# ***INDEX :***

# ***Class And Object :***

```
class Gadgets {

    var model : String = "Samsung s20"

    var price : Float = 1199999.50F


    fun disp(){

        println("Model name is :  $model")

        println("Price of the device is :  $price")

    }

}


fun main() {

    val obj = Gadgets()       // -> Creating object of the class.

    obj.disp()                // -> Calling function member using object.


}
```

# ***Output :-***

```
PS D:\15. Tutorial Of Kotlin> cd "d:\15.
AndObject.jar }
Model name is :  Samsung s20
Price of the device is :  1199999.5
PS D:\15. Tutorial Of Kotlin>
```

# ***Primary Constructor :***

```
// It is not neccessory to write constructor in following syntex.

// If we don't write 'val' by default it is 'var'.  Following is 'Primary' constructor.

class Person constructor(val name : String, val age : Int){

    var greet : String = "You are Welcome"


    fun disp(){

        println(greet)

        println("Name of the person is :  $name")

        println("Age of the person is :  $age")

    }


}


class Human (name : String, age : Int){

    var hName : String

    var hAge : Int

    var ask : String = "How are you ?"


    // Initializer Block

    init {

        hName = name

        hAge = age

    }


    fun disp(){

        println(ask)

        println("Name of the person is :  $hName")

        println("Age of the person is :  $hAge")

    }


}


fun main() {

    val obj1 = Person("Aman", 21)
```

```
    obj1.disp()


    val obj2 = Person("Nikki", 20)

    obj2.greet = "You are also welcome"  // We can access the data by the use of object.

    obj2.disp()


    val obj3 = Human("Aman", 21)

    obj3.disp()



}
```

## ***Output :-***

```
PS D:\15. Tutorial Of Kotlin> cd "d:\15.
 P_PrimaryConstructor.jar }
You are Welcome
Name of the person is :  Aman
Age of the person is :  21
You are also welcome
Name of the person is :  Nikki
Age of the person is :  20
How are you ?
Name of the person is :  Aman
Age of the person is :  21
PS D:\15. Tutorial Of Kotlin>
```

# ***Secondary Constructor :***

```
class People{

    var myName: String

    var age: Int


    constructor(name: String, age: Int){

        println("Constructor called")

        myName = name

        this.age = age

    }


    fun disp(){

        println("Name of the person is :  $myName")

        println("Age of the person is :  $age")

    }


}


fun main() {

    val obj = People("Aman", 21)

    obj.disp()


}
```

# ***Output :-***

```
PS D:\15. Tutorial Of Kotlin> cd "d:
-jar Q_SecondaryConstructor.jar }
Constructor called
Name of the person is :  Aman
Age of the person is :  21
PS D:\15. Tutorial Of Kotlin>
```

# ***Primary And Secondary Constructor :***

```
class Registration (email: String, password: String){

    var name: String = ""

    var age: Int? = null

    var email: String = email // -> We can also initialise variable from here.

    var password: String

    var gender: String = "Male"


    // Primary Constructor

    constructor(name: String, age: Int, email: String, password: String):this(email, password){

        this.name = name

        this.age = age

    }


    // Initializer Block

    init{

        this.password = password

    }


    fun dispValue(){

        println("Name :  $name")

        println("Age :  $age")

        println("Gender :  $gender")

        println("E-mail :  $email")

        println("Password :  $password")

    }


}


fun main() {

    val obj = Registration("Aman", 21, "amanvermalmv211@gmail.com", "122333")

    obj.dispValue()

}
```

## ***Output :-***

```
PS D:\15. Tutorial Of Kotlin> cd "d:\15. Tutorial
 { java -jar R_PrimaryAndSecondaryConst.jar }
Name :  Aman
Age :  21
Gender :  Male
E-mail :  amanvermalmv211@gmail.com
Password :  122333
PS D:\15. Tutorial Of Kotlin>
```

# ***Getter And Setter :***

```
class User(id: Int, name: String, age: Int){

    var id: Int = id

    get() = field

    var name: String = name

    get() = field

    set(value){

        field = value

    }

    var age: Int = age

    get() = field

    set(value){

        field = value

    }

}

fun main() {

    val obj1 = User(101, "Aman Verma", 21)

    println(obj1.id)    // get property

    println(obj1.name)  // get property

    println(obj1.age)   // get property


    obj1.id = 123

    println(obj1.id)


}
```

# ***Output :-***

```
PS D:\15. Tutorial Of Kotlin> cd
terAndSetter.jar }
101
Aman Verma
21
123
PS D:\15. Tutorial Of Kotlin>
```

# ***Inheritance*** :

```
open class Father{

    var car: String = "BMW"

    var money: Int = 123321


    fun disp(){

        println("Father's car :  $car")

        println("Money in parent class :  $money")

    }

}


class Son: Father() {

    var bike: String = "Yamaha RX100"


    fun show(){

        println("Father's car :  $car")

        println("Showing money in child class :  $money")

        println("Son's bike is :  $bike")

    }

}


fun main() {

    val obj = Father()

    obj.disp()


    println("Creating the object of Son :-")

    val obj1 = Son()

    obj1.show()


}
```

# ***Output :-***

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    JUPYTER

Install the latest PowerShell for new features and improvements!

