 고차 함수
  - high-order function, 2
- 구체적 문법 구조
  - concrete syntax, 2
- 구현된 속사정이 감추어진 타입
  - abstract type, 2
- 공극의 의미하는 바를 표현하는 의미구조
  - denotational semantics, 2
- 그동네 정의
  - local definition, 2
- 그리고-식
  - conjunction, 2
- 그리고-조합 바른 식
  - conjunctive normal form, 2
- 그리고-조합 표준형
  - conjunctive normal form, 2
- 글자열
  - string, 2
- 기계 학습
  - machine learning, 2
- 기계중심의 언어
  - imperative language, 2
- 기록하기

- assignment, 2
- 기본
  - primitive, 2
- 기획 환경의 적응
  - environment enrichment, 2
- 깊은 신경망
  - deep neural net, 2
- 끝 재귀호출
  - tail recursive, 2
- 나무구조
  - tree, 2
- 나열식
  - sequence, 2
- 넘침
  - overflow, 2
- 네트워크
  - network, 2
- 노드
  - node, 2
- 논리 관계
  - logical relation, 2
- 다대다 분석
  - poly-variant analysis, 2
- 다대일 분석
  - mono-variant analysis, 2
- 다른 언어로 짜여진 프로그램과 연결하는 방법
  - foreign language interface, 2
- 다수의 프로그램 흐름을 하나로 요약하는 분석
  - mono-variant analysis, 2
- 다수의 프로그램 흐름을 하나이상으로 요약하는 분석
  - poly-variant analysis, 2
- 다시 쓰기
  - rewrite, 2
- 다시쓰기 규칙
  - rewrite rule, 2
- 다시쓰기로 정의한 의미구조
  - rewrite semantics, 2
- 다형 함수
  - polymorphic function, 2

- 다형성
  - polymorphism, 2
- 다형성을 가지는 분석
  - poly-variant analysis, 2
- 다형의
  - polymorphic, 2
- 단순 등식 논리
  - first-order equational logic, 2
- 단순 타입
  - simple type, 2
- 단순한 재귀 함수
  - primitive recursive function, 2
- 대상이 사라진 포인터
  - dangling pointer, 2
- 대표 타입
  - principal type, 2
- 데이타 구성자
  - constructor, 2
  - data constructor, 2
- 데이타 구성자 접속방안
  - constructor description, 2
- 데이타 구성자 정의
  - constructor bind, 2
- 데이타 구조
  - data structure, 2
- 데이타 타입 접속방안
  - data description, 2
- 데카르트 곱
  - Cartesian product, 2
- 동일화
  - unification, 2
- 동적 의미구조
  - dynamic semantics, 2
- 동형의
  - homomorphic, 2
- 두개의
  - binary, 2
- 뒤에 붙는
  - postfix, 2
- 뒤집기



- negation, 2
- 듬성듬성한 데이터 구조
  - sparse data structure, 2
- 디덕
  - deduction, 2
- 딥뉴럴넷
  - deep neural net, 2
- 따라 일어나는 일
  - side-effect, 2
- 또는-식
  - disjunction, 2
- 또는-조합 바른 식
  - disjunctive normal form, 2
- 또는-조합 표준형
  - disjunctive normal form, 2
- 똑같은
  - isomorphic, 2
- 뜻
  - semantics, 2
- 래티스
  - lattice, 2
- 레코드
  - record, 2
- 레코드 패턴
  - pattern row, 2
- 리스트
  - list, 2
- 마구잡이 sw깨기
  - fuzzing, 2
- 마구잡이 깨기
  - fuzzing, 2
- 마지막에 자기자신을 부르는
  - tail recursive, 2
- 말단노드
  - leaf, 2
- 맞나 확인
  - model checking, 2
- 맞음
  - correctness, 2
- 멈춤문제

