CS Practical Portfolio

-Sabyasachi Choudhury

# Assignment 1

Problem

Write a user defined function that takes two integer numbers and an arithmetic operator as arguments/parameters and returns the result of the arithmetic operation. The valid arithmetic operators are +, -, \*, / and %. The function s should return “Invalid operator” if the operator is other than +, -, \*, / and %. Also write a main program to invoke this function.

Solution

*"""Assignment 1"""*def calculate(num1, num2, operator):  
 if operator not in "+-\*/%":  
 return "Invalid Operator"  
 else:  
 if operator == "+":  
 return num1 + num2  
 elif operator == "-":  
 return num1-num2  
 elif operator == "\*":  
 return num1\*num2  
 elif operator == "/":  
 return num1/num2  
 elif operator == "%":  
 return num1%num2  
  
n1 = int(input("Enter first number: "))  
n2 = int(input("Enter second number: "))  
op = input("Enter operator from +, -, \*, /, %: ")  
print(calculate(n1, n2, op))

Output

Enter first number: 10  
Enter second number: 22  
Enter operator from +, -, \*, /, %: \*  
220

# Assignment 2

Problem

Write a user defined function that takes an integer number as argument and returns the factorial of that number. Also write a main program to invoke this function.

Solution

*"""Assignment 2"""*def factorial(n):  
 res = 1  
 for i in range(1, n+1):  
 res \*= i  
 return res  
  
num = int(input("Enter number whose factorial you wish to find: "))  
result = factorial(num)  
print(result)

Output

Enter number whose factorial you wish to find: 8  
40320

# Assignment 3

Problem

Write a user defined function that takes a string and an empty dictionary as arguments and the function stores the characters of the string as keys and frequency of its occurrence as value to the dictionary. Also write a main program to invoke this function.

Solution

*"""Assignment 3"""*def letter\_count(string: str, counts: dict):  
 for char in string:  
 if char in counts.keys():  
 counts[char] += 1  
 else:  
 counts[char] = 1  
  
numcount = {}  
s = input("enter string you want to count: ")  
letter\_count(s, numcount)  
for (key, value) in numcount.items():  
 print(f"{key} : {value}")

Output

enter string you want to count: hello! how are you?  
h : 2  
e : 2  
l : 2  
o : 3  
! : 1  
 : 3  
w : 1  
a : 1  
r : 1  
y : 1  
u : 1  
? : 1

# Assignment 4

Problem

Gianluca has started to create a program to analyze pupil attendance. He wants to use a menu system with subprograms.

The program must include subprograms to:

• Display the names of students whose percentage attendance was less than 75%

• Count and display the number of students whose percentage attendance was 90% or higher.

Complete the program code to implement these requirements using the structure given below.

Solution

*"""Assignment 4"""*pupilAttendance = [  
 ["Faroukh" , "Salah" , 70],  
 ["Kelvin", "Glintode" , 85],  
 ["Lara" , "Godfrey" , 90],  
 ["Amara" , "Grzinski" , 70],  
 ["Aaron" , "Grimshaw" , 90],  
 ["Farihaa" , "Mohan" , 95],  
 ["Taz" , "Grimstow" , 60],  
 ["Ali" , "Aisha" , 95],  
 ["Charlene" , "Hall" ,85],  
 ["Asra" , "Ashiq" , 90],  
 ["Sadia" , "Bhatti" , 65],  
 ["Ria" , "Hall" , 90],  
 ["Fernado" , "Askabat" , 60],  
 ["Richard" , "Hawkins" , 80],  
 ["Siyao" , "Wang" , 60],  
 ["Marketta" , "Hosier" , 100]  
]  
  
option = 0  
print("Attendance menu options")  
print("1 - Display names of low attendance")  
print("2 - Display names of high attendance")  
  
def low\_attendance():  
 i = 0  
 for student in pupilAttendance:  
 if student[2] < 75:  
 print(f"{student[0]} {student[1]}")  
 i += 1  
 print(f"\n{i} students have attendance below 75%")  
  
def high\_attendance():  
 i = 0  
 for student in pupilAttendance:  
 if student[2] >= 90:  
 print(f"{student[0]} {student[1]}")  
 i += 1  
 print(f"\n{i} students have attendance above or equal to 90%")  
  
while option != 3:  
 option = int(input("\nEnter option number: "))  
 if option == 1:  
 low\_attendance()  
 elif option == 2:  
 high\_attendance()  
 elif option == 3:

print("exiting program")  
 break  
 else:  
 print("Please enter a valid option")

Output

Attendance menu options  
1 - Display names of low attendance  
2 - Display names of high attendance  
  
Enter option number: 1  
Faroukh Salah  
Amara Grzinski  
Taz Grimstow  
Sadia Bhatti  
Fernado Askabat  
Siyao Wang  
  
6 students have attendance below 75%  
  
Enter option number: 2  
Lara Godfrey  
Aaron Grimshaw  
Farihaa Mohan  
Ali Aisha  
Asra Ashiq  
Ria Hall  
Marketta Hosier  
  
7 students have attendance above or equal to 90%  
  
Enter option number: 3  
exiting program

# Assignment 5

Problem

Robert runs an online order delivery system.

He wants a program to check product codes stored in a file. The file details.txt contains the list of product and product codes.

The program must (assuming that all product and product codes are separated by @ sign ):

ii. check each product code to ensure that:

1. the first three letters of the product code are first three letters of the product.

2. if product code is also including at least one digit (0–9)

3. if a product code does not meet these requirements:

display the product code and write those product codes in a new file called, “Errors.txt” file.

4. If a product code does meet the requirements: display the product code and write those product codes in a new file called, “Finalcodes.txt” file.

Solution

with open("details.txt", "r") as file:  
 bad, good = [], []  
 for string in file.readlines():  
 at = string.index("@")  
 product = string[:at]  
 code = string[at+1:-1]  
 if code[:3] != product[:3]:  
 bad.append(code)  
 else:  
 for char in code:  
 if char in "1234567890":  
 good.append(code)  
 break  
 if good[-1] != code:  
 bad.append(code)  
  
with open("Errors.txt", "w") as file:  
 print("Bad codes")  
 for elem in bad:  
 print(elem)  
 file.write(elem + "\n")  
  
print()  
  
with open("Finalcodes.txt", "w") as file:  
 print("Good codes")  
 for elem in good:  
 print(elem)  
 file.write(elem + "\n")

Console Output

Bad codes  
BALL  
PEN003  
SCALE002  
MICR##  
PEN004  
SCALE003  
  
Good codes  
PENCIL001  
MICRO001  
STEELSCALE123  
DRAWINGPINS23  
PENCIL004  
BALL002  
STEELSCALE124  
DRAWINGPINS2

File Output

PENCIL001  
MICRO001  
STEELSCALE123  
DRAWINGPINS23  
PENCIL004  
BALL002  
STEELSCALE124  
DRAWINGPINS2

# Assignment 6

Problem

**Email.txt** contains a list of email addresses.

Write an UDF EMAILCHECK() to implement these requirements.

The code must:

• check each email address to ensure it contains the ‘@’ symbol.

• write email addresses that do not contain the ‘@’ symbol to an **Error.txt** file.

Write an UDF COUNTUK(filename) to implement these requirements.

The code must:

* Ensure that the function take filename as the argument
* Count and display the emails ending with uk

Solution

def emailcheck():  
 writefile = open("Error.txt", "w")  
 with open("Email.txt", "r") as file:  
 for elem in file:  
 if "@" not in elem:  
 writefile.write(elem)  
 writefile.close()  
  
def countuk(filename):  
 with open(filename, "r") as file:  
 n = 0  
 for elem in file.readlines():  
 if elem[-3:-1] == "uk":  
 n += 1  
 print(elem[:-1])  
 print(f"{n} emails ending with uk")  
  
emailcheck()  
countuk("Email.txt")

Console Output

Ursa.Collins.@anuverplaice.co.uk  
Leanne.Young.@newmail1919.co.uk  
Zane.Quaid.&myschul2018.co.uk  
Aaron.Zachary.@somewer2000.co.uk  
Francis.Ulrich.%newmail1919.co.uk  
Thomas.Adams.@newmail1919.co.uk  
Zane.Adams.\*myschul2018.co.uk  
Zane.Pearson.@anuverplaice.co.uk  
Zane.Montgomerie.@anuverplaice.co.uk  
Victoria.Drew.+newmail1919.co.uk  
Mary.Rawson.@newmail1919.co.uk  
Wilma.Rawson.@myschul2018.co.uk  
Thomas.Evans.@mycoleg2010.co.uk  
Aaron.Valdez.%anuverplaice.co.uk  
Yvette.King.@mycoleg2010.co.uk  
Thomas.Green.@newmail1919.co.uk  
Thomas.Xiong.@newmail1919.co.uk  
Ursa.King.@mycoleg2010.co.uk  
Xavier.Ulrich.@myschul2018.co.uk  
Steven.Zachary.@mycoleg2010.co.uk  
Ursa.Collins.@newmail1919.co.uk  
Colin.Nelson.@somewer2000.co.uk  
Colin.Turner.@newmail1919.co.uk  
Yvette.Hiatt.+somewer2000.co.uk  
Jason.Forrest.\*mycoleg2010.co.uk  
Peter.Winston.@mycoleg2010.co.uk  
Everett.Forrest.+myschul2018.co.uk  
Bonnie.Young.%somewer2000.co.uk  
Yvette.Nelson.@newmail1919.co.uk  
Wilma.Rawson.@myschul2018.co.uk  
Diana.Pearson.@anuverplaice.co.uk  
Peter.Ulrich.\*mycoleg2010.co.uk  
Victoria.Hiatt.@myschul2018.co.uk  
Everett.Green.@myschul2018.co.uk  
George.Jones.@newmail1919.co.uk  
Leanne.Hiatt.\*anuverplaice.co.uk  
Wilma.Montgomerie.%anuverplaice.co.uk  
Karla.Quaid.@mycoleg2010.co.uk  
Diana.Lawrence.@mycoleg2010.co.uk  
Colin.Smith.@somewer2000.co.uk  
Yvette.Smith.@mycoleg2010.co.uk  
Steven.King.@mycoleg2010.co.uk  
Leanne.Young.@mycoleg2010.co.uk  
Randall.Young.@somewer2000.co.uk  
Leanne.Rawson.@newmail1919.co.uk  
Randall.King.@mycoleg2010.co.uk  
Harriett.Turner.@mycoleg2010.co.uk  
Yvette.Zachary.@mycoleg2010.co.uk  
Zane.Brown.@somewer2000.co.uk  
Karla.Pearson.@myschul2018.co.uk  
50 emails ending with uk

