**ADVANCE FUNCTIONAL THINKING LAB**

**LAB 8**

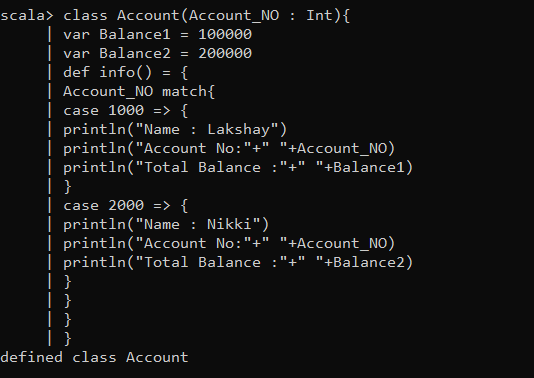
**Lakshay Vasuja**

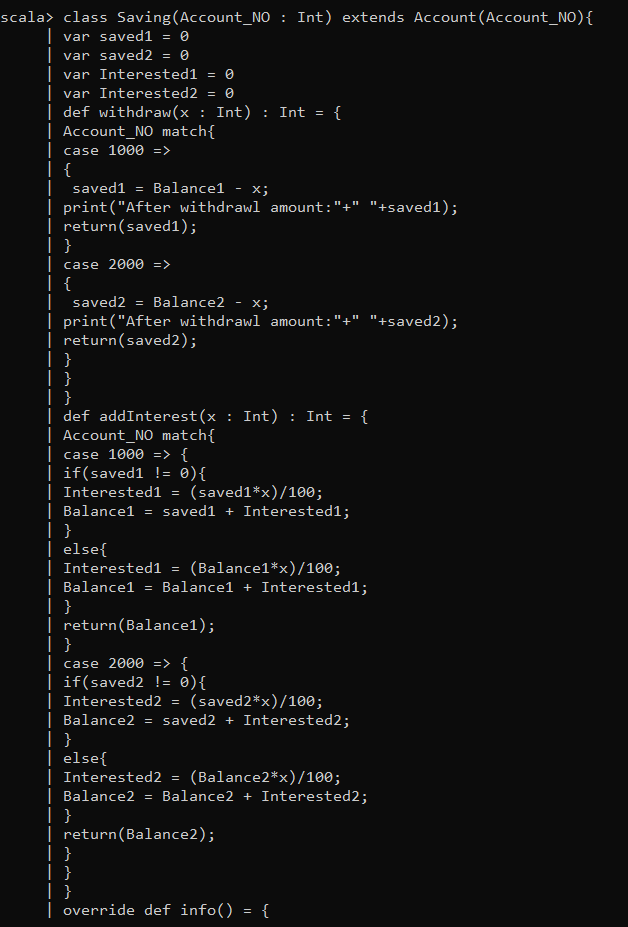
**500067177**

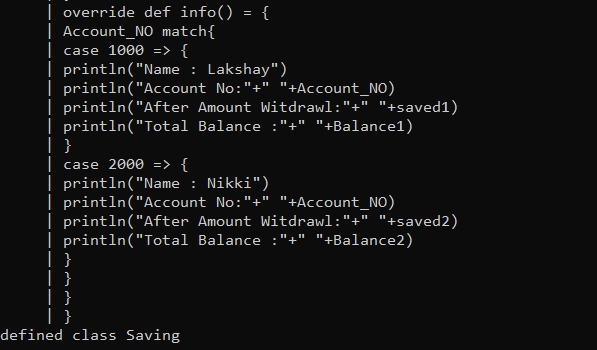
**R172218064**

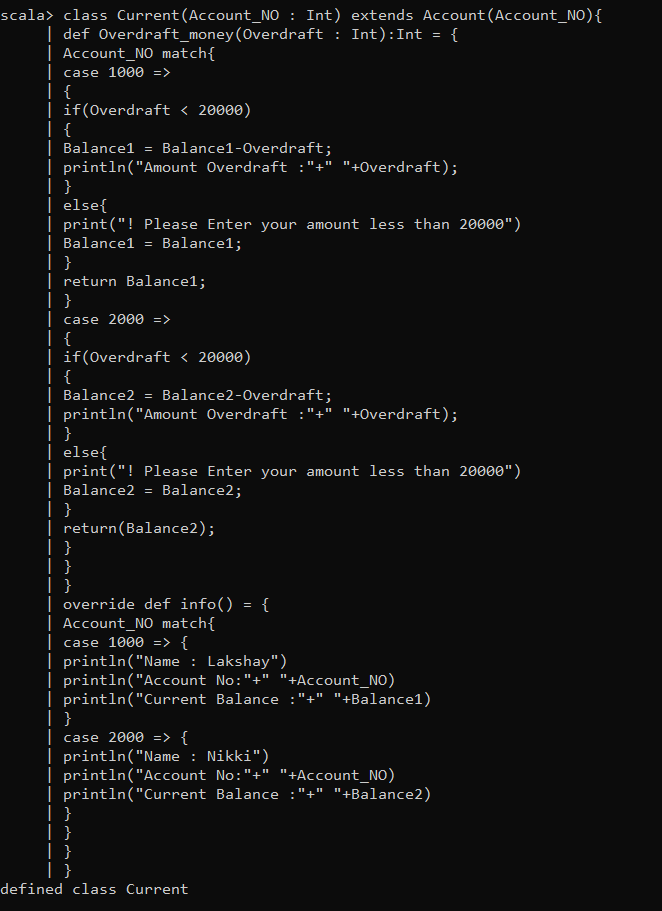
**QUESTION:**

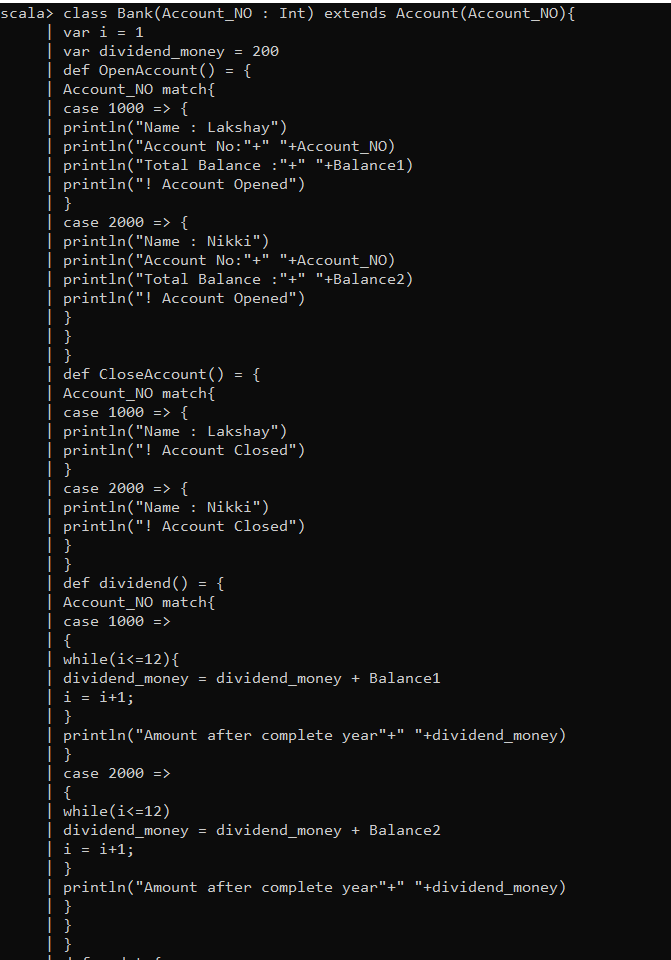
Using the Account class as a base class, write two derived classes called savingsAccount and currentAccount. A savingsAccount object, in addition to the attributes of an account object, should have an interest variable and a method which adds interest to the account. A CurrentAccount object, in addition to the attributes of an Account object, should have an overdraft limit variable. Ensure that you have overridden methods of the Account class as necessary in both derived classes. Now create a Bank class, an object of which contains an array of Account objects. Accounts in the array could be instances of the Account class, the savingsAccount class, or the currentacccunt class. Create some test accounts (some of each type). Write an update method in the bank class. It iterates through each account, updating it in the following ways: Savings accounts get interest added (via the method you already wrote); get a letter sent if they are in overdraft. The Bank class requires methods for opening and closing accounts, and for paying a dividend into each account.

