**Case Study #1: Student Placement System using Dictionaries:** You are tasked with building a Student Placement System for a university using Python, with a focus on utilizing dictionaries. Implement the following functionalities:

**1.**      **Add Students:**

a.       Create an empty dictionary to represent student details.

b.      Allow the user to add students by entering details such as student ID, name, department, and skills.

c.       Use the dictionary to store student IDs as keys and their details as values.

**2.**      **Display Students:**

a.       Provide an option for the user to display the list of all students along with their details.

b.      Display each student's ID, name, department, and skills in a readable format.

**3.**      **Search for Students by Department:**

a.       Implement a search functionality where the user can enter a department to find all students in that department.

b.      If students are found, display their details.

c.       If no students are found, inform the user.

**4.**      **Update Student Skills:**

a.       Allow the user to update the skills of a specific student by entering their ID.

b.      If the student is found, prompt the user to update the skills.

c.       If the student is not found, inform the user.

**5.**      **Remove Student:**

a.       Implement a feature to remove a specific student from the placement system by entering their ID.

b.      If the student is found, remove their information from the dictionary.

c.       If the student is not found, inform the user.

**6.**      **Generate Placement Reports:**

a.       Provide an option to generate placement reports for students.

b.      Use the skills information to identify potential job opportunities for students.

**7.**      **Exit System:**

a.       Allow the user to exit the system.

PROGRAM:

l = {}

eligibility = {

   "java":"software engineer",

   "python":"software engineer",

   "sql":"software engineer",

   "js":"software engineer",

   "networking":"networks engineer",

   "cloud computing":"cloud engineer",

   "marketing":"sales executive",

}

def add():

  id = input("Enter student id: ")

  dn = input("Enter name: ")

  dt = input("Enter department: ")

  di = input("Enter skills: ")

  l[id]={"name":dn,"department":dt,"skills":di.split(',')}

def show():

  print("students :")

  print("ID\tname\tdepartment\tskills")

  for i in l: print(i,"\t",l[i]["name"],"\t",l[i]["department"],"\t",l[i]["skills"])

def search():

  dn = input("Enter the department to search: ")

  found=False

  for i in l:

    if l[i]["department"]==dn:

      print(i,"\t",l[i]["name"],"\t",l[i]["department"],"\t",l[i]["skills"])

      found=True

  if found==False:

    print("student not found")

def update():

  dn = input("Enter the student ID to update: ")

  found=False

  if dn in l:

      sk = input("Enter new skills")

      l[dn]["skills"]=sk.split(',')

      print("Student skills updated successfully!")

  else:

        print("student not found")

def remove():

  dn = input("Enter the student ID to remove: ")

  found=False

  if dn in l:

      l.pop(dn)

      print("Student removed from the placement system.")

  else:

        print("student not found")

def reports():

  print("Placement Report for Computer Science Students:")

  for i in l:

     jobs = set()

     for skill in l[i]["skills"]:

        jobs.add(eligibility[skill])

     print("- ",l[i]["name"],"is eligible for ",str(jobs) ,"positions.")

while(True):

  print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

  print("Welcome to the Student Placement System!")

  print("1. Add students")

  print("2. display students")

  print("3. Search for Students by Department")

  print("4. Update Student Skills")

  print("5. remove student")

  print("6. Generate Placement Reports")

  print("7. exit")

  c = int(input("Enter ur choice: "))

  if(c==1):

    add()

  elif c==2:

    show()

  elif c==3:

    search()

  elif c==4:

    update()

  elif c==5:

    remove()

  elif c==6:

     reports()

  elif c==7:

    # exit

    print("Exit")

    break

  else:

    print("invalid input")

