زمانی که در مورد توابع در کاتلین صحبت شد به این اشاره کردم که باید بتونیم با توابع به صورت کلاس های شهروند مرتبه اول برخورد کنیم و یکی از ویژگی های مهمی که یک زبون رو Functional میکنه اینه که توابعش بتونن به صورت Higher-Order ظاهر شن و اگر با مفهوم Higher-Order اشنا نیستین به این معنیه که توابع بتونن توابع دیگر رو به صورت پارامتر ورودی قبول کنن یا این که یک تابع بتونه توابع دیگر رو برگردونه. کاتلین این رو و البته چیز های دیگه ای مثل عبارت های Lambda و یا توابع بی نام و غیره و غیره رو هم ساپورت میکنه و الان میخوایم ببنیم چجوری میتونیم از تموم ویژگی های یک زبان Functional استفاده کنیم.

بیاین چند خط کد بنویسیم:

fun operation**(**x: Int, y: Int,op: **(**Int,Int**)->**Int**)**: Int {  
 return op**(**x,y**)**}

یک تابع نوشتیم به نام operation که دو متغیر x و y و همچنین یک تابع دیگر رو به عنوان ورودی دریافت میکنه که البته این تابع ورودی، دو متغیر از جنس Int دریافت میکنه و یک مقدار Int برمیگردونه و در انتها هم ما مقدار x و y رو به تابع op دادیم و مقدارش رو برمیگردونیم.به همین سادگی در کاتلین یک تابع High-Order نوشتیم.

خب ما چطور میتونیم از این استفاده کنیم؟ مسلما باید یک تابع بنوسیم که با الگو (Int,Int) -> Int جور دربیاد، به عنوان مثال تابع زیر رو درنظر بگیرین

fun sum**(**x: Int, y: Int**)** = x + y

توی کاتلین برای صدا زدن به وسیله نام اون تابع باید از "::" استفاده کنیم

fun operation**(**x: Int, y: Int,op: **(**Int,Int**)->**Int**)**: Int {  
 return op**(**x,y**)**}

fun sum**(**x: Int, y: Int**)** = x + y

fun main**(**args: Array<String>**)** {  
 println**(**operation**(**2,3,::sum**))**}

شما توی کاتلین هرجور که دوست دارین میتونین توابع High-Order بنویسین:

fun operation**(**x:Int, op: **(**Int**)->** Unit**)**{  
  
}  
  
fun route**(**path: String,vararg actions: **(**String,String**)->** String**)**{  
   
}

و هیچ گونه محدودیتی در استفاده از توابع High-Order نداریم.