- halting problem, 2
- 메모리 계측기
  - heap profiler, 2
- 메모리 누수
  - memory leak, 2
- 메모리 반응
  - side-effect, 2
- 메모리 재활용
  - garbage collection, 2
- 메모리 주소
  - reference, 2
- 메모리 중심의 언어
  - imperative language, 2
- 메모리 출혈
  - memory leak, 2
- 메모리 할당
  - allocation, 2
- 메모리값을 변동시키는
  - destructive, 2
- 메모리에 쓰기
  - assignment, 2
- 명령형 언어
  - imperative language, 2
- 명명하기
  - binding, 2
- 명세
  - specification, 2
- 모델 검증
  - model checking, 2
- 모델 체크
  - model checking, 2
- 모듈
  - module, 2
  - structure, 2
- 모듈 만드는 함수
  - functor, 2
- 모듈 접속방안
  - structure description, 2
- 모듈 정의
  - structure bind, 2

- 모듈 타입의 실현
  - signature instantiation, 2
- 모듈식
  - structure expression, 2
- 모듈타입
  - signature, 2
- 모듈타입 정의
  - signature bind, 2
- 모듈타입에 대보기
  - signature matching, 2
- 모듈타입에 맞추기
  - signature matching, 2
- 모듈함수
  - functor, 2
- 모듈함수 타입의 실현
  - functor signature instantiation, 2
- 모든-어떤 논리
  - predicate logic, 2
- 모든가지를 한꺼번에 다하는
  - non-deterministic, 2
- 모든가지를 한꺼번에 다할때 다항시간에 풀리는
  - non-deterministic polynomial, 2
- 모든게 정해진
  - deterministic, 2
- 모양이 다양한
  - polymorphic, 2
- 무더기 패턴
  - or-pattern, 2
- 무용지물 코드
  - dead code, 2
- 무작위
  - randomization, 2
- 무작위 알고리즘
  - randomized algorithm, 2
- 묶다
  - bind, 2
- 묶이지 않은 변수
  - free variable, 2
- 묶이지 않은 이름
  - free identifier, 2

- 묵이지 않은 타입 이름
  - free type name, 2
- 문맥
  - context, 2
- 문법
  - grammar, 2
  - syntax, 2
- 문법 구조 분석
  - syntax analysis, 2
- 문법구조
  - syntax, 2
- 문법적인 제약
  - syntactic constraint, 2
- 물건
  - object, 2
- 물건 중심의 언어
  - object-oriented language, 2
- 미리 결정된 이름의 유효 범위
  - static scope, 2
- 믿을만함
  - soundness, 2
- 바꿔치기
  - substitution, 2
- 바닥
  - bottom, 2
- 바닥이 갖추어진
  - well-founded, 2
- 바닥이 있는
  - well-founded, 2
- 바른꼴
  - normal form, 2
- 반드시 이끌기
  - deduction, 2
- 반복
  - iteration, 2
- 방향성
  - associativity, 2
- 번역 단위
  - compilation unit, 2
- 변수

- variable, 2
- 보편만능 기계
  - universal machine, 2
- 볼록 프로그래밍
  - convex programming, 2
- 부울식
  - Boolean expression, 2
- 불변성질
  - invariant, 2
- 불완전성 정리
  - incompleteness theorem, 2
- 불완전한
  - incomplete, 2
- 불박이
  - built-in, 2
- 빠뜨리게 없는
  - complete, 2
- 빠뜨리게 있는
  - incomplete, 2
- 빠뜨림없음
  - completeness, 2
- 빠뜨림이 없는
  - complete, 2
- 사용법
  - interface, 2
- 사이끼기
  - infix, 2
- 사전적 순서
  - lexicographic order, 2
- 삽입 정렬
  - insertion sort, 2
- 상수
  - constant, 2
- 새 메모리를 소모하지않는
  - non-expansive, 2
- 생긴구조가 같은
  - homomorphic, 2
- 생김새
  - syntax, 2