File Output

George.Green.+anuverplaice.com  
Mary.Ulrich.%sumwere19.com  
Zane.Quaid.&myschul2018.co.uk  
Francis.Ulrich.%newmail1919.co.uk  
Zane.Adams.\*myschul2018.co.uk  
Wilma.Lawrence.%anuverplaice.com  
Victoria.Drew.+newmail1919.co.uk  
Aaron.Valdez.%anuverplaice.co.uk  
Nancy.Turner.&sumwere19.com  
Mary.Green.%sumwere19.com  
Yvette.Hiatt.+somewer2000.co.uk  
Jason.Forrest.\*mycoleg2010.co.uk  
Ursa.Montgomerie.%mycoleg2010.com  
Karla.Winston.&myschul2018.com  
Everett.Forrest.+myschul2018.co.uk  
Bonnie.Young.%somewer2000.co.uk  
Steven.Nelson.\*myschul2018.com  
Olive.Green.&myschul2018.com  
Ursa.Jones.+newhere2019.com  
Peter.Ulrich.\*mycoleg2010.co.uk  
Leanne.Hiatt.\*anuverplaice.co.uk  
Wilma.Montgomerie.%anuverplaice.co.uk  
Zane.Zachary.&myschul2018.com  
Steven.Forrest.&anuverplaice.com  
Xavier.Ulrich.&mycoleg2010.com  
Aaron.Quaid.&newhere2019.com  
Leanne.Turner.+newhere2019.com  
Randall.Young.&newhere2019.com  
Francis.Forrest.+anuverplaice.com  
Peter.Xiong.&mycoleg2010.com  
Thomas.Jones.\*sumwere19.com

# Assignment 7

Problem

Farshia is the regional manager for an insurance company.

She manages a team of sales staff.

She wants a program to analyse the performance of her team over a number of

months. Sales details are stored in a text file Sales.txt

Write a program to:

• calculate and display the total sales made by each member of the team

• calculate and display the total sales made by the whole team.

# Structure of sales record is

# StaffID, First name, Last name, January sales, February sales,

# March sales, April sales, May sales, June sales

Solution

with open("Sales.txt", "r") as file:  
 total = 0  
 for row in file.readlines():  
 data = row[:-1].split(",")  
 s = sum([int(x) for x in data[3:]])  
 print(f"Employee {data[0]}: {data[1]} {data[2]} | net sales: {s} USD")  
 total += s  
 print(f"\nTotal sales by team: {total} USD")

Output

Employee 101TGY: George Taylor | net sales: 28466 USD  
Employee 103FCY: Fehlix Chayne | net sales: 35250 USD  
Employee 102SBY: Sumren Bergen | net sales: 30186 USD  
Employee 104SBK: Samira Beckle | net sales: 37430 USD  
Employee 105NBT: Nellie Bogart | net sales: 20448 USD  
Employee 106CGT: Cheryl Grouth | net sales: 34383 USD  
Employee 107DGT: Danuta Graunt | net sales: 26421 USD  
Employee 108JDN: Jaiden Deckle | net sales: 21507 USD  
Employee 109JCK: Jimran Caliks | net sales: 32415 USD  
Employee 110DDN: Deynar Derran | net sales: 27459 USD  
  
Total sales by team: 293965 USD

# Assignment 8

Problem

Write a program that will process a series of zeros and ones to read or more credit card transactions by reading in the traction record from a binary file transaction.dat, validating the credit card, and then Outputting the verdict as to the validity of the credit card. A transaction record consists of the following fields stored in the list format:

Name on the credit card: first name, last name

Credit card number

Card expiry date: two digit month [space] two digit year

Transaction number: a positive integer up to 5 digit length

Date: day [space] month [space] year

Time: 24 hour time

Amount:

Example: [Jane, Fred Smith, 4556737586899855, 05 23, 00001, 25 05 2021, 2105, 45.03]

You may assume that all of the information is correctly formatted and is of the appropriate type, so the only form of validation that is required to check whether or not the credit card number is valid. The program will validate VISA card numbers now. The card number is validated by applying Luhn algorithm. For example consider the following 16 digit number 4556737586899855.

* If the first digit is not a 4, the card is invalid
* If the length of the card number is neither 13 or 16, the card is invalid
* The right most digit is check digit. In the example, this is the digit 5. Starting from the check digit and moving to the left double every second digit ( marked in red and underline )

Solution

*"""Assuming credit cards are stored as ints"""*import pickle  
  
with open("transaction.dat", "rb") as file:  
 while True:  
 try:  
 data = pickle.load(file)  
 num = str(data[2])  
   
 valid = True  
  
 if num[0] != 4:  
 valid = False  
 if len(num) not in [13, 16]:  
 valid = False  
 if sum([2\*int(char) for char in num[-2::-2]])%10 != 0:  
 valid = False  
   
 print(data[4], '/'.join(data[5].split(" ")), ':'.join(data[3].split(" ")), "$"+str(data[-1]), data[2], data[0], end=" ")  
 if valid:  
 print("Valid")  
 else:  
 print("Invalid")  
 except:  
 break

Output

1 25/05/2021 05:23 $45.03 4556737586899855 Jane Invalid  
12002 10/02/2020 06:25 $23.89 46724561782234 Frank Invalid  
12 11/09/2022 15:24 $48.45 4455667788992 Heena Invalid

# Assignment 9

Problem

Write a menu driven program using UDFs to perform the following binary file operations on Member.dat

* To create a binary file Member.dat / append Member data : Structure of member data is

{'MemberNo': 8003, 'Name': 'Archana',’Department’:HR’}

* Given a member no, display its associated name, else display appropriate error message.
* Remove all the members belonging to Sales Department from the file.
* Display all members

Solution

import pickle  
  
"""Assuming member number is unique"""  
  
def member\_name(num):  
 found = False  
 with open("Member.dat", "rb") as file:  
 while True:  
 try:  
 data = pickle.load(file)  
 if data['MemberNo'] == num:  
 print(data['Name'])  
 found = True  
 break  
 except:  
 break  
 if not found:  
 print("No such member")  
  
def del\_sales():  
 new\_data = []  
 with open("Member.dat", "rb") as file:  
 while True:  
 try:  
 data = pickle.load(file)  
 if data['Department'].lower() != 'sales':  
 new\_data.append(data)  
 except:  
 break  
 with open("Member.dat", "wb") as file:  
 for elem in new\_data:  
 pickle.dump(elem, file)  
  
def display():  
 with open("Member.dat", "rb") as file:  
 while True:  
 try:  
 print(pickle.load(file))  
 except:  
 break  
  
def new\_file(elems):  
 with open("Member.dat", "ab") as file:  
 for row in elems:  
 pickle.dump(row, file)  
  
print("Choose options")  
print("1: Update file")  
print("2: Find name by member number")  
print("3: Remove all sales members")  
print("4: Display all members")  
print("5: Exit Program")  
choice = int(input("Enter choice: "))  
while choice != 5:  
   
 if choice == 1:  
 print("Enter data in following format")  
 print("<Member number 1>,<Name 1>,<Department 1>")  
 print("<Member number 2>,<Name 2>,<Department 2>")  
 print("Keep entering until data completed, at which point enter '0'")  
 x = input()  
 data = []  
 while x != '0':  
 x = x.split(',')  
 data.append({'MemberNo': int(x[0]), 'Name': x[1], 'Department': x[2]})  
 x = input()  
 new\_file(data)  
 print("Changes have been made. Display all members to see changes")  
  
 elif choice == 2:  
 x = int(input('Enter member number: '))  
 member\_name(x)  
  
 elif choice == 3:  
 del\_sales()  
 print("Changes have been made. Display all members to see changes")  
  
 elif choice == 4:  
 display()  
  
 elif choice == 5:  
 print("Exiting program")  
 break  
  
 choice = int(input("Enter next choice: "))

