4SETB1.kt

class Employee(val eno: Int, val ename: String, val salary: Double, val designation: String) {

fun displayDetails() {

println("Employee Details:")

println("Emp No: $eno")

println("Name: $ename")

println("Salary: $$salary")

println("Designation: $designation")

}

}

fun main() {

print("Enter Employee Number: ")

val eno = readLine()?.toIntOrNull() ?: 0

print("Enter Employee Name: ")

val ename = readLine() ?: ""

print("Enter Salary: ")

val salary = readLine()?.toDoubleOrNull() ?: 0.0

print("Enter Designation: ")

val designation = readLine() ?: ""

val emp = Employee(eno, ename, salary, designation)

emp.displayDetails()

}

4SETB2.kt

fun sumOfDigits(n: Int): Int {

if (n == 0) return 0

return (n % 10) + sumOfDigits(n / 10) // Recursive call

}

fun main() {

print("Enter a number: ")

val number = readLine()?.toIntOrNull() ?: 0

println("Sum of digits: ${sumOfDigits(number)}")

}

4SETB3.kt

class VowelExtractor(val input: String) {

fun extractVowels(): String {

return input.filter { i in "AEIOUaeiou" } // Filters only vowels

}

}

fun main() {

print("Enter a string: ")

val str = readLine() ?: ""

val extractor = VowelExtractor(str) // Create an object

println("Vowels in the string: ${extractor.extractVowels()}")

}

4SETB4.kt

class BankAccount(private var balance: Double) {

fun deposit(amount: Double) {

if (amount > 0) {

balance += amount

println("Deposited $$amount. New Balance: $$balance")

} else {

println("Invalid deposit amount.")

}

}

fun withdraw(amount: Double) {

if (amount > balance) {

println("Insufficient funds! Available balance: $$balance")

} else if (amount > 0) {

balance -= amount

println("Withdrew $$amount. New Balance: $$balance")

} else {

println("Invalid withdrawal amount.")

}

}

fun showBalance() {

println("Current Balance: $$balance")

}

}

fun main() {

val account = BankAccount(1000.0) // Initial balance

while (true) {

println("\nBanking System")

println("1. Deposit")

println("2. Withdraw")

println("3. Show Balance")

println("4. Exit")

print("Choose an option: ")

when (readLine()?.toIntOrNull()) {

1 -> {

print("Enter amount to deposit: ")

val amount = readLine()?.toDoubleOrNull() ?: 0.0

account.deposit(amount)

}

2 -> {

print("Enter amount to withdraw: ")

val amount = readLine()?.toDoubleOrNull() ?: 0.0

account.withdraw(amount)

}

3 -> account.showBalance()

4 -> {

println("Thank you for using our banking system.")

break

}

else -> println("Invalid option. Try again.")

}

}

}

4SETB5.kt

class UppercaseConverter(val input: String) {

fun convertToUppercase(): String {

return input.uppercase()

}

}

fun main() {

print("Enter a string: ")

val str = readLine() ?: ""

val converter = UppercaseConverter(str)

println("Uppercase: ${converter.convertToUppercase()}")

}