PS D:\15. Tutorial Of Kotlin> cd "d:\15. Tutorial Of Kotlin\" ;
.jar }
Father's car :  BMW
Money in parent class :  123321
Creating the object of Son :-
Father's car :  BMW
Showing money in child class :  123321
Son's bike is :  Yamaha RX100
PS D:\15. Tutorial Of Kotlin>
```

# ***Primary Constructor In Inheritance :***

```
open class Father (Ccar: String, Cmoney: Int){

    var car: String = Ccar

    var money: Int = Cmoney

    fun disp(){

        println("Father's car :  $car")

        println("Money in parent class :  $money")

    }

}

class Son (Ccar: String, Cmoney: Int, Cbike: String) : Father(Ccar, Cmoney) {

    var bike: String = Cbike

    fun show(){

        println("Father's car :  $car")

        println("Showing money in child class :  $money")

        println("Son's bike is :  $bike")

    }

}

fun main() {

    val obj = Father("Alto 800", 123321)

    obj.disp()


    println("Creating the object of Son :-")

    val obj1 = Son("BMW", 321123, "Yamaha RX100")

    obj1.show()


}
```

# ***Output :-***

```
PS D:\15. Tutorial Of Kotlin> cd "d:\15. Tutorial
?) { java -jar U_PConstructorINInheritance.jar }
Father's car :  Alto 800
Money in parent class :  123321
Creating the object of Son :-
Father's car :  BMW
Showing money in child class :  321123
Son's bike is :  Yamaha RX100
PS D:\15. Tutorial Of Kotlin>
```

# ***Secondary Constructor In Inheritance :***

```
open class Father{

    var car: String

    var money: Int


    fun disp(){

        println("Father's car :  $car")

        println("Money in parent class :  $money")

    }


    // Secondary Constructor

    constructor(car: String, money: Int){

        this.car = car

        this.money = money

    }


}


class Son: Father {

    var bike: String


    // Secondary Constructor

    constructor(car: String, money: Int, bike: String) : super(car, money) {

        this.bike = bike

    }


    fun show(){

        println("Father's car :  $car")

        println("Showing money in child class :  $money")

        println("Son's bike is :  $bike")

    }

}

fun main() {

    val obj = Father("Alto 800", 123321)

    obj.disp()
```

```
    println("Creating the object of Son :-")

    val obj1 = Son("BMW", 321123, "Yamaha RX100")

    obj1.show()



}
```

## ***Output :-***

```
PS D:\15. Tutorial Of Kotlin> cd "d:\15. Tutorial
?) { java -jar V_SConstructorINInheritance.jar }
Father's car :  Alto 800
Money in parent class :  123321
Creating the object of Son :-
Father's car :  BMW
Showing money in child class :  321123
Son's bike is :  Yamaha RX100
PS D:\15. Tutorial Of Kotlin>
```

# ***Function Overriding :***

```
open class Father{

    open var car: String = "BMW"

    var money: Int = 123321


    open fun disp(){

        println("Father's car :  $car")

        println("Money in parent class :  $money")

    }

}


class Son: Father() {

    var bike: String = "Yamaha RX100"

    override var car: String = "Audi"


    fun show(){

        println("Father's car :  $car")

        println("Showing money in child class :  $money")

        println("Son's bike is :  $bike")

    }


    override fun disp(){

        println("Overriding done")

    }

}

fun main() {

    val obj = Father()

    obj.disp()


    println("Creating the object of Son :-")

    val obj1 = Son()

    obj1.show()

    obj1.disp()


}
```

## ***Output :-***

```
PS D:\15. Tutorial Of Kotlin> cd "d:\15. Tutorial
verriding.jar }
Father's car :  BMW
Money in parent class :  123321
Creating the object of Son :-
Father's car :  Audi
Showing money in child class :  123321
Son's bike is :  Yamaha RX100
Overriding done
PS D:\15. Tutorial Of Kotlin>
```