OUTPUT:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Add students

2. display students

3. Search for Students by Department

4. Update Student Skills

5. remove student

6. Generate Placement Reports

7. exit

Enter ur choice: 1

Enter student id: 1

Enter name: charan

Enter department: cse

Enter skills: java,sql

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Add students

2. display students

3. Search for Students by Department

4. Update Student Skills

5. remove student

6. Generate Placement Reports

7. exit

Enter ur choice: 1

Enter student id: 2

Enter name: sandeep

Enter department: ece

Enter skills: networking

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Add students

2. display students

3. Search for Students by Department

4. Update Student Skills

5. remove student

6. Generate Placement Reports

7. exit

Enter ur choice: 2

students :

ID name department skills

1 charan cse ['java', 'sql']

2 sandeep ece ['networking']

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Add students

2. display students

3. Search for Students by Department

4. Update Student Skills

5. remove student

6. Generate Placement Reports

7. exit

Enter ur choice: 3

Enter the department to search: cse

1 charan cse ['java', 'sql']

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Add students

2. display students

3. Search for Students by Department

4. Update Student Skills

5. remove student

6. Generate Placement Reports

7. exit

Enter ur choice: 4

Enter the student ID to update: 2

Enter new skillsjava,networking

Student skills updated successfully!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Add students

2. display students

3. Search for Students by Department

4. Update Student Skills

5. remove student

6. Generate Placement Reports

7. exit

Enter ur choice: 2

students :

ID name department skills

1 charan cse ['java', 'sql']

2 sandeep ece ['java', 'networking']

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Add students

2. display students

3. Search for Students by Department

4. Update Student Skills

5. remove student

6. Generate Placement Reports

7. exit

Enter ur choice: 6

Placement Report for Computer Science Students:

- charan is eligible for {'software engineer'} positions.

- sandeep is eligible for {'networks engineer', 'software engineer'} positions.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Add students

2. display students

3. Search for Students by Department

4. Update Student Skills

5. remove student

6. Generate Placement Reports

7. exit

Enter ur choice: 5

Enter the student ID to remove: 2

Student removed from the placement system.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Add students

2. display students

3. Search for Students by Department

4. Update Student Skills

5. remove student

6. Generate Placement Reports

7. exit

Enter ur choice: 2

students :

ID name department skills

1 charan cse ['java', 'sql']

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Add students

2. display students

3. Search for Students by Department

4. Update Student Skills

5. remove student

6. Generate Placement Reports

7. exit

Enter ur choice: 7

Exit

**Case Study #2: Analyzing Student Grades using Lambda, Map, Filter, and Reduce Functions:** You are tasked with building a program to analyze student grades using Python, with a focus on utilizing lambda, map, filter, and reduce functions. Implement the following functionalities:

**1.**      **Enter Student Grades:**

a.       Create a list to represent the grades of students in a course.

b.      Allow the user to enter the grades for each student.

**2.**      **Display Student Grades:**

a.       Provide an option for the user to display the list of all student grades.

**3.**      **Calculate Grade Points:**

a.       Implement a lambda function to calculate the grade points for each student based on their grades.

b.      Use the map function to apply the lambda function to each student's grade.

**4.**      **Filter Students:**

a.       Implement a lambda function and the filter function to identify students who need additional assistance (e.g., students with grades below a certain threshold).

**5.**      **Calculate Average Grade:**

a.       Implement a lambda function and the reduce function to calculate the average grade of all students.

**6.**      **Exit System:**

a.       Allow the user to exit the system.

PROGRAM:

from functools import reduce

l = []

def add():

  l.extend(list(map(int,input("Enter the grades for each student (comma-separated): ").split(','))))

  print("Student grades entered successfully!")

