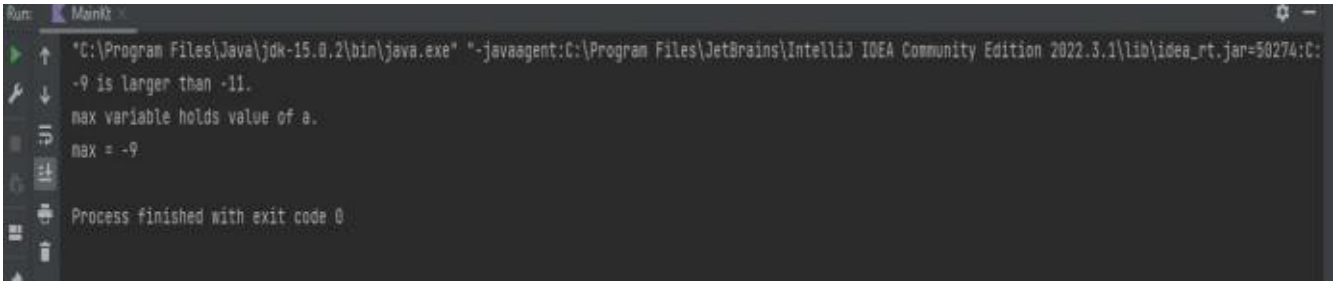
Practical 1- write a program using kotlin to implement control structures and loops Using if-else statement.

```
Val number =-10

if(number>0){

        print(“positive number”)

}

else{

        print(“negative number”)

}
```

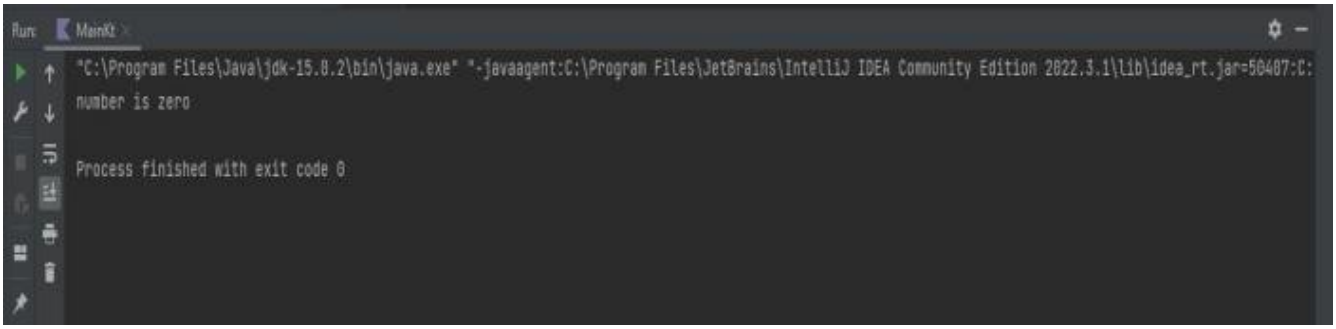
output:



If-elseif ladder:

```
fun main(args: Array<String>) {
        val number = 0
        val result = if (number > 0)
        "positive number"
else if (number < 0)
        "negative number"
else
        "zero"
        println("number is $result")
}
```

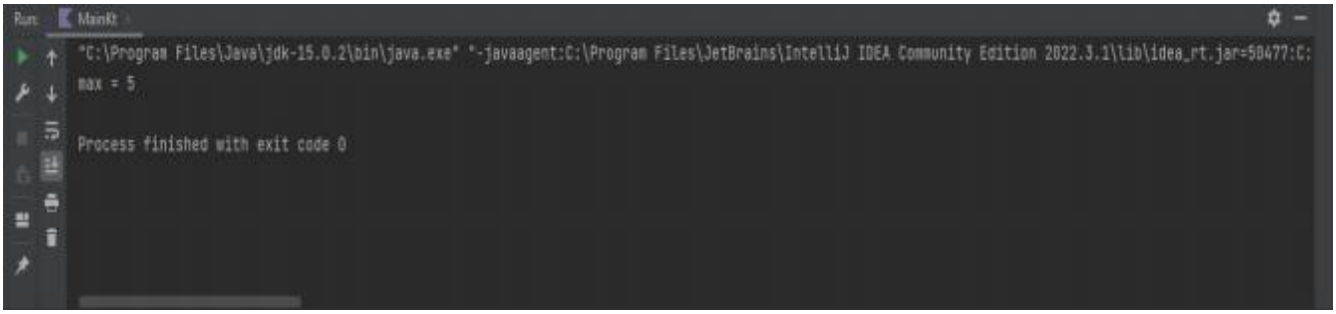
output:



Nested if statements:

```
        fun main(args: Array<String>) {
                val n1 = 3
                val n2 = 5
                val n3 = -2
                val max = if (n1 > n2){
        if (n1 > n3)
                n1
        else
                n3
        } else {
        if (n2 > n3)
                n2
        else
                n3
        }
        println("max = $max")
}
```

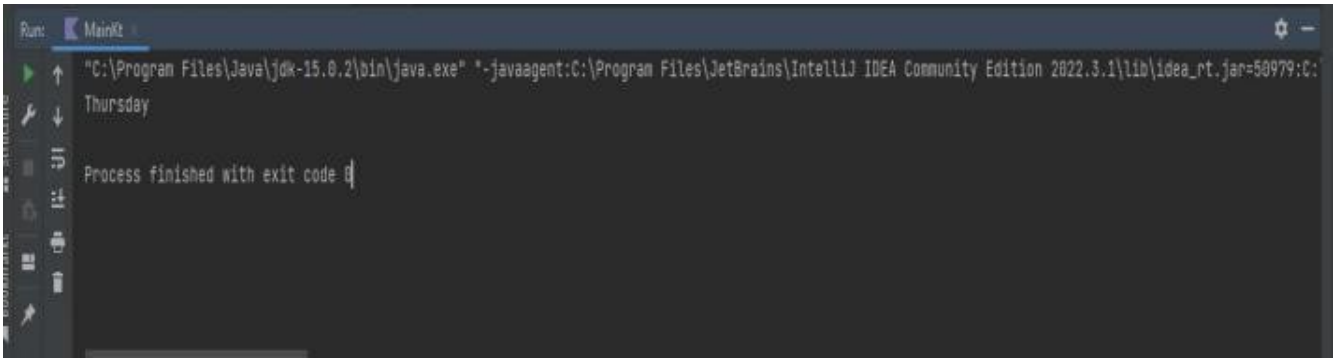
output:



when statement

```
        fun main(args: Array<String>) {
                val day = 4
                val result = when (day) {
                1 -> "Monday"
                2 -> "Tuesday"
                3 -> "Wednesday"
                4 -> "Thursday"
                5 -> "Friday"
                6 -> "Saturday"
                7 -> "Sunday"
        else -> "Invalid day."
        }
                println(result)

        }
```

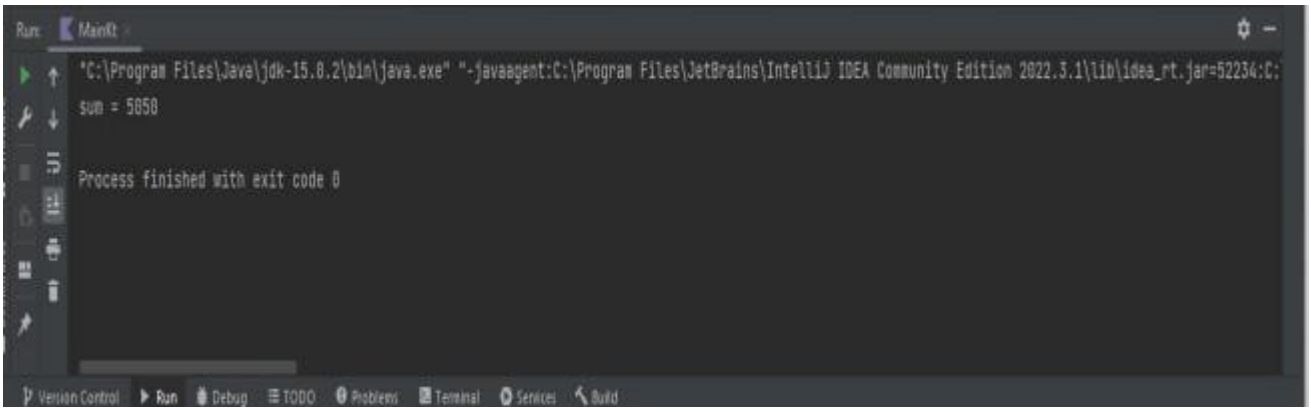
output:



while loop :

```
fun main(args: Array<String>) {
        var sum = 0
        var i = 100
while (i != 0) {
        sum += i
}
        println("sum = $sum")
}
```

