Tampoco existen sentencias imperativas(al menos en un lenguaje de programación funcional puro) y todas las funciones tienen transparencia referencial, es decir, el resultado devuelto por una función sólo depende de los argumentos que se le pasan en esa llamada.

La idea de “repetición” se modela utilizando la recursividad ya que no que no existe el concepto de valor de una variable.

Con el lenguaje Haskell se pretendía unificar las características más importantes de los lenguajes funcionales, como las funciones de orden superior, la evaluación perezosa, la inferencia estática de tipos, los tipos de datos definidos por el usuario, el encaje de patrones y las listas por comprensión. Al diseñar el lenguaje se observó que no existía un tratamiento sistemático de la sobrecarga con lo cual se construyó una nueva solución conocida como las clases de tipos.

El lenguaje incorporaba, además, Entrada/Salida puramente funcional y definición de arrays por comprensión.

Todos los lenguajes funcionales están basados en el Lambda-Calculo. De hecho su influencia es evidente en lenguajes de programación funcionales como Lisp, Hope, Miranda, Haskell, ML y Scheme. El lambda cálculo provee de una sintáxis básica para definir primitivas de programación y usa el concepto de función matemática para transformar argumentos en resultados.

LAMBDA-CALCULO

Cálculo lambda: es un sistema formal diseñado para investigar la definición de función, la noción de aplicación de funciones y la recursión. Se puede considerar al cálculo lambda como el más pequeño lenguaje universal de programación. Consiste de una regla de transformación simple (substitución de variables) y un esquema simple para definir funciones.

El cálculo lambda es universal porque cualquier función computable puede ser expresada y evaluada a través de él.

Por ejemplo, si una función f es definida por la ecuación f(x)=t, donde t es algún término que contiene a x, entonces la aplicación f(u) devuelve el valor t[u/x], donde t[u/x] es el término que resulta de sustituir u en cada aparición de x en t.

Por ejemplo, si f(x)=x\*x, entonces f(3)=3\*3=9.

• Lambda Expresiones

Church propuso una forma especial (más compacta) de escribir estas funciones. En vez de decir “la función f donde f(x)=t”, él simplemente escribió \x.t.

Para el ejemplo anterior: \x.x\*x.

Un término de la forma \x.t se llama “lambda expresión”. La principal característica de lambda cálculo es su simplicidad ya que permite efectuar solo dos operaciones:

• Definir funciones de un solo argumento y con un cuerpo especifico, denotado por la siguiente terminología: x.B, en donde x determina el parámetro o argumento formal y B representa el cuerpo de la función, es decir f(x) = B.