**Amrita School of Engineering**

**Department of Computer Science and Engineering**

**19CSE313 – Principles of programming Languages**

**CRUD Operations – Create, Read, Update, Delete**

**Date: 05/05/2022** **Topic: Pair Programming**

**Team Members:**

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Name | Roll No. | Program |
| 1 | Nusum Karthik | CB.EN.U4CSE19444 | Scala |
| 2 | Ravella Abhinav | CB.EN.U4CSE19453 | Haskell |

**Haskell:**

**Code:**

database = [["Abhinav", "19", "2500"], ["Karthik", "20", "2000"]]

*-- Format: [["name","age","salary"]]*

*-- Read all tweets*

readEmpDets :: [[String]]

readEmpDets = database

*-- Create a New Tweet into the list of all tweets*

createEmp :: [String] -> [[String]]

createEmp newEmpArr = database ++ [newEmpArr]

*-- Delete an em from the list of all tweets based on ID*

deleteEmp :: String -> [[String]] -> [[String]]

deleteEmp empName [] = []

deleteEmp empName (x : xs)

  | x !! 0 == empName = xs

  | otherwise = x : (deleteEmp empName xs)

*-- Update a emp on the list of all employees based on ID*

updateEmp :: String -> [String] -> [[String]] -> [[String]]

updateEmp empName changeEmp [] = []

updateEmp empName changeEmp (x : xs)

  | x !! 0 == empName = changeEmp : xs

  | otherwise = x : (updateEmp empName changeEmp xs)

main = do

  putStrLn "==========Menu==========="

  putStrLn "1.  Read Employee"

  putStrLn "2.  Create Employee"

  putStrLn "3.  Delete Employee"

  putStrLn "4.  Update Employee"

  putStrLn "5.  Exit"

  putStrLn "========================="

  putStrLn "Enter your choice: "

  choice <- getLine

  case choice of

    "1" -> do

      putStrLn "Employee Details:"

      putStrLn $ show $ readEmpDets

      main

    "2" -> do

      putStrLn "Enter the Employee Details: "

      empDetails <- getLine

      let empArr = words empDetails

      putStrLn "Employee Details:"

      putStrLn $ show $ createEmp empArr

      main

    "3" -> do

      putStrLn "Enter the Employee name: "

      empName <- getLine

      putStrLn "Employee Details:"

      putStrLn $ show $ deleteEmp empName readEmpDets

      main

    "4" -> do

      putStrLn "Enter the Employee name: "

      empName <- getLine

      putStrLn "Enter the Employee Details: "

      empDetails <- getLine

      let empArr = words empDetails

      putStrLn "Employee Details:"

      putStrLn $ show $ updateEmp empName empArr readEmpDets

      main

    "5" -> putStrLn "Exiting..."

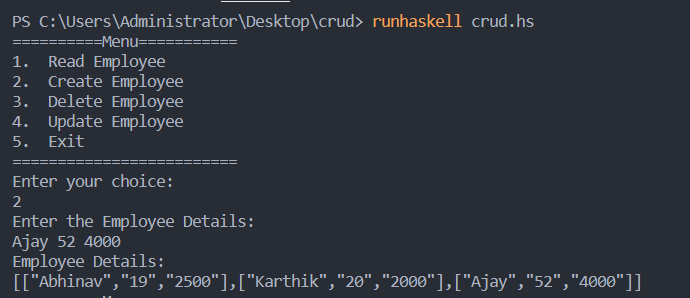
    \_ -> do

      putStrLn "Invalid Choice"

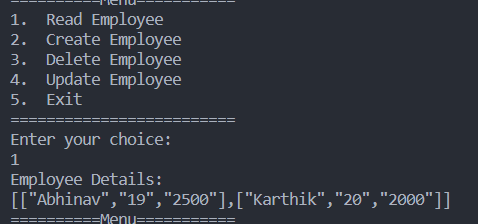
      main

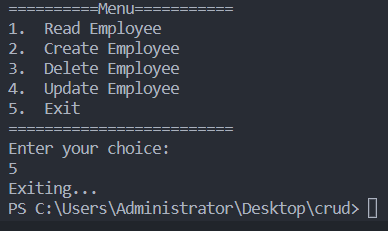
**Output:**

**Create:**

****

**Read:**

****

****

**Scala:**

* **Code:**

import scala.collection.mutable.ArrayBuffer

class Employee(val empname: String, val empage: String , val empsalary: String) {