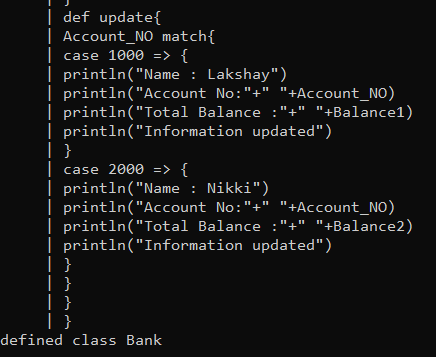




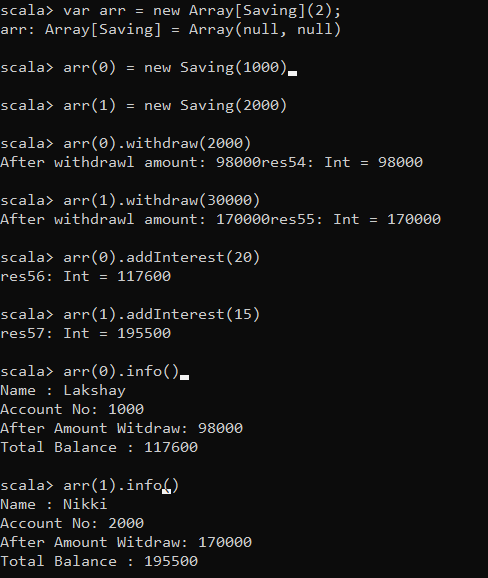


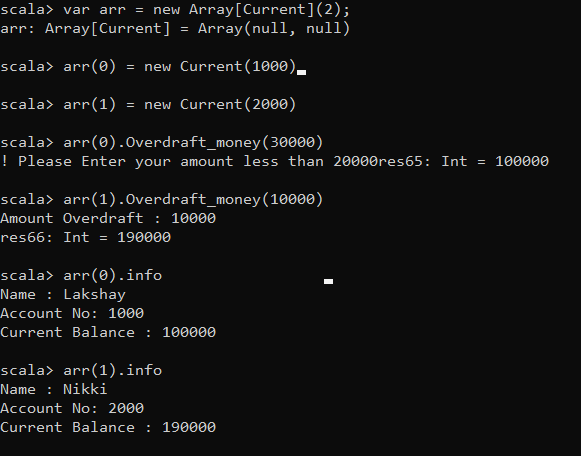


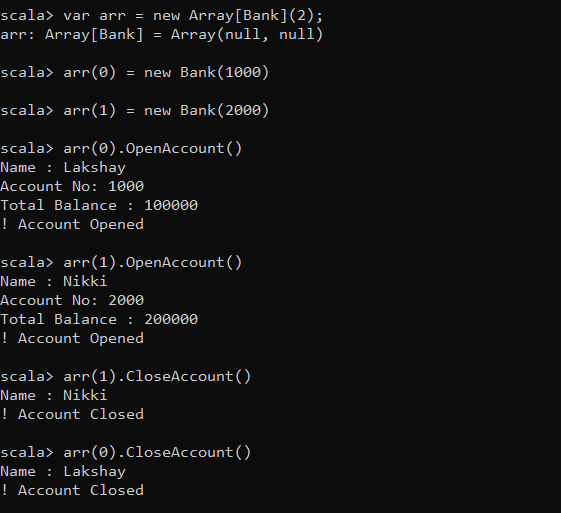




Output:







Code :

class Account(Account\_NO : Int){

var Balance1 = 100000

var Balance2 = 200000

def info() = {

Account\_NO match{

case 1000 => {

println("Name : Lakshay")

println("Account No:"+" "+Account\_NO)

println("Total Balance :"+" "+Balance1)

}

case 2000 => {

println("Name : Nikki")

println("Account No:"+" "+Account\_NO)

println("Total Balance :"+" "+Balance2)