Output

Choose options  
1: Update file  
2: Find name by member number  
3: Remove all sales members  
4: Display all members  
5: Exit Program  
Enter choice: 4  
{'MemberNo': 1002, 'Name': 'archita', 'Department': 'HR'}  
{'MemberNo': 2984, 'Name': 'Sabya', 'Department': 'RnD'}  
{'MemberNo': 32048, 'Name': 'Avni', 'Department': 'Health Services'}  
{'MemberNo': 2123, 'Name': 'Kabir', 'Department': 'Marketing'}  
{'MemberNo': 222, 'Name': 'Anjelica', 'Department': 'Janitor'}  
{'MemberNo': 30273, 'Name': 'Falak', 'Department': 'RnD'}  
Enter next choice: 2  
Enter member number: 2984  
Sabya  
Enter next choice: 1  
Enter data in following format  
<Member number 1>,<Name 1>,<Department 1>  
<Member number 2>,<Name 2>,<Department 2>  
Keep entering until data completed, at which point enter '0'  
3948,Dhaval,sales  
0  
Changes have been made. Display all members to see changes  
Enter next choice: 4  
{'MemberNo': 1002, 'Name': 'archita', 'Department': 'HR'}  
{'MemberNo': 2984, 'Name': 'Sabya', 'Department': 'RnD'}  
{'MemberNo': 32048, 'Name': 'Avni', 'Department': 'Health Services'}  
{'MemberNo': 2123, 'Name': 'Kabir', 'Department': 'Marketing'}  
{'MemberNo': 222, 'Name': 'Anjelica', 'Department': 'Janitor'}  
{'MemberNo': 30273, 'Name': 'Falak', 'Department': 'RnD'}  
{'MemberNo': 3948, 'Name': 'Dhaval', 'Department': 'sales'}  
Enter next choice: 3  
Changes have been made. Display all members to see changes  
Enter next choice: 4  
{'MemberNo': 1002, 'Name': 'archita', 'Department': 'HR'}  
{'MemberNo': 2984, 'Name': 'Sabya', 'Department': 'RnD'}  
{'MemberNo': 32048, 'Name': 'Avni', 'Department': 'Health Services'}  
{'MemberNo': 2123, 'Name': 'Kabir', 'Department': 'Marketing'}  
{'MemberNo': 222, 'Name': 'Anjelica', 'Department': 'Janitor'}  
{'MemberNo': 30273, 'Name': 'Falak', 'Department': 'RnD'}  
Enter next choice: 5  
Exiting program

# Assignment 10

Problem

Ria is a school librarian.

She wants a program to analyse pupil use of the library.

She wants to encourage reading by awarding gold, silver and bronze medals to the three pupils who have read the most books.

Test data has been stored in a binary file library.dat. Structure of each record in the binary file is (Student Index number, First name, last name, number of books read)

Write a program to read each record, calculate and display:

• the total number and average number of books pupils have read

• the IDs of pupils who have read fewer than ten books

• the details of the gold, silver and bronze medal winners.

Solution

import pickle  
  
data = []  
with open("library.dat", "rb") as file:  
 while True:  
 try:  
 data.append(pickle.load(file))  
 except EOFError:  
 break  
  
key = lambda x: x[-1]  
data.sort(key = key)  
  
# Total and Average Books  
s = sum([k[-1] for k in data])  
print(f"Total Books Read: {s}\nAverage Books Read: {s/len(data)}")  
  
# less than 10 books  
print("\nThe following students have read less than 10 books")  
for student in data:  
 if student[-1] < 10:  
 print(student[0])  
 else:  
 break  
  
# Top 3  
print("\nGold Medallist")  
print(f"ID: {data[-1][0]} | Name: {data[-1][1]} {data[-1][2]} | Books Read: {data[-1][3]}\n")  
print("Silver Medallist")  
print(f"ID: {data[-2][0]} | Name: {data[-2][1]} {data[-2][2]} | Books Read: {data[-2][3]}\n")  
print("Bronze Medallist")  
print(f"ID: {data[-3][0]} | Name: {data[-3][1]} {data[-3][2]} | Books Read: {data[-3][3]}\n")

Output

Total Books Read: 846  
Average Books Read: 42.3  
  
The following students have read less than 10 books  
103AZ  
123SG  
  
Gold Medallist  
ID: 127AL | Name: Abduraheim Leahy | Books Read: 94  
  
Silver Medallist  
ID: 134KD | Name: Kevin Dawson | Books Read: 74  
  
Bronze Medallist  
ID: 128JS | Name: Justin Slater | Books Read: 68

# Assignment 11

Problem

You work for the Bank of Tawara. The bank sends a csv file encryptTransactions.csv containing recent transactions in an encrypted form. You will decrypt the transactions. Each transaction must be decrypted and stored in a csv file decrypt.csv in the following format:

Following are the rules to decrypt the transactions:

 ·   the first digit and place this in the Transmitted check digit

• the next 6 digits and display these as the Bank sort code,

for example if the digits are 240912 the bank sort code would be displayed as 24-09-12

• the next 9 digits and display these as a 9-digit Account number

• the next 8 digits and display these in the Date digits

• the next character and display this in the Transaction

• any other digits and place these in the Working 2

·   Calculated check digit  to calculate a single digit checksum (by adding all the digits in the account number column and extracting the last digit from this total).

·   Working1  to compare the Transmitted and Calculated check digits.

If the two data items for each transaction do not match, indicate that there is an error.

· Calculate the value of each transaction in Credit or Debit. The Transaction must be used and the value held in the Working 2 is the value of each transaction in cents (there are 100 cents in a dollar). The letters C and D in the Transaction column represent Credit and Debit. either the credit or debit.

Solution

import csv  
  
def decrypt():  
 f2 = open("decrypt.csv", "w", newline='')  
 writer = csv.writer(f2)  
 to\_write = [['Encrypted',  
 'Date Digits',  
 'Date of Transaction',  
 'Bank Sort Code',  
 'Account Number',  
 'Transmitted Check Digit',  
 'Calculated Check Digit',  
 'Working 1',  
 'Transaction',  
 'Working 2',  
 'Credit',  
 'Debit']]  
  
 with open('encryptTransactions.csv', "r") as file:  
 for [x] in csv.reader(file):  
 check\_digit = int(x[0])  
 bank\_sort\_code = f"{x[1]}{x[2]}-{x[3]}{x[4]}-{x[5]}{x[6]}"  
 account\_number = x[7:16]  
 date\_digits = x[16:24]  
 date = f"{x[16]}{x[17]}/{x[18]}{x[19]}/{x[20:24]}"  
 transaction = x[24]  
 working\_2 = x[25:]  
 check\_sum = sum([int(c) for c in account\_number])%10  
 if check\_sum == check\_digit:  
 working\_1 = 'checks match'  
 else:  
 working\_1 = 'checks failed'  
 if transaction == 'C':  
 credit = int(working\_2)/100  
 debit = 0  
 else:  
 debit = int(working\_2)/100  
 credit = 0  
  
 to\_write.append([x, date\_digits, date, bank\_sort\_code, account\_number, check\_digit, check\_sum, working\_1, transaction, working\_2, credit, debit])  
  
 writer.writerows(to\_write)  
 f2.close()  
  
def display():  
 reader = csv.reader(open("decrypt.csv", "r"))  
 for elem in reader:  
 print(elem)  
  
decrypt()  
display()

Output

['Encrypted', 'Date Digits', 'Date of Transaction', 'Bank Sort Code', 'Account Number', 'Transmitted Check Digit', 'Calculated Check Digit', 'Working 1', 'Transaction', 'Working 2', 'Credit', 'Debit']  
['626155401952554714012018C4251', '14012018', '14/01/2018', '26-15-54', '019525547', '6', '8', 'checks failed', 'C', '4251', '42.51', '0']  
['126157735214568714012018D125400', '14012018', '14/01/2018', '26-15-77', '352145687', '1', '1', 'checks match', 'D', '125400', '0', '1254.0']  
['526154401254852415012018C6500', '15012018', '15/01/2018', '26-15-44', '012548524', '5', '1', 'checks failed', 'C', '6500', '65.0', '0']  
['534125436279130415012018C564213', '15012018', '15/01/2018', '34-12-54', '362791304', '5', '5', 'checks match', 'C', '564213', '5642.13', '0']  
['234012101205296715012018D168573', '15012018', '15/01/2018', '34-01-21', '012052967', '2', '2', 'checks match', 'D', '168573', '0', '1685.73']

# Assignment 12

Problem

Write a menu driven program using UDFs to perform the following operations on a CSV file named PLANTS\_STOCK.csv

1. Add a new plant
2. Search a plant using its ReferenceID
3. Update a plant ( increase the price of all medium (‘M’) height plants by 1)
4. Display the plants whose Category ‘Climber’
5. Delete a plant using its ReferenceID

The structure of each row of the PLANTS\_STOCK.csv is as follows:

[Category,Name,Variety,Code,ReferenceID,Unit,Price,Height]