생김새로 결정되는 유효 범위  
     lexical scope, 2  
 서로 맞물려서 호출하는  
     mutual recursive, 2  
 서로 호출하는  
     mutual recursive, 2  
 서술식  
     predicate, 2  
 선언  
     declaration, 2  
 선택식  
     case expression, 2  
 설탕 구조를 풀다  
     de-sugar, 2  
 설탕구조  
     syntactic sugar, 2  
 설탕구조를 녹이다  
     de-sugar, 2  
 셈법  
     calculus, 2  
 소극적 계산법  
     lazy evaluation, 2  
 속내용  
     semantics, 2  
 속내용 감추기  
     abstraction, 2  
 속내용 감추며 차곡차곡 쌓기  
     abstraction hierarchy, 2  
 속내용이 감추어진 타입  
     abstract type, 2  
 속성 문법  
     attribute grammar, 2  
 수렴하는  
     convergent, 2  
 수반되는 반응  
     side-effect, 2  
 수행  
     reduction, 2  
 순서 관계  
     ordered relation, 2

- 술어 논리
  - predicate logic, 2
- 숫자
  - digit, 2
- 식
  - expression, 2
  - term, 2
- 식전달 호출
  - call by name, 2
- 실체가 없는 이름
  - free identifier, 2
- 실행
  - evaluation, 2
  - interpretation, 2
- 실행 흐름
  - control structure, 2
- 실행 흐름 분석
  - control flow analysis, 2
- 실행과정을 드러내는 의미구조
  - operational semantics, 2
- 실행기
  - interpreter, 2
- 실행전에 결정되는 이름의 실체
  - static scoping, 2
- 실행중에 드러나는 이름의 실체
  - dynamic scoping, 2
- 실행환경 펼치기
  - environment unroll, 2
- 심벌
  - symbol, 2
- 쓸데없는 코드
  - dead code, 2
- 안전성
  - soundness, 2
- 안전하게 정량자 제거하기
  - skolemization, 2
- 안전한 정량자 제거
  - skolemization, 2
- 안전함
  - soundness, 2

- 앞에 붙는
  - prefix, 2
- 앞으로 할 계산
  - continuation, 2
- 앞으로 할 일
  - continuation, 2
- 앞으로 할 일 전달 변환
  - continuation passing style transform, 2
- 앞으로 할 일을 전달하는
  - continuation passing style, 2
- 앞으로 할 일을 함수로 정리해서 전달하는
  - continuation passing style, 2
- 앱덕
  - abduction, 2
- 야금야금 함수
  - curried function, 2
- 야금야금 함수를 단번 함수로
  - uncurrying, 2
- 어울리기
  - match, 2
- 어휘 만드는 방법
  - lexical conventions, 2
- 언어를 설명하는 언어
  - metalanguage, 2
- 언어의 기계어 변환
  - compilation, 2
- 언커링
  - uncurrying, 2
- 얼추거의맞기
  - Probably Approximately Correct, PAC, 2
- 엄밀한 논리 시스템
  - axiomatic thoery, 2
- 여러 모양의
  - polymorphic, 2
- 여러 타입을 가지는
  - polymorphic, 2
- 여러인자를 야금야금 받을 수 있는 함수
  - curried function, 2
- 연산자
  - operator, 2



- 예-아니오 문제
  - decision problem, 2
- 예-아니오 자동 판정기
  - decision procedure, 2
- 예-아니오 판정 알고리즘
  - decision procedure, 2
- 예-아니오 판정 프로그램
  - decision procedure, 2
- 예외상황
  - exception, 2
- 예외상황 접속방안
  - exception description, 2
- 예외상황 정의
  - exception bind, 2
- 오류
  - error, 2
- 오류율을 잡아둘 수 있는 확률형 다항
  - bounded probabilistic polynomial, 2
- 오리알 포인터
  - dangling pointer, 2
- 오염된 메모리
  - dangling pointer, 2
- 올바르게 기초한
  - well-founded, 2
- 올바름
  - correctness, 2
  - soundness, 2
- 완전곱
  - Cartesian product, 2
- 완전성
  - completeness, 2
- 완전하지않은
  - incomplete, 2
- 완전한
  - complete, 2
- 완전한 부분순서 집합
  - complete partial-order set, 2
- 완전함
  - completeness, 2
- 완전히 정의된 함수

