Definition : Functional Programming

A computer programming paradigm that relies on fucntions modeled on mathematical functions.

The essence of functional programming is that programs are combinations of expressions. Expressions include conrete values, variables, and also functions.

Functions are expressions that are applied to an argument or input, and once applied, can be reduced or evaluated.

The word *purity* in functional programming is sometimes also used to mean what is more properly called *referential transparency*. Referential transparency means that the same function, given the same values to evaluate, will always return the same result in pure functional programming, as they do in math.

Definition : What is a function?

A function is a relation between a set of possible inputs and a set of possible outputs. The input set is known as the domain, and the set if possible outputs for the function is called the codomain. The subset of the codomain that contains possible outputs related to different inputs is know as the image.

1 The structure of lambda terms

THe lambda calculus has 3 basic components, or *lambda terms*: expressions, variables, and abstractions.

- expression : refers to a superset of all those things.
  An expression can be a variable name, an abstraction, or a combination of those.
- abstraction : a function. It is a lambda term that has a head (a lambda) and a body and is applied to an argument (an input value).

$$\lambda x.x$$

2 Alpha equivalence

Same function, fx.x=fy.y

3 Beta reduction

Definition : Beta reduction

When applying a function to an argument, we substiture the input expression for all instances of bound variables within the body of the abstraction. We also eliminate the head of abstraction, since its onlu purpiose was to bind a variable.

$$(\lambda x.x)2$$
$$2$$

This function is the *identity* function. And applications in the lambda calculus are *left associative.*

3.1 Free variables

Free variables are those in the body expression that are not named in the head.

$$\lambda x.xy$$

Alpha equivalence does not apply to free variables.

4 Multiple arguments

$$\lambda xy.xy \rightarrow \lambda x(\lambda y.xy)$$

5 Evaluation is simplification

Definition : Beta normal form

Beta normal form is when you cannot beta reduce the terms any further. This corresponds to a fully evaluated expression.

Eg: 2000/1000 and 2.

6 Combinators

Definition

A comibnator is a lambda term with no free variables. Combinators serve only to combine the arguments they are given.

7 Divergence

$$(\lambda x.xx)(\lambda x.xx)$$

8 Definition

1. A lambda abstraction is an anonymous function or lambda term. $(\lambda x.x + 1)$
2. Application is reducing lambdas, which binds the argument to whatever the lambda was applied to.

$$(\lambda x.x)1$$

3. Lambda calculus is a formal system for expressing programs in terms of abstraction and application.
4. Normal order is a common evaluation strategy in lambda calculi.

Normal order means evaluating (beta reducing) the leftmost outermost lambdas first.