Solution

import csv  
  
filename = "PLANTS\_STOCK.csv"  
  
def display\_all():  
 with open(filename, "r") as file:  
 reader = csv.reader(file)  
 for elem in reader:  
 print(elem)  
  
def add\_plant():  
 category = input("Enter plant category:")  
 name = input("Enter plant name:")  
 variety = input("Enter plant variety:")  
 code = input("Enter plant code:")  
 reference\_id = input("Enter plant reference\_id:")  
 unit = input("Enter plant unit:")  
 price = input("Enter plant price:")  
 height = input("Enter plant height:")  
 with open(filename, "a", newline='') as file:  
 writer = csv.writer(file)  
 writer.writerow([category, name, variety, code, reference\_id, unit, price, height])  
  
def search\_id(ref):  
 with open(filename, 'r') as file:  
 reader = csv.reader(file)  
 found = False  
 for elem in reader:  
 if elem[4] == ref:  
 found = True  
 print(elem)  
 break  
 if not found:  
 print("ID not found")  
  
def update\_medium():  
 new\_data = []  
 with open(filename, 'r') as file:  
 reader = csv.reader(file)  
 for elem in reader:  
 if elem[-1] == 'M':  
 elem[-2] = str(float(elem[-2]) + 1)  
 new\_data.append(elem)  
  
 with open(filename, 'w', newline='') as file:  
 writer = csv.writer(file)  
 writer.writerows(new\_data)  
  
def display\_climber():  
 with open(filename, 'r') as file:  
 reader = csv.reader(file)  
 for elem in reader:  
 if elem[0].lower() == 'climber':  
 print(elem)  
  
def delete(ref):  
 data = [elem for elem in csv.reader(open(filename, 'r')) if elem[4] != ref]  
 with open(filename, 'w', newline='') as file:  
 w = csv.writer(file)  
 w.writerows(data)  
  
print("Choose option from following")  
s = """  
1. Add a new plant  
2. Search for a plant with its reference ID  
3. Increase price of all medium plants by 1  
4. Display all climbers  
5. Delete a plant using its refID  
6. Display all plants  
7. Show options again  
0. Exit program  
"""  
  
print(s)  
x = int(input("Pick an Option: "))  
  
while x != 0:  
 if x == 1:  
 add\_plant()  
 elif x == 2:  
 search\_id(input("Enter Reference ID: "))  
 elif x == 3:  
 update\_medium()  
 elif x == 4:  
 display\_climber()  
 elif x == 5:  
 delete(input("Enter Reference ID: "))  
 elif x == 6:  
 display\_all()  
 elif x == 7:  
 print(s)  
 else:  
 print("No such Option")  
  
 x = int(input("\n\nPick next option: "))

Output

Choose option from following  
  
1. Add a new plant  
2. Search for a plant with its reference ID  
3. Increase price of all medium plants by 1  
4. Display all climbers  
5. Delete a plant using its refID  
6. Display all plants  
7. Show options again  
0. Exit program  
  
Pick an Option: 4  
['Climber', 'Clematis', 'Bouchard', 'P', 'C02', '1', '5.00', 'T']  
['Climber', 'Rose', 'Compassion', 'Y', 'C11', '1', '8.00', 'T']  
['Climber', 'Vine', 'Crimson Glory', 'P', 'C26', '1', '10.50', 'T']  
['Climber', 'Honeysuckle', 'Belgica', 'YP', 'C51', '1', '12.25', 'M']  
['Climber', 'Jasmin', 'Winter', 'YPB', 'C59', '1', '7.50', 'T']  
['Climber', 'Wisteria', 'Wonder', 'B', 'C64', '1', '15.00', 'T']  
  
  
Pick next option: 6  
['Category', 'Name', 'Variety', 'Code', 'ReferenceID', 'Unit', 'Price', 'Height']  
['Perennial', 'Aconitem', 'Monkshood', 'B', 'P24', '1', '7.0', 'M']  
['Perennial', 'Anemone', 'Pink Star', 'P', 'P29', '3', '8.50', 'S']  
['Perennial', 'Astilbe', 'Regal', 'WP', 'P32', '1', '7.5', 'M']  
['Perennial', 'Echinops', 'Blue Wing', 'B', 'P38', '1', '7.25', 'M']  
['Perennial', 'Hosta', 'Albo', 'V', 'P41', '1', '3.50', 'S']  
['Perennial', 'Iris', 'Sibirica', 'Y', 'P42', '3', '10.0', 'M']  
['Perennial', 'Phlox', 'Edward', 'RW', 'P46', '1', '3.30', 'S']  
['Patio', 'Eunonymous', 'Mixed', 'VY', 'B07', '12', '12.50', 'S']  
['Patio', 'Hebe', 'Varigatum', 'P', 'B11', '1', '16.0', 'M']  
['Patio', 'Pieris', 'Carnival', 'B', 'B17', '1', '16.0', 'M']  
['Patio', 'Begonia', 'Cocunut Ice', 'Y', 'B21', '1', '4.50', 'S']  
['Patio', 'Begonia', 'Cocunut Ice', 'Y', 'B22', '3', '9.00', 'S']  
['Patio', 'Begonia', 'Cocunut Ice', 'Y', 'B23', '9', '19.50', 'S']  
['Patio', 'Dahlia', 'Moonfire', 'B', 'B24', '1', '9.5', 'M']  
['Patio', 'Carnation', 'Trailing', 'RP', 'B35', '5', '12.25', 'M']  
['Patio', 'Bamboo', 'Pleioblastus', 'N', 'B43', '1', '10.0', 'M']  
['Climber', 'Clematis', 'Bouchard', 'P', 'C02', '1', '5.00', 'T']  
['Climber', 'Rose', 'Compassion', 'Y', 'C11', '1', '8.00', 'T']  
['Climber', 'Vine', 'Crimson Glory', 'P', 'C26', '1', '10.50', 'T']  
['Climber', 'Honeysuckle', 'Belgica', 'YP', 'C51', '1', '12.25', 'M']  
['Climber', 'Jasmin', 'Winter', 'YPB', 'C59', '1', '7.50', 'T']  
['Climber', 'Wisteria', 'Wonder', 'B', 'C64', '1', '15.00', 'T']  
['Fruit', 'Apple', 'Bramley', 'G', 'F17', '1', '16.50', 'T']  
['Fruit', 'Apple', 'Cox', 'R', 'F19', '1', '16.30', 'T']  
['Fruit', 'Apple', 'Discovery', 'G', 'F26', '1', '16.25', 'T']  
['Fruit', 'Cherry', 'June', 'R', 'F31', '1', '16.00', 'T']  
['Fruit', 'Pear', 'Conference', 'G', 'F32', '1', '16.00', 'T']  
['Fruit', 'Plum', 'Victoria', 'P', 'F48', '1', '16.25', 'T']  
['Fruit', 'Peach', 'Nancy', 'O', 'F72', '1', '20.0', 'M']  
['Fuchsia', 'Annabel', 'Erne', 'W', 'U17', '6', '6.95', 'S']  
['Fuchsia', 'Darkeyes', 'Bann', 'RW', 'U21', '6', '6.50', 'S']  
['Fuchsia', 'Dancing Flame', 'Lagan', 'R', 'U25', '6', '10.95', 'M']  
['Fuchsia', 'Thalia', 'Strule', 'PR', 'U37', '6', '6.50', 'S']  
['Fuchsia', 'Swingtime', 'Shimna', 'PW', 'U52', '6', '10.25', 'M']  
['Hedge', 'Berberis', 'Barberry', 'Y', 'H25', '5', '39.0', 'M']  
['Hedge', 'Holly', 'Common', 'G', 'H29', '1', '50.00', 'T']  
['Hedge', 'Bamboo', 'Blue Zion', 'B', 'H59', '10', '149.00', 'T']  
['Hedge', 'Pyracantha', 'Orat', 'O', 'H60', '5', '34.0', 'M']  
['Hedge', 'Dog Wood', 'Red Doa', 'R', 'H64', '10', '50.00', 'T']  
['Shrub', 'Acer', 'Purple Mood', 'P', 'S08', '1', '18.00', 'T']  
['Shrub', 'Buddleia', 'White Will', 'W', 'S14', '1', '11.5', 'M']  
['Shrub', 'Hebe', 'Varigata', 'VR', 'S21', '3', '20.50', 'S']  
['Shrub', 'Daphne', 'Purple Haze', 'P', 'S38', '1', '16.50', 'S']  
['Shrub', 'Camellia', 'Sure Red', 'R', 'S60', '1', '17.0', 'M']  
['Shrub', 'Hydrangea', 'Elizabeth', 'B', 'S69', '1', '7.00', 'T']  
['Shrub', 'Magnolia', 'Soulang', 'WP', 'S73', '1', '19.0', 'M']  
['Shrub', 'Magnolia', 'Paul', 'W', 'S79', '1', '19.0', 'M']  
['Shrub', 'Bay Tree', 'Standard', 'G', 'S81', '1', '74.0', 'M']  
['Shrub', 'Bay Tree', 'Pyramid', 'G', 'S83', '1', '40.00', 'S']  
  
  
Pick next option: 2  
Enter Reference ID: U25  
['Fuchsia', 'Dancing Flame', 'Lagan', 'R', 'U25', '6', '10.95', 'M']  
  
  
Pick next option: 3  
Prices updated. Display all plants to see change  
  
  
Pick next option: 5  
Enter Reference ID: S73  
plant S73 deleted. Display all plants to see change  
  
  
Pick next option: 1  
Enter plant category:Shrub  
Enter plant name:Bay Tree  
Enter plant variety:Cuboid  
Enter plant code:G  
Enter plant reference\_id:S85  
Enter plant unit:1  
Enter plant price:45  
Enter plant height:M  
New plant added. Display all data to see change  
  