- total function, 2
- 외부 언어와 연결하는 방법
  - foreign language interface, 2
- 요약
  - abstraction, 2
- 요약 의미
  - abstract semantics, 2
- 요약된 의미구조
  - abstract semantics, 2
- 요약해석
  - abstract interpretation, 2
- 우물안 정의
  - local definition, 2
- 우선순위
  - precedence, 2
- 운에 기대는
  - non-deterministic, 2
- 운에 기대면 다항시간 안에 풀리는
  - non-deterministic polynomial, 2
- 원시적 자기참조 정의
  - recursive primitive definition, 2
- 원시적인 재귀 함수
  - primitive recursive function, 2
- 원인 짐작하기
  - abduction, 2
- 유효범위
  - scope, 2
- 의미
  - semantics, 2
- 의미구조
  - semantics, 2
- 이름
  - identifier, 2
- 이름의 실체를 보여주는 목록
  - environment, 2
- 이름의 유효범위가 미리 결정된
  - static scoping, 2
- 이름의 유효범위가 실행 중에 결정되는
  - dynamic scoping, 2
- 이름짓기

- binding, 2
- 이름짓다
  - bind, 2
- 이름표 목록
  - environment, 2
- 이미 있는
  - built-in, 2
- 이치따지기
  - reasoning, 2
- 인덕
  - induction, 2
- 인자
  - parameter, 2
- 인자 타입에 상관없는 함수
  - polymorphic function, 2
- 인자가 하나인
  - unary, 2
- 일관성
  - consistency, 2
- 일단 값을 계산하고 보는
  - strict evaluation, 2
- 일반화된 모듈
  - parameterized module, 2
- 일부만 정의된 함수
  - partial function, 2
- 일없는 함수
  - identity function, 2
- 임의 패턴
  - wild pattern, 2
- 자기 호출이 마지막인
  - tail recursive, 2
- 자기자신을 부르는
  - recursive, 2
- 자기자신을 부르는 함수
  - recursive function, 2
- 자기호출
  - recursive, 2
- 자기호출함수
  - recursive function, 2
- 자료 구성자

- data constructor, 2
- 자료 구조
  - data structure, 2
- 자유로운 변수
  - free variable, 2
- 자유로운 이름
  - free identifier, 2
  - free variable, 2
- 자유변수
  - free variable, 2
- 재귀함수
  - recursive function, 2
- 재활용 놓치는 메모리
  - memory leak, 2
- 재활용된 메모리
  - dangling pointer, 2
- 저장값을 변동시키는
  - destructive, 2
- 적극적 계산법
  - strict evaluation, 2
- 적극적인 계산법
  - eager evaluation, 2
  - evaluation by value, 2
- 접속 방안
  - interface, 2
- 접속 형태
  - interface, 2
- 정의하기
  - binding, 2
- 정의하다
  - bind, 2
- 정적 의미구조
  - static semantics, 2
- 정적 프로그램 분석
  - static analysis, 2
- 정적분석
  - static analysis, 2
- 정적인 유효 범위
  - static scope, 2
- 제대로 생긴

- well-formed, 2
- 제때 계산법
  - lazy evaluation, 2
- 제약
  - constraint, 2
- 제약식
  - constraint expression, 2
- 조건식 요약
  - predicate abstraction, 2
- 조건식을 하나의 변수로 요약하기
  - predicate abstraction, 2
- 조건자
  - predicate, 2
- 조립식 의미구조
  - denotational semantics, 2
- 주소전달 호출
  - call by reference, 2
- 줄이기
  - reduction, 2
- 지연 계산법
  - lazy evaluation, 2
- 직선 함수
  - linear function, 2
- 짐작해서 이끌기
  - induction, 2
- 짜
  - tuple, 2
- 참고서
  - reference manual, 2
- 최대한 미루어 계산하기
  - delayed evaluation, 2
- 최대한 미루어 계산하는 방법
  - delayed evaluation, 2
- 추상적인 타입
  - abstract type, 2
- 출신기억 합집합
  - separated sum, 2
- 출신을 기억하는 합집합
  - separated sum, 2
- 커리형 함수