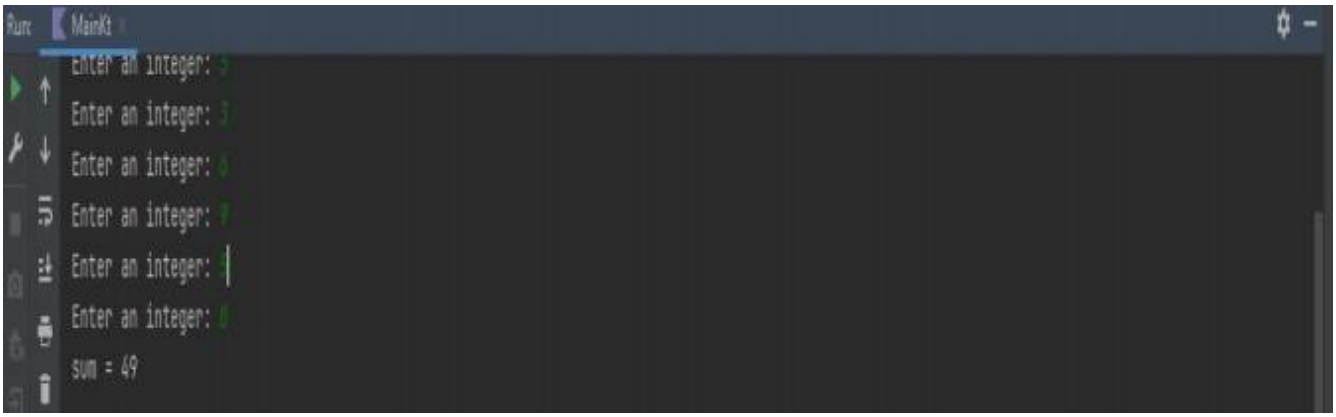
output:



Do-while loop:

```
fun main(args: Array<String>) {
        var sum: Int = 0
        var input: String

do {
        print("Enter an integer: ")
        Input = readLine()!!
        sum += input.toInt()
} while (input != "0")
        println("sum = $sum")
}
```

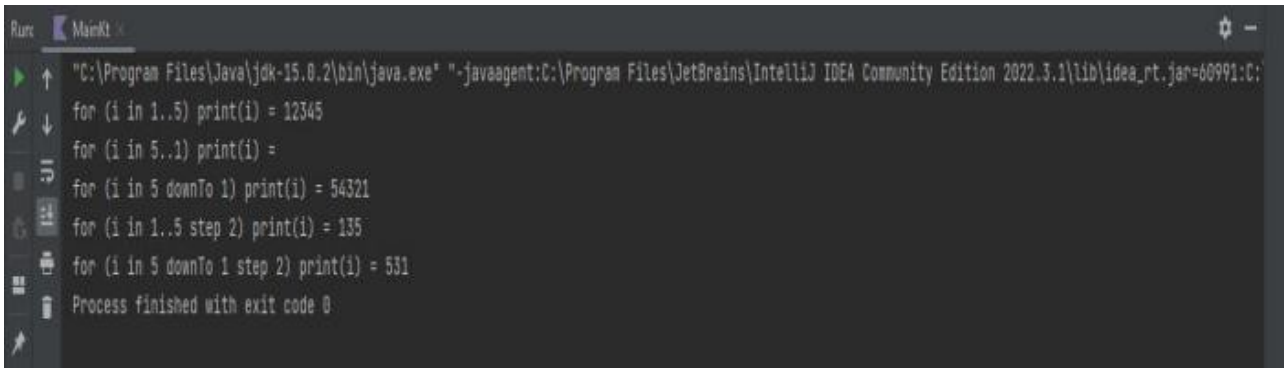
output :



for loop

```
fun main(args: Array<String>) {

    print("for (i in 1..5) print(i) = ")
    for (i in 1..5) print(i)
    println()
    print("for (i in 5..1) print(i) = ")
    for (i in 5..1) print(i)            // prints nothing
    println()
    print("for (i in 5 downTo 1) print(i) = ")
    for (i in 5 downTo 1) print(i)

    println()

    print("for (i in 1..5 step 2) print(i) = ")
    for (i in 1..5 step 2) print(i)

    println()

    print("for (i in 5 downTo 1 step 2) print(i) = ")
    for (i in 5 downTo 1 step 2) print(i)
}
```

Output:



Practical 2- inheritance of class

```
// Superclass
open class MyParentClass {
  val x = 5
}

// Subclass
class MyChildClass: MyParentClass() {
  fun myFunction() {
    println(x) // x is defined in the superclass
  }
}
// Create an object of the MyChildClass and call myFunction
fun main() {
  val myObj = MyChildClass()
  myObj.myFunction()
}
```

Output:

5