

SCALA PROGRAMMING

Write a Scala program that creates a class `BankAccount` with properties `accountNumber` and `balance`. Implement methods to deposit and withdraw money from the account.

Code:

```
class BankAccount(val accountNumber: String, var balance: Double) {  
  def deposit(amount: Double): Unit = {  
    balance += amount  
    println(s"Deposited $amount. New balance: $balance")  
  }  
  def withdraw(amount: Double): Unit = {  
    if (amount <= balance) {  
      balance -= amount  
      println(s"Withdrew $amount. New balance: $balance")  
    }  
    else  
    {  
      println(s"Want to withdraw $amount? Insufficient balance!")  
    }  
  }  
  def simple_interest(time: Float, rate: Float): Unit={  
    balance+= (balance*time*rate)/100  
    println(s"simple_interest rate for $time year is $rate, New balance: $balance")  
  }  
}  
  
object BankAccountApp {  
  def main(args: Array[String]): Unit = {  
    val account = new BankAccount("SB-1234", 1000.0)  
    println(s"Account Number: ${account.accountNumber}")  
  }  
}
```

```

println(s"Initial Balance: ${account.balance}")

account.deposit(500.0)

account.withdraw(200.0)

account.withdraw(2000.0)

account.simple_interest(1,3.5f)

}

}

```

Output:

Output:

```

Account Number: SB-1234
Initial Balance: 1000.0
Deposited 500.0. New balance: 1500.0
Withdrew 200.0. New balance: 1300.0
Want to withdraw 2000.0? Insufficient balance!
simple_interest rate for 1.0 year is 3.5, New balance: 1345.5

```

Screenshot:

```

1 class BankAccount(val accountNumber: String, var balance: Double) {
2   def deposit(amount: Double): Unit = {
3     balance += amount
4     println(s"Deposited $amount. New balance: $balance")
5   }
6   def withdraw(amount: Double): Unit = {
7     if (amount <= balance) {
8       balance -= amount
9       println(s"Withdrew $amount. New balance: $balance")
10    }
11    else
12    {
13      println(s"Want to withdraw $amount? Insufficient balance!")
14    }
15  }
16  def simple_interest(time: Float, rate: Float): Unit={
17    balance+= (balance*time*rate)/100
18    println(s"simple_interest rate for $time year is $rate, New balance: $balance")
19  }
20 }
21 object BankAccountApp {
22   def main(args: Array[String]): Unit = {
23     val account = new BankAccount("SB-1234", 1000.0)
24     println(s"Account Number: ${account.accountNumber}")
25     println(s"Initial Balance: ${account.balance}")
26     account.deposit(500.0)
27     account.withdraw(200.0)
28     account.withdraw(2000.0)
29     account.simple_interest(1,3.5f)
30   }
31 }

```

STDIN

Input for the program (Optional)

Output:

```

Account Number: SB-1234
Initial Balance: 1000.0
Deposited 500.0. New balance: 1500.0
Withdrew 200.0. New balance: 1300.0
Want to withdraw 2000.0? Insufficient balance!
simple_interest rate for 1.0 year is 3.5, New balance: 1345.5

```