  
Pick next option: 6  
['Category', 'Name', 'Variety', 'Code', 'ReferenceID', 'Unit', 'Price', 'Height']  
['Perennial', 'Aconitem', 'Monkshood', 'B', 'P24', '1', '8.0', 'M']  
['Perennial', 'Anemone', 'Pink Star', 'P', 'P29', '3', '8.50', 'S']  
['Perennial', 'Astilbe', 'Regal', 'WP', 'P32', '1', '8.5', 'M']  
['Perennial', 'Echinops', 'Blue Wing', 'B', 'P38', '1', '8.25', 'M']  
['Perennial', 'Hosta', 'Albo', 'V', 'P41', '1', '3.50', 'S']  
['Perennial', 'Iris', 'Sibirica', 'Y', 'P42', '3', '11.0', 'M']  
['Perennial', 'Phlox', 'Edward', 'RW', 'P46', '1', '3.30', 'S']  
['Patio', 'Eunonymous', 'Mixed', 'VY', 'B07', '12', '12.50', 'S']  
['Patio', 'Hebe', 'Varigatum', 'P', 'B11', '1', '17.0', 'M']  
['Patio', 'Pieris', 'Carnival', 'B', 'B17', '1', '17.0', 'M']  
['Patio', 'Begonia', 'Cocunut Ice', 'Y', 'B21', '1', '4.50', 'S']  
['Patio', 'Begonia', 'Cocunut Ice', 'Y', 'B22', '3', '9.00', 'S']  
['Patio', 'Begonia', 'Cocunut Ice', 'Y', 'B23', '9', '19.50', 'S']  
['Patio', 'Dahlia', 'Moonfire', 'B', 'B24', '1', '10.5', 'M']  
['Patio', 'Carnation', 'Trailing', 'RP', 'B35', '5', '13.25', 'M']  
['Patio', 'Bamboo', 'Pleioblastus', 'N', 'B43', '1', '11.0', 'M']  
['Climber', 'Clematis', 'Bouchard', 'P', 'C02', '1', '5.00', 'T']  
['Climber', 'Rose', 'Compassion', 'Y', 'C11', '1', '8.00', 'T']  
['Climber', 'Vine', 'Crimson Glory', 'P', 'C26', '1', '10.50', 'T']  
['Climber', 'Honeysuckle', 'Belgica', 'YP', 'C51', '1', '13.25', 'M']  
['Climber', 'Jasmin', 'Winter', 'YPB', 'C59', '1', '7.50', 'T']  
['Climber', 'Wisteria', 'Wonder', 'B', 'C64', '1', '15.00', 'T']  
['Fruit', 'Apple', 'Bramley', 'G', 'F17', '1', '16.50', 'T']  
['Fruit', 'Apple', 'Cox', 'R', 'F19', '1', '16.30', 'T']  
['Fruit', 'Apple', 'Discovery', 'G', 'F26', '1', '16.25', 'T']  
['Fruit', 'Cherry', 'June', 'R', 'F31', '1', '16.00', 'T']  
['Fruit', 'Pear', 'Conference', 'G', 'F32', '1', '16.00', 'T']  
['Fruit', 'Plum', 'Victoria', 'P', 'F48', '1', '16.25', 'T']  
['Fruit', 'Peach', 'Nancy', 'O', 'F72', '1', '21.0', 'M']  
['Fuchsia', 'Annabel', 'Erne', 'W', 'U17', '6', '6.95', 'S']  
['Fuchsia', 'Darkeyes', 'Bann', 'RW', 'U21', '6', '6.50', 'S']  
['Fuchsia', 'Dancing Flame', 'Lagan', 'R', 'U25', '6', '11.95', 'M']  
['Fuchsia', 'Thalia', 'Strule', 'PR', 'U37', '6', '6.50', 'S']  
['Fuchsia', 'Swingtime', 'Shimna', 'PW', 'U52', '6', '11.25', 'M']  
['Hedge', 'Berberis', 'Barberry', 'Y', 'H25', '5', '40.0', 'M']  
['Hedge', 'Holly', 'Common', 'G', 'H29', '1', '50.00', 'T']  
['Hedge', 'Bamboo', 'Blue Zion', 'B', 'H59', '10', '149.00', 'T']  
['Hedge', 'Pyracantha', 'Orat', 'O', 'H60', '5', '35.0', 'M']  
['Hedge', 'Dog Wood', 'Red Doa', 'R', 'H64', '10', '50.00', 'T']  
['Shrub', 'Acer', 'Purple Mood', 'P', 'S08', '1', '18.00', 'T']  
['Shrub', 'Buddleia', 'White Will', 'W', 'S14', '1', '12.5', 'M']  
['Shrub', 'Hebe', 'Varigata', 'VR', 'S21', '3', '20.50', 'S']  
['Shrub', 'Daphne', 'Purple Haze', 'P', 'S38', '1', '16.50', 'S']  
['Shrub', 'Camellia', 'Sure Red', 'R', 'S60', '1', '18.0', 'M']  
['Shrub', 'Hydrangea', 'Elizabeth', 'B', 'S69', '1', '7.00', 'T']  
['Shrub', 'Magnolia', 'Paul', 'W', 'S79', '1', '20.0', 'M']  
['Shrub', 'Bay Tree', 'Standard', 'G', 'S81', '1', '75.0', 'M']  
['Shrub', 'Bay Tree', 'Pyramid', 'G', 'S83', '1', '40.00', 'S']  
['Shrub', 'Bay Tree', 'Cuboid', 'G', 'S85', '1', '45', 'M']  
  
  
Pick next option: 7  
  
1. Add a new plant  
2. Search for a plant with its reference ID  
3. Increase price of all medium plants by 1  
4. Display all climbers  
5. Delete a plant using its refID  
6. Display all plants  
7. Show options again  
0. Exit program  
  
  
  
Pick next option: 0  
  
Process finished with exit code 0

# Assignment 13

Problem

A list contains following record of customer:

[customer\_name, Room\_Type]

Write the following user defined functions to perform given operations on the stack named Hotel.

1. Push\_Cust() – to Push customers names of those customers who are staying in ‘Delux’ Room\_Type.
2. Pop\_Cust() – To pop the names of customers from the stack and display them. Also display “Underflow” when there are no customers in the stack.
3. Display\_Status() – To display the content of the stack. Also display “Underflow” if the stack is empty.

Solution

details = [  
 ["Siddarth", "Delux"],  
 ["Rahul","Standard"],  
 ['Jerry', "Delux"]  
 ]  
  
stack = []  
  
def push\_cust():  
 for elem in details:  
 if elem[1] == 'Delux':  
 stack.insert(0, elem[0])  
  
def pop\_cust():  
 while stack:  
 print(stack.pop(0))  
 print("Underflow")  
  
def display\_status():  
 for elem in stack:  
 print(elem)  
  
s = """  
Pick an option  
1. push  
2. pop  
3. display  
4. exit  
"""  
  
print(s)  
x = int(input("Enter option: "))  
while x != 4:  
 if x == 1:  
 push\_cust()  
 elif x == 2:  
 pop\_cust()  
 elif x == 3:  
 display\_status()  
 else:  
 print("choose valid option")  
  
 x = int(input("Enter next option: "))

Output

Pick an option  
1. push  
2. pop  
3. display  
4. exit  
  
Enter option: 1  
Enter next option: 3  
Jerry  
Siddarth  
Enter next option: 2  
Jerry  
Siddarth  
Underflow  
Enter next option: 4

# Assignment 14

Problem

A dictionary contains the details of vehicles – { car\_name:Maker}.

Write the following user defined functions to perform given operations on the stack named CAR:

1. PUSH(vehicle) where , vehicle is a dictionary containing details of vehicles – { car\_name:Maker}. The function should push the name of the car manufactured by TATA. (including all the possible cases like Tata, TaTa, etc) to the stack)
2. POP() – To pop the car on the top of the stack and display. If no car, display “Underflow”
3. COUNT() – To return number of cars in the stack. If no car, return ‘Underflow”.
4. DISPLA() – To display all the car names from the stack. If no car, display empty.

Solution

data = {  
 'Santro': 'Hyundai',  
 "Nexon": "TATA",  
 "Safari": "tata"  
 }  
  
car = []  
  
def push(vehicle):  
 for elem in vehicle:  
 if vehicle[elem].lower() == 'tata':  
 car.insert(0, elem)  
  
def pop():  
 if car:  
 print(car.pop(0))  
 else:  
 print("Underflow")  
  
def count():  
 if len(car) == 0:  
 print("Underflow")  
 else:  
 print(len(car))  
  
def display():  
 for elem in car:  
 print(elem)  
  
s = """  
Pick an option from below  
1. PUSH  
2. POP  
3. COUNT  
4. DISPLAY  
5. EXIT  
"""

print(s)  
x = int(input("Enter option: "))  
while x != 5:  
 if x == 1:  
 push(data)  
 elif x == 2:  
 pop()  
 elif x == 3:  
 count()  
 elif x == 4:  
 display()  
 else:  
 print("Invalid option")  
  
 x = int(input("enter next option: "))

Output

Pick an option from below  
1. PUSH  
2. POP  
3. COUNT  
4. DISPLAY  
5. EXIT  
  
Enter option: 1  
enter next option: 3  
2  
enter next option: 4  
Safari  
Nexon  
enter next option: 2  
Safari  
enter next option: 4  
Nexon  
enter next option: 5

# Assignment 15

Problem

A list contains following record of a customer:

[Customer\_name, Phone\_number, City]

Write the following user defined functions to perform given operations on the stack named ‘**status’**:

(i) Push\_element() - To Push an object containing name and Phone number of customers who live in Goa to the stack

(ii) Pop\_element() - To Pop the objects from the stack and display them. Also, display “Stack Empty” when there are no elements in the stack.