}

}

}

}

class Saving(Account\_NO : Int) extends Account(Account\_NO){

var saved1 = 0

var saved2 = 0

var Interested1 = 0

var Interested2 = 0

def withdraw(x : Int) : Int = {

Account\_NO match{

case 1000 =>

{

saved1 = Balance1 - x;

print("After withdrawl amount:"+" "+saved1);

return(saved1);

}

case 2000 =>

{

saved2 = Balance2 - x;

print("After withdrawl amount:"+" "+saved2);

return(saved2);

}

}

}

def addInterest(x : Int) : Int = {

Account\_NO match{

case 1000 => {

if(saved1 != 0){

Interested1 = (saved1\*x)/100;

Balance1 = saved1 + Interested1;

}

else{

Interested1 = (Balance1\*x)/100;

Balance1 = Balance1 + Interested1;

}

return(Balance1);

}

case 2000 => {

if(saved2 != 0){

Interested2 = (saved2\*x)/100;

Balance2 = saved2 + Interested2;

}

else{

Interested2 = (Balance2\*x)/100;

Balance2 = Balance2 + Interested2;

}

return(Balance2);

}

}

}

override def info() = {

Account\_NO match{

case 1000 => {

println("Name : Lakshay")

println("Account No:"+" "+Account\_NO)

println("After Amount Witdraw:"+" "+saved1)

println("Total Balance :"+" "+Balance1)

}

case 2000 => {

println("Name : Nikki")

println("Account No:"+" "+Account\_NO)

println("After Amount Witdraw:"+" "+saved2)

println("Total Balance :"+" "+Balance2)

}

}

}

}

class Current(Account\_NO : Int) extends Account(Account\_NO){

def Overdraft\_money(Overdraft : Int):Int = {

Account\_NO match{

case 1000 =>

{

if(Overdraft < 20000)

{

Balance1 = Balance1-Overdraft;

println("Amount Overdraft :"+" "+Overdraft);

}

else{

print("! Please Enter your amount less than 20000")

Balance1 = Balance1;

}

return Balance1;

}

case 2000 =>

{

if(Overdraft < 20000)

{

Balance2 = Balance2-Overdraft;

println("Amount Overdraft :"+" "+Overdraft);

}

else{

print("! Please Enter your amount less than 20000")

Balance2 = Balance2;

}

return(Balance2);

}

}

}

override def info() = {

Account\_NO match{

case 1000 => {

println("Name : Lakshay")

println("Account No:"+" "+Account\_NO)

println("Current Balance :"+" "+Balance1)

}

case 2000 => {

println("Name : Nikki")

println("Account No:"+" "+Account\_NO)

println("Current Balance :"+" "+Balance2)

}

}

}

}

class Bank(Account\_NO : Int) extends Account(Account\_NO){

var i = 1

var dividend\_money = 200

def OpenAccount() = {

Account\_NO match{

case 1000 => {

println("Name : Lakshay")

println("Account No:"+" "+Account\_NO)

println("Total Balance :"+" "+Balance1)

println("! Account Opened")

}

case 2000 => {

println("Name : Nikki")

println("Account No:"+" "+Account\_NO)

println("Total Balance :"+" "+Balance2)

println("! Account Opened")

}

}

}

def CloseAccount() = {

Account\_NO match{

case 1000 => {

println("Name : Lakshay")

println("! Account Closed")

}

case 2000 => {

println("Name : Nikki")

println("! Account Closed")

}

}

def dividend() = {

Account\_NO match{

case 1000 =>

{

while(i<=12){

dividend\_money = dividend\_money + Balance1

i = i+1;

}

println("Amount after complete year"+" "+dividend\_money)

}

case 2000 =>

{

while(i<=12)

dividend\_money = dividend\_money + Balance2

i = i+1;

}

println("Amount after complete year"+" "+dividend\_money)

}

}

}

def update{

Account\_NO match{

case 1000 => {

println("Name : Lakshay")

println("Account No:"+" "+Account\_NO)

println("Total Balance :"+" "+Balance1)

println("Information updated")

}

case 2000 => {

println("Name : Nikki")

println("Account No:"+" "+Account\_NO)

println("Total Balance :"+" "+Balance2)

println("Information updated")

}

}

}

}

output6.PNG