- curried function, 2
- 커리형 함수 적용
  - curried application, 2
- 컴파일 단위
  - compilation unit, 2
- 컴퓨터로는 불가능한
  - undecidable, 2
- 컴퓨터로는 할 수 없는
  - undecidable, 2
- 타입
  - type, 2
- 타입 구성자
  - type constructor, 2
- 타입 구조
  - type structure, 2
- 타입 변수
  - type variable, 2
- 타입 유추
  - type inference, 2
- 타입 접속방안
  - type description, 2
- 타입 정의
  - type bind, 2
- 타입 줄임말
  - type abbreviation, 2
- 타입 틀
  - type scheme, 2
- 타입 틀 구체화
  - type instantiation, 2
- 타입 틀 만들기
  - type scheme generalization, 2
- 타입 틀 실현
  - type realization, 2
- 타입 틀 유추
  - type scheme instantiation, 2
- 타입 틀 적용
  - type instantiation, 2
- 타입 틀로 일반화하기
  - type scheme generalization, 2

- 타입결정하는 규칙
  - typing rule, 2
- 타입식
  - type construct, 2
  - type expression, 2
- 타입유추하는 규칙
  - typing rule, 2
- 테스터 코드
  - scaffolding code, 2
- 테스트 발판 코드
  - scaffolding code, 2
- 틀
  - scheme, 2
- 패턴
  - pattern, 2
- 패턴에 대보기
  - pattern match, 2
- 패턴에 맞추기
  - match, 2
  - pattern match, 2
- 표준형
  - normal form, 2
- 프로그래밍 언어
  - programming language, 2
- 프로그램 번역
  - compilation, 2
- 프로그램식
  - expression, 2
- 프로그램의 기획
  - static semantics, 2
- 프로그램의 실행
  - dynamic semantics, 2
- 프로그램의 실행 의미구조
  - dynamic semantics, 2
- 프로그램의 타입 의미구조
  - static semantics, 2
- 필요한 때만 값을 계산하는
  - lazy evaluation, 2
- 한가지로 정하는
  - deterministic, 2

- 한가지로 정하지 않는
  - non-deterministic, 2
- 할일이 달려있는 문법
  - attribute grammar, 2
- 함께오는 반응
  - side-effect, 2
- 함수
  - function, 2
  - function abstraction, 2
  - functional, 2
- 함수 변환
  - closure conversion, 2
- 함수 사용
  - function application, 2
- 함수 사용하기
  - function application, 2
- 함수 중심 스타일
  - functional style, 2
- 함수 중심의 언어
  - functional language, 2
- 함수 호출
  - function application, 2
- 함수 흐름 분석
  - control flow analysis, 2
- 함수가 인자를 통해서만 외부와 소통하게 하는 변환
  - closure conversion, 2
- 함수로 만들기
  - function abstraction, 2
- 함수로 속내용 감추기
  - function abstraction, 2
- 함수를 야금야금 적용하기
  - curried application, 2
- 함수를 주고 받는 함수
  - high-order function, 2
- 함수식
  - function expression, 2
- 함수의 인자
  - function argument, 2
- 함수의 자유변수를 없애주는 변환
  - closure conversion, 2



- 함수형 스타일
  - functional style, 2
- 함수형 언어
  - functional language, 2
- 함수형 프로그래밍
  - functional programming, 2
- 핵심 드러내기
  - abstraction, 2
- 핵심 문법구조
  - abstract syntax, 2
- 행동지침형 언어
  - imperative language, 2
- 헛 포인터
  - dangling pointer, 2
- 호출
  - application, 2
- 홀짝 함수
  - parity function, 2
- 확실한 연산만 있는
  - deterministic, 2
- 환경
  - context, 2
  - environment, 2
- 환경 함수
  - environment function, 2