   var name: String = empname

   var age: String = empage

   var salary: String = empsalary

}

var database:Array[Employee] = new Array[Employee](100)

object Crud{

    def Menu():Int = {

        println("=======Menu========")

        println("1. Insert")

        println("2. Update")

        println("3. Delete")

        println("4. read")

        println("5. Exit")

        println("===================")

        println("\nEnter your choice: ")

        var choice = scala.io.StdIn.readInt()

        return choice

    }

    def insert() = {

        var i = 0

        print("\nEnter name: ")

        var name = scala.io.StdIn.readLine()

        print("\nEnter age: ")

        var age = scala.io.StdIn.readLine()

        print("\nEnter salary: ")

        var salary =  scala.io.StdIn.readLine()

        var obj = new Employee(name, age, salary)

        database(i) = obj

        database(i+1) = null

        i += 1

    }

    def update() = {

        var i = 0

        print("\nEnter name : ")

        var name = scala.io.StdIn.readLine()

        print("\nEnter age : ")

        var age = scala.io.StdIn.readLine()

        print("\nEnter salary : ")

        var salary = scala.io.StdIn.readLine()

        while(database(i) != null){

            if(database(i).name == name){

                database(i).age = age

                database(i).salary = salary

            }

            i += 1

        }

    }

    def delete()= {

        var i = 0

        print("\nEnter name : ")

        var name = scala.io.StdIn.readLine()

*//delete name from database*

        while(database(i) != null){

            if(database(i).name == name){

                database(i) = null

            }

            i += 1

        }

    }

    def read():Unit = {

        var i = 0

        while(i < database.length) {

            if(database(i) != null) {

                println("\nName: " + database(i).name)

                println("Age: " + database(i).age)

                println("Salary: " + database(i).salary)

            }else{

                return

            }

            i += 1

        }

    }

    def main(*args*: Array[String]):Unit = {

        var choice = Menu()

        while(choice != 5){

            if(choice == 1){

                insert()

            }else if(choice == 2){

                update()

            }else if(choice == 3){

                delete()

            }else if(choice == 4){

                read()

            }else if(choice == 5){

                println("\nExiting...")

            }else{

                println("\nInvalid choice")

            }

            choice = Menu()

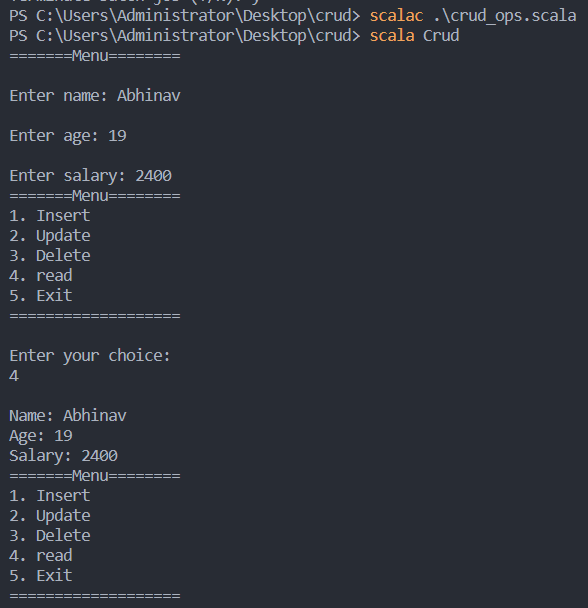
        }

    }

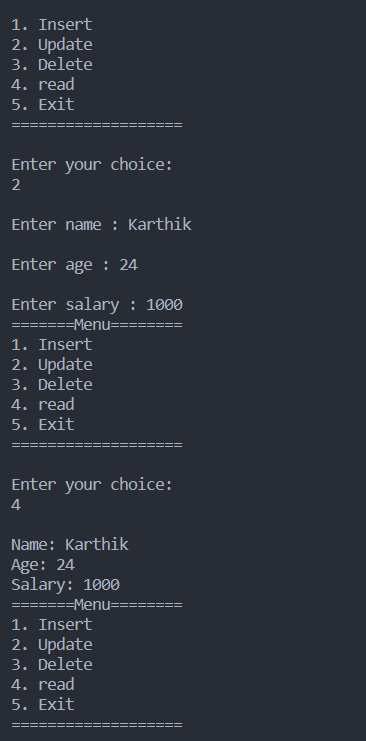
}

**Output:**

**Insert & Read:**

****

**Update:**

****