Solution

data = [  
 ["Gurdas", "99999999999", "Goa"],  
 ["Julee", "88888888888", "Mumbai"],  
 ["Murugan", "77777777777", "Cochin"],  
 ["Ashmit", "10101010101", "Goa"]  
 ]  
  
status = []  
  
def push():  
 for elem in data:  
 if elem[-1].lower() == 'goa':  
 status.insert(0, elem)  
  
def pop():  
 while status:  
 print(status.pop(0))  
 print("Underflow")  
  
  
s = """"  
Pick an option from the following  
1. PUSH  
2. POP  
3. EXIT  
"""  
x = int(input("Enter option: "))  
while x != 3:  
 if x == 1:  
 push()  
 elif x == 2:  
 pop()  
 else:  
 print("Invalid option")  
  
 x = int(input("Enter next option: "))

Output

Enter option: 1  
Enter next option: 2  
['Ashmit', '10101010101', 'Goa']  
['Gurdas', '99999999999', 'Goa']  
Underflow  
Enter next option: 3

# Assignment 16

Problem

Create a table ORDER, without any constraints based on the data given below.



1. Write an SQL statement to find the total purchase amount for all orders.
2. Write a query that counts the number of salesmen with their order date for each day.
3. Write an SQL statement that counts all orders for a date October 5th, 2012.
4. Write an SQL statement to find the highest purchase amount with their ID, for only those

salesmen whose ID is within the range 5003 and 5008.

1. Write an SQL statement to change the salesman\_id to a number incremented by 1000 for all salesmen(5002 should be 6002, 5005 should be 6005 and so on)
2. Write an SQL statement to display the Purchase amount increased by 3% for those salesmen who have more than 1 order. (Use column Alias)
3. Write the command to add a field SalesmanName with appropriate data type.
4. Write the command to set ord\_no as Primary key.
5. Write the command to display the Total Purchase Amount generated by each salesman.
6. Write the command to display the number of orders and the minimum Purchase Amount in the month of September.

Code and Output

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |
| 9 |  |
| 10 |  |

# Assignment 17

Consider the tables given below and write SQL commands as per the requirement





1. To display names of all staff who are in Sales department having experience less than 9 years and commission more than 700.
2. Display the average salary of staff working in Finance department

(Salary= Basic+Allowance)

1. Update the Basic and Allowance of Aryan to 13000 and 55000 respectively.
2. Display the number of staff with a Basic Salary more than 30000, along with their Department.
3. Display the number of departments with more than 1 male staff.
4. Display the minimum salary and maximum commission earned by any staff in the company.
5. Display the number of departments with only female teachers in Sales Department

|  |  |
| --- | --- |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |

# Assignment 18

Problem

Golden key Hospitals owns hospitals around the world. Robin Sharma is the owner of the company.

Robin has collected information about the hospitals and their customers. The **PATIENT** table contains details of customers from a region in England.

The structure of the **PATIENT** table is:



**GENDER**

Create the PATIENT table with the above details.

Write a python connectivity program to perform the following tasks.

1. Insert 5 records into the PATIENT Table

Sample record:



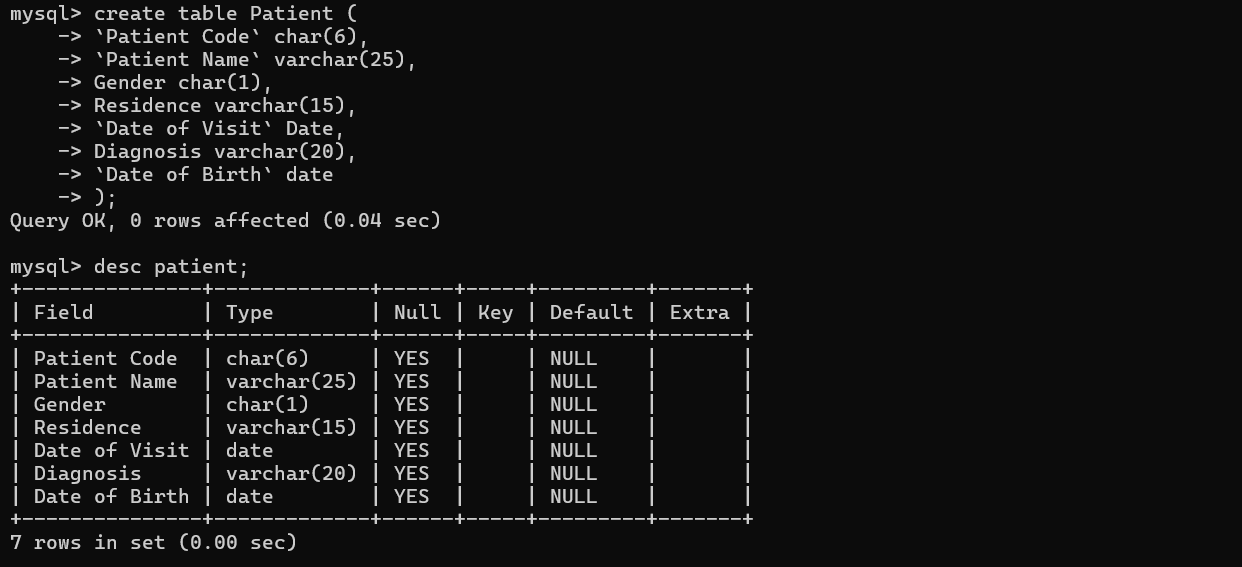
**GENDER**

1. Sort AND DISPLAY the **PATIENT** table into alphabetical order by **PATIENT NAME**
2. Robin wants a list of Patients who are females. He wants all fields displayed. Create a query on the **PATIENT** table to find this information.

4)Robin wants a list of patients whose Date of birth is between 1 April 1987 and 30 April 1987.

Solutions and Output

Creating table:

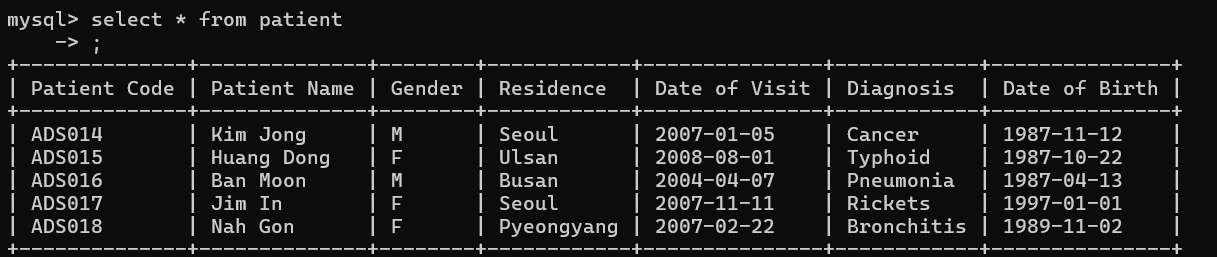


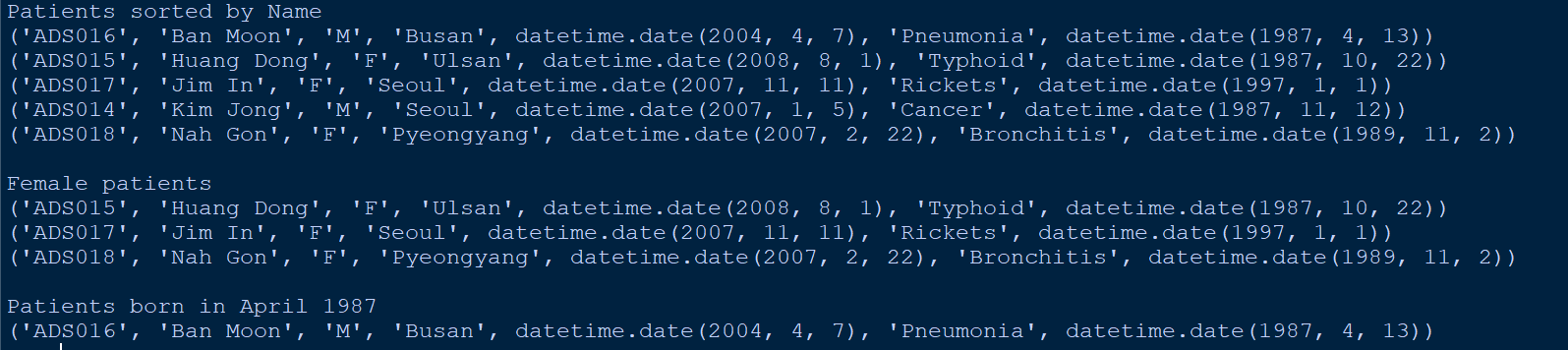
Code

import mysql.connector  
  
connection = mysql.connector.connect(host='localhost', user='root', passwd='Sabya19sachi05', database='cs\_practicals')  
cursor = connection.cursor()  
  
# Inserting records  
data = [  
["ADS014", "Kim Jong", "M", "Seoul", "2007/01/05", "Cancer", "1987/11/12"],  
["ADS015", "Huang Dong", "F", "Ulsan", "2008/08/01", "Typhoid", "1987/10/22"],  
["ADS016", "Ban Moon", "M", "Busan", "2004/04/07", "Pneumonia", "1987/04/13"],  
["ADS017", "Jim In", "F", "Seoul", "2007/11/11", "Rickets", "1997/01/01"],  
["ADS018", "Nah Gon", "F", "Pyeongyang", "2007/02/22", "Bronchitis", "1989/11/02"]  
 ]  
  
for row in data:  
 cursor.execute("insert into patient values ('%s', '%s', '%s', '%s', '%s', '%s', '%s');"%tuple(row))  
  
connection.commit()  
  
#Sort and display  
print("Patients sorted by Name")  
cursor.execute("select \* from patient ORDER BY `Patient Name`;")  
  
for row in cursor.fetchall():  
 print(row)  
  
print()  
  
#Select Female Patients  
print("Female patients")  
cursor.execute("select \* from patient where Gender = 'F';")  
for row in cursor.fetchall():  
 print(row)  
  
print()  
  
#select patients born in april of 1987  
cursor.execute("select \* from patient where `Date of Birth` between '1987/04/01' and '1987/04/30';")  
print("Patients born in April 1987")  
for row in cursor.fetchall():  
 print(row)

connection.close()

Output





# Assignment 19

Problem

Simon is the owner of *Celtic Springs Scuba (CS Scuba).* The company teaches diving and

offers diving trips around the world. Simon has collected information about the holidays. He wants to put the holidays information in a table DESTINATIONS.

