LAB 6

Q1. Define a class BankAccount with private fields balance and accountNumber. Provide methods to access these private fields. Also, include methods deposit and withdraw to modify the balance. Create object the class and illustrate use of the created methods.

Ans:- CODE:-

```
class BankAccount(private var balance: Double, private val accountNumber: String) {
  def getBalance: Double = balance

  def getAccountNumber: String = accountNumber

  def deposit(amount: Double): Unit = {
    if (amount > 0) {
      balance += amount
      println(s"Deposited $amount into account $accountNumber")
    } else {
      println("Deposit amount should be greater than 0.")
    }
  }

  def withdraw(amount: Double): Unit = {
    if (amount > 0 && amount <= balance) {
      balance -= amount
      println(s"Withdrew $amount from account $accountNumber")
    } else {
      println("Insufficient funds or invalid withdrawal amount.")
    }
  }
}
val account1 = new BankAccount(1000, "123456789")
println("Current Balance: " + account1.getBalance)
account1.deposit(500)
println("Current Balance: " + account1.getBalance)
account1.withdraw(200)
println("Current Balance: " + account1.getBalance)
//vaswati//
```

OUTPUT:

Output:

```
Current Balance: 1000.0
Deposited 500.0 into account 123456789
Current Balance: 1500.0
Withdrew 200.0 from account 123456789
Current Balance: 1300.0
```

Q2. Define a recursive function sumDigits that takes a double as input and returns the sum of its digits. Write complete scala code to check the function.

Ans: CODE:-

```
object Main extends App {

  def sumDigits(num: Double): Int = {

    val numStr = num.abs.toString

    def sumHelper(index: Int, accumulator: Int): Int = {
      if (index < numStr.length) {

        val digit = numStr.charAt(index).asDigit
        sumHelper(index + 1, accumulator + digit)
      } else {
        accumulator
      }
    }

    sumHelper(0, 0)
  }


  val number = 123.456
  println(s"The sum of digits in $number is: ${sumDigits(number)}")
}

//vaswati//
```

OUTPUT:-

Output:

```
The sum of digits in 123.456 is: 20
```

Q3. Create a list named words containing some strings in Scala. Iterate over the elements of the list words, compute number of vowels in the string using a user defined function and print the results.

Ans:- Code:

```scala
object Main extends App {

  def countVowels(word: String): Int = {
    word.toLowerCase.count(Set('a', 'e', 'i', 'o', 'u'))
  }

  val words = List("Scala", "is", "awesome", "and", "functional")

  words.foreach(word => println(s"Number of vowels in '$word': ${countVowels(word)}"))
}

//vaswati//
```

OUTPUT:-

Output:

```
Number of vowels in 'Scala': 2
Number of vowels in 'is': 1
Number of vowels in 'awesome': 4
Number of vowels in 'and': 1
Number of vowels in 'functional': 4
```

Q4. Define a class Person with attributes names and age. Create a Class Employee that inherits Person class and has additional attributes employee_Id, department, and salary. Person class has a method ‘introduce’ with default arguments that prints "Hi, my name is [name] and I'm [age] years old. Similarly, Employee class has a method ‘employee_details’ with default argumets that prints, “I am [name]. I work in [department] and my employee id is [employee_Id]." Create an instance of both the classes and call the methods created.

Ans: CODE:-

```scala
class Person(val name: String, val age: Int) {
  def introduce(): Unit = {
    println(s"Hi, my name is $name and I'm $age years old.")
  }
}

class Employee(name: String, age: Int, val employeeId: String, val department: String, val salary: Double)
  extends Person(name, age) {

  def employeeDetails(): Unit = {
    println(s"I am $name. I work in $department and my employee id is $employeeId.")
  }
}

object Main extends App {

  val person = new Person("John", 30)
  person.introduce()

  val employee = new Employee("Rohan", 25, "E123", "Engineering", 50000.0)
  employee.introduce()
  employee.employeeDetails()
}

//vaswati//
```

OUTPUT:-

```
Output:

Hi, my name is John and I'm 30 years old.
Hi, my name is Rohan and I'm 25 years old.
I am Rohan. I work in Engineering and my employee id is E123.
```

Q5. Implement a Scala Singleton object named Counter that keeps track of a variable count. Provide methods to increment, decrement, and retrieve the current count value. Demonstrate how to use this Singleton object to maintain a count in your application.

Ans: CODE:-

```scala
object Counter {
  private var count: Int = 0

  def increment(): Unit = {
    count += 1
  }

  def decrement(): Unit = {
    count -= 1
  }

  def getCount: Int = {
    count
  }
}

object Main extends App {

  println("Initial count: " + Counter.getCount)
  Counter.increment()
  println("After increment: " + Counter.getCount)
  Counter.decrement()
  println("After decrement: " + Counter.getCount)
}

//vaswati//
```

OUTPUT:-

```
Output:

Initial count: 0
After increment: 1
After decrement: 0
```