def show():

  print("Student Grades:")

  print(l)

def calc():

  grade = lambda x: 5.0 if x>=95 else 4.0 if x>=90 else 3.0 if x>=80 else 2.0 if x>=70 else 1.0 if x>=60 else 0.0

  r = list(map(grade,l))

  print("Grade Points Calculated:")

  print(r)

def filter\_students():

  n = float(input("Enter the threshold grade for students needing assistance: "))

  print("Students Needing Assistance: ")

  grade = lambda x: 5.0 if x>=95 else 4.0 if x>=90 else 3.0 if x>=80 else 2.0 if x>=70 else 1.0 if x>=60 else 0.0

  r = list(map(grade,l))

  r = list(filter(lambda x : x<n , r))

  print(r)

def average():

  print("Average Marks: ", reduce(lambda x,y:x+y, l)/len(l))

  grade = lambda x: 5.0 if x>=95 else 4.0 if x>=90 else 3.0 if x>=80 else 2.0 if x>=70 else 1.0 if x>=60 else 0.0

  r = list(map(grade,l))

  print("Average Grade: ", reduce(lambda x,y:x+y, r)/len(l))

while(True):

  print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

  print("Welcome to the Student Grade Analysis System!")

  print("1. Enter Student Grades")

  print("2. Display Student Grades")

  print("3. Calculate Grade Points")

  print("4. Filter Students Needing Assistance")

  print("5. Calculate Average Grade")

  print("6. exit")

  c = int(input("Enter ur choice: "))

  if(c==1):

    add()

  elif c==2:

    show()

  elif c==3:

    calc()

  elif c==4:

    filter\_students()

  elif c==5:

    average()

  elif c==6:

    print("Exit")

    break

  else:

    print("invalid input")

OUTPUT:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Grade Analysis System!

1. Enter Student Grades

2. Display Student Grades

3. Calculate Grade Points

4. Filter Students Needing Assistance

5. Calculate Average Grade

6. exit

Enter ur choice: 1

Enter the grades for each student (comma-separated): 85, 90, 75, 60, 92

Student grades entered successfully!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Enter Student Grades

2. Display Student Grades

3. Calculate Grade Points

4. Filter Students Needing Assistance

5. Calculate Average Grade

6. exit

Enter ur choice: 2

Student Grades:

[85, 90, 75, 60, 92]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Enter Student Grades

2. Display Student Grades

3. Calculate Grade Points

4. Filter Students Needing Assistance

5. Calculate Average Grade

6. exit

Enter ur choice: 3

Grade Points Calculated:

[3.0, 4.0, 2.0, 1.0, 4.0]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Enter Student Grades

2. Display Student Grades

3. Calculate Grade Points

4. Filter Students Needing Assistance

5. Calculate Average Grade

6. exit

Enter ur choice: 4

Enter the threshold grade for students needing assistance: 3.0

Students Needing Assistance:

[2.0, 1.0]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Enter Student Grades

2. Display Student Grades

3. Calculate Grade Points

4. Filter Students Needing Assistance

5. Calculate Average Grade

6. exit

Enter ur choice: 5

Average Marks: 80.4

Average Grade: 2.8

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Student Placement System!

1. Enter Student Grades

2. Display Student Grades

3. Calculate Grade Points

4. Filter Students Needing Assistance

5. Calculate Average Grade

6. exit

Enter ur choice: 6

Exit

**Case Study #3: Restaurant Order Management System with Variable-Length Arguments:** You are tasked with building a Restaurant Order Management System using Python, highlighting the utility of variable-length arguments. Implement the following functionalities:

**1.**      **Place Order:**

a.       Create a function that accepts variable-length arguments to represent items ordered by a customer.

b.      Allow the user to place an order by specifying the item name, quantity, and price for each item.

**2.**      **Display Order Details:**

a.       Create a function that displays the details of the customer's order.

b.      Display each item's name, quantity, and total cost in a readable format.

**3.**      **Calculate Total Cost:**

a.       Create a function that calculates and displays the total cost of the customer's order.

b.      Use variable-length arguments to handle a variable number of items.

**4.**      **Remove Items from Order:**

a.       Create a function that removes specific items from the order based on the item name.

b.      Use variable-length arguments to handle a variable number of items to remove.

**5.**      **Exit System:**

a.       