The structure of the **DESTINATIONS** table is:



**Create the DESTINATIONS table using the above details and insert five records.**

Simon wants a MySql python connectivity program to perform the following tasks:

1. Add these details to the **DESTINATIONS** table.

****

1. Display the records of the **DESTINATIONS** table sort by **COUNTRY** into alphabetical order.
2. Simon wants a list of holidays that:

• are liveaboard

• are 7 days duration

• use Blue Water dive boats.

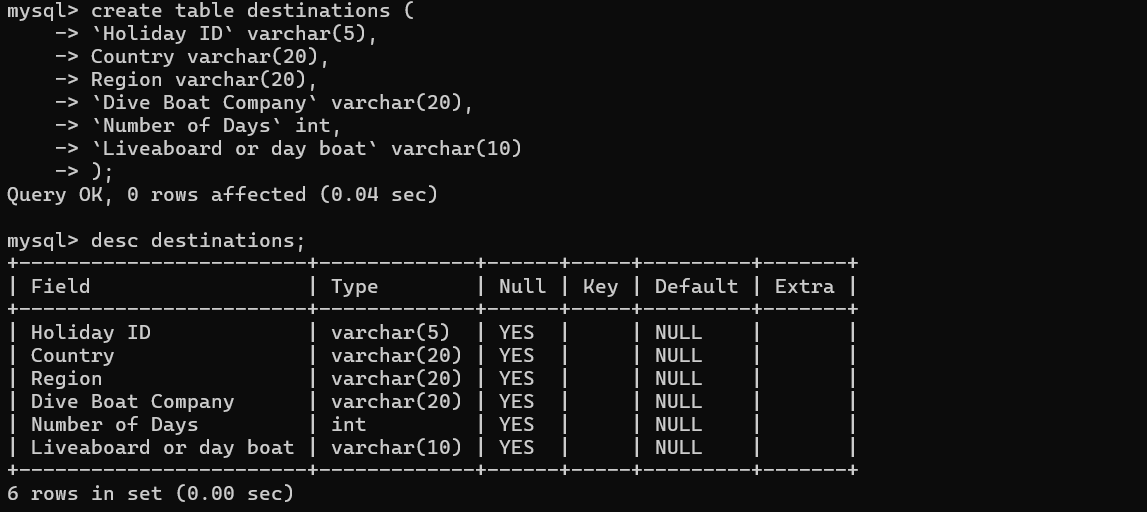
The list must:

• show only these fields in this order **Holiday ID, Region, Country**

• be sorted in descending order of **Holiday ID**

**Solutions and Output**

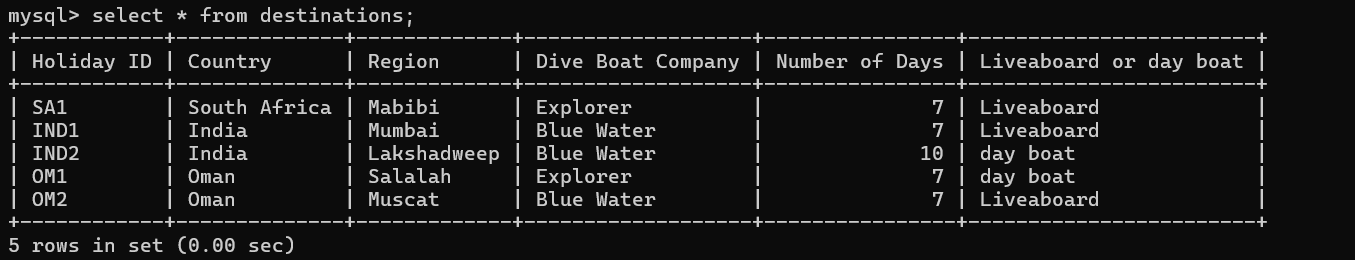
**Creating table**

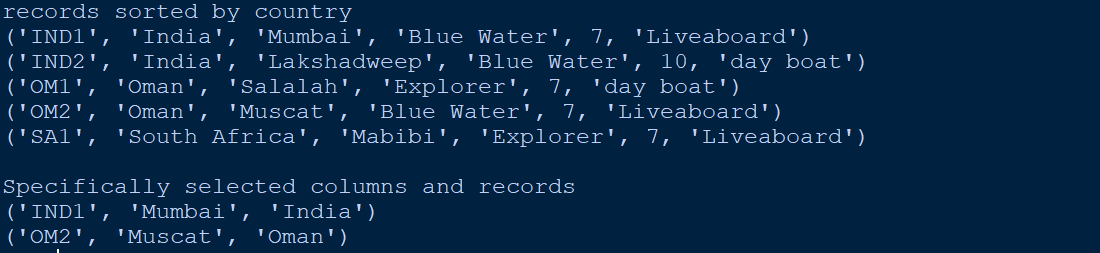
****

**Code**

import mysql.connector  
  
connection = mysql.connector.connect(host='localhost', user='root', passwd='Sabya19sachi05', database='cs\_practicals')  
cursor = connection.cursor()  
  
# Inserting records  
data = [  
 ['SA1', 'South Africa', 'Mabibi', 'Explorer', 7, 'Liveaboard'],  
 ['IND1', 'India', 'Mumbai', 'Blue Water', 7, 'Liveaboard'],  
 ['IND2', 'India', 'Lakshadweep', 'Blue Water', 10, 'day boat'],  
 ['OM1', 'Oman', 'Salalah', 'Explorer', 7, 'day boat'],  
 ['OM2', 'Oman', 'Muscat', 'Blue Water', 7, 'Liveaboard']  
 ]  
  
for row in data:  
 cursor.execute("insert into destinations values ('%s', '%s', '%s', '%s', %s, '%s');"%tuple(row))  
  
connection.commit()  
  
#Display records sorted by country  
print("records sorted by country")  
cursor.execute("select \* from destinations order by country;")  
for row in cursor.fetchall():  
 print(row)  
  
print()  
  
#Q3: specific fields and records  
print("Specifically selected columns and records")  
command = "select `Holiday ID`, Region, Country from destinations"  
command += " where `Liveaboard or day boat`='Liveaboard' AND `Number of days`=7 AND `Dive Boat Company`='Blue Water'"  
command += " ORDER BY `Holiday ID`;"  
cursor.execute(command)  
for row in cursor.fetchall():  
 print(row)  
  
connection.close()

**Output**

****

****

# Assignment 20

Problem

**ABC infotech Pvt.Ltd. needs to store, retrieve and delete the records of its employees, Develop an interface that provides front-end interaction through python , and stores and updates records using MYSQL.**

**The operations on MYSQL table “EMP” involving Adding, reading, searching, updating and deleting records of employees as per user’s request.**

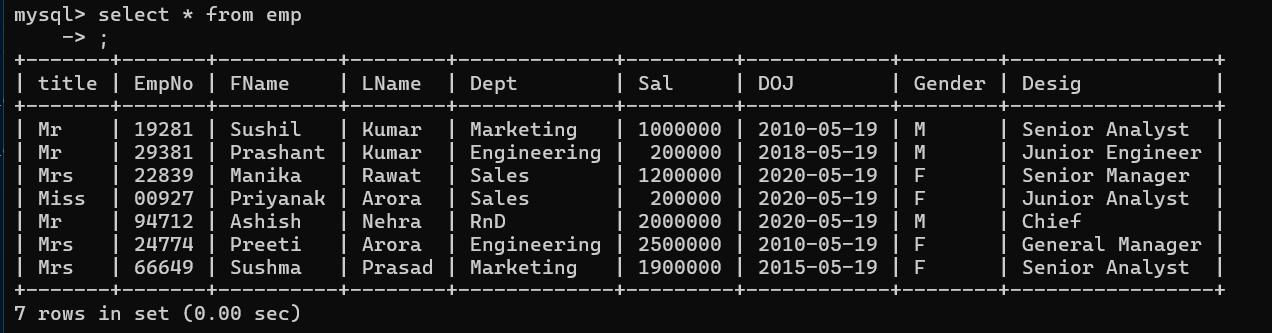
**Following is the structure of EMP table:**

|  |  |
| --- | --- |
| **Field Name** | **Description** |
| **Title** | **Mr or Mrs or Miss** |
| **EmpNo** | **Employee Number** |
| **FName** | **First Name** |
| **LName** | **Last Name** |
| **Dept** | **Department Name** |
| **Sal** | **Salary** |
| **DOJ** | **Date of Join** |
| **Gender** | **M or F** |
| **Desig** | **Designation** |

Solution and Output

Table creation with dummy values

import mysql.connector  
  
connection = mysql.connector.connect(host='localhost', user='root', passwd='Sabya19sachi05', database='cs\_practicals')  
cursor = connection.cursor()  
  
# Inserting dummy records  
data = [  
 ["Mr", '19281', 'Sushil', 'Kumar', 'Marketing', 1000000, '2010/05/19', 'M', 'Senior Analyst'],  
 ["Mr", '29381', 'Prashant', 'Kumar', 'Engineering', 200000, '2018/05/19', 'M', 'Junior Engineer'],  
 ["Mrs", '22839', 'Manika', 'Rawat', 'Sales', 1200000, '2020/05/19', 'F', 'Senior Manager'],  
 ["Miss", '00927', 'Priyanak', 'Arora', 'Sales', 200000, '2020/05/19', 'F', 'Junior Analyst'],  
 ["Mr", '94712', 'Ashish', 'Nehra', 'RnD', 2000000, '2020/05/19', 'M', 'Chief'],  
 ["Mrs", '24774', 'Preeti', 'Arora', 'Engineering', 2500000, '2010/05/19', 'F', 'General Manager'],  
 ["Mrs", '66649', 'Sushma', 'Prasad', 'Marketing', 1900000, '2015/05/19', 'F', 'Senior Analyst']  
 ]  
  
for row in data:  
 cursor.execute("insert into emp values ('%s', '%s', '%s', '%s', '%s', %s, '%s', '%s', '%s');"%tuple(row))  
  
connection.commit()  
  
connection.close()



Python Menu Based program to alter data

import mysql.connector  
  
connection = mysql.connector.connect(host='localhost', user='root', passwd='Sabya19sachi05', database='cs\_practicals')  
cursor = connection.cursor()  
  
## Prompts to help make usage easier  
prompt = """  
Choose what you want to do  
  
1 - Display all records  
2 - Read a record by employee number  
3 - Add a new record  
4 - Delete a record by employee number  
5 - Update an existing record  
0 - Exit program  
"""  
  
prompt2 = """  
Enter the following details in order, separated by commas.  
  
Title, employee number, first name, last name, department, salary, date of joining(YYYY/MM/DD), gender (M or F), Designation  
  
Example  
Mr,29373,Sushil,Kumar,Marketing,10000,2010/04/19,M,Senior Analyst  
"""  
  
prompt3 = """  
Pick the fields you wish to update  
  
Fields available:  
1: Title  
2: Employee Number  
3: First Name  
4: Last Name  
5: Department  
6: Salary  
7: Date of Joinint  
8: Gender  
9: Designation  
  
Input fields you wish to update as the corresponding in single line, separated by commas.  
Eg: 1,3,4,8  
"""  
  
field\_dict = {  
 '1': 'title',  
 '2': 'empno',  
 '3': 'fname',  
 '4': 'lname',  
 '5': 'dept',  
 '6': 'sal',  
 '7': 'doj',  
 '8': 'gender',  
 '9': 'desig'  
 }  
  
#Main Interface Loop  
while True:  
   
 print(prompt)  
 choice = int(input("Enter number of what you want to do: "))  
  
 #Choice 1 code: display full table  
 if choice == 1:  
 cursor.execute("select \* from emp;")  
 for row in cursor.fetchall():  
 print(row)  
 print()  
  
 #Choice 2 code: find specific record by EmpNO  
 elif choice == 2:  
 num = input("Enter unique 5 digit employee number: ").strip()  
 cursor.execute("select \* from emp where EmpNo = '%s';"%(num,))  
 print(cursor.fetchone())  
  
 #Choice 3 code: Insert new record  
 elif choice == 3:  
 print(prompt2)  
 data = input().split(",")  
 data[5] = int(data[5])  
 cursor.execute("insert into emp values ('%s', '%s', '%s', '%s', '%s', %s, '%s', '%s', '%s');"%tuple(data))  
 connection.commit()  
 print("New record")  
 cursor.execute(f"select \* from emp where empno={data[1]};")  
  
 #Choice 4 code: Delete record by EmpNo  
 elif choice == 4:  
 num = input("Enter employee number to delete: ").strip()  
 cursor.execute("delete from emp where EmpNo = '%s';"%(num,))  
 connection.commit()  
 print("Record Deleted. Display all records to see change.")  
  
 # Choice 5 code: Update record by EmpNo  
 elif choice == 5:  
 num = input("Enter employee number to update: ").strip()  
 print(prompt3)  
 fields = input().split(",")  
 command = "update emp set"  
 for f in fields:  
 command += " " + field\_dict[f] + " = "  
 if f != '6':  
 command += "'" + input(f"Enter new value for {f}: ") + "',"  
 else:  
 command += input(f"Enter new value for {f}: ") + ","  
  
 command = command[:-1]  
 command += f" where empno = '{num}';";  
 print(command)  
 cursor.execute(command)  
 connection.commit()  
 print("Data updated. Display all records to see change")  
  
 # Choice 0 code: exit loop  
 elif choice == 0:

print(“Exiting Program”)  
 break;  
  
 # Invalid choices  
 else:  
 print("Enter valid choice\n")

Output

Choose what you want to do

1 - Display all records

2 - Read a record by employee number

3 - Add a new record

4 - Delete a record by employee number

5 - Update an existing record

0 - Exit program

Enter number of what you want to do: 1

('Mr', '30482', 'Sushil', 'Kumar', 'Marketing', 1100000, datetime.date(2010, 5, 19), 'M', 'Senior Analyst')

('Mr', '29381', 'Prashant', 'Kumar', 'Engineering', 200000, datetime.date(2018, 5, 19), 'M', 'Junior Engineer')

('Mrs', '22839', 'Manika', 'Rawat', 'Sales', 1200000, datetime.date(2020, 5, 19), 'F', 'Senior Manager')

('Miss', '00927', 'Priyanak', 'Arora', 'Sales', 200000, datetime.date(2020, 5, 19), 'F', 'Junior Analyst')

('Mr', '94712', 'Ashish', 'Nehra', 'RnD', 2000000, datetime.date(2020, 5, 19), 'M', 'Chief')

('Mrs', '24774', 'Preeti', 'Arora', 'Engineering', 2500000, datetime.date(2010, 5, 19), 'F', 'General Manager')

('Mrs', '66649', 'Sushma', 'Prasad', 'Marketing', 1900000, datetime.date(2015, 5, 19), 'F', 'Senior Analyst')

Choose what you want to do

1 - Display all records

2 - Read a record by employee number

3 - Add a new record

4 - Delete a record by employee number

5 - Update an existing record

0 - Exit program

Enter number of what you want to do: 2

Enter unique 5 digit employee number: 29381

('Mr', '29381', 'Prashant', 'Kumar', 'Engineering', 200000, datetime.date(2018, 5, 19), 'M', 'Junior Engineer')

Choose what you want to do

1 - Display all records

2 - Read a record by employee number

3 - Add a new record

4 - Delete a record by employee number

5 - Update an existing record

0 - Exit program

Enter number of what you want to do: 5

Enter employee number to update: 00927

Pick the fields you wish to update

Fields available:

1: Title

2: Employee Number

3: First Name

4: Last Name

5: Department

6: Salary

7: Date of Joinint

8: Gender

9: Designation

Input fields you wish to update as the corresponding in single line, separated by commas.

Eg: 1,3,4,8

6

Enter new value for 6: 220000

update emp set sal = 220000 where empno = '00927';

Data updated. Display all records to see change

Choose what you want to do

1 - Display all records

2 - Read a record by employee number

3 - Add a new record

4 - Delete a record by employee number

5 - Update an existing record

0 - Exit program

Enter number of what you want to do: 4

Enter employee number to delete: 66649

Record Deleted. Display all records to see change.

Choose what you want to do

1 - Display all records

2 - Read a record by employee number

3 - Add a new record

4 - Delete a record by employee number

5 - Update an existing record

0 - Exit program

Enter number of what you want to do: 3

Enter the following details in order, separated by commas.

Title, employee number, first name, last name, department, salary, date of joining(YYYY/MM/DD), gender (M or F), Designation

Example

Mr,29373,Sushil,Kumar,Marketing,10000,2010/04/19,M,Senior Analyst

Mrs,39201,Swara,Rao,RnD,120000,2008/02/22,F,Senior Researcher

New record

Choose what you want to do

1 - Display all records

2 - Read a record by employee number

3 - Add a new record

4 - Delete a record by employee number

5 - Update an existing record

0 - Exit program

Enter number of what you want to do: 1

('Mr', '30482', 'Sushil', 'Kumar', 'Marketing', 1100000, datetime.date(2010, 5, 19), 'M', 'Senior Analyst')

('Mr', '29381', 'Prashant', 'Kumar', 'Engineering', 200000, datetime.date(2018, 5, 19), 'M', 'Junior Engineer')

('Mrs', '22839', 'Manika', 'Rawat', 'Sales', 1200000, datetime.date(2020, 5, 19), 'F', 'Senior Manager')

('Miss', '00927', 'Priyanak', 'Arora', 'Sales', 220000, datetime.date(2020, 5, 19), 'F', 'Junior Analyst')

('Mr', '94712', 'Ashish', 'Nehra', 'RnD', 2000000, datetime.date(2020, 5, 19), 'M', 'Chief')

('Mrs', '24774', 'Preeti', 'Arora', 'Engineering', 2500000, datetime.date(2010, 5, 19), 'F', 'General Manager')

('Mrs', '39201', 'Swara', 'Rao', 'RnD', 120000, datetime.date(2008, 2, 22), 'F', 'Senior Researcher')

Choose what you want to do

1 - Display all records

2 - Read a record by employee number

3 - Add a new record

4 - Delete a record by employee number

5 - Update an existing record

0 - Exit program

Enter number of what you want to do: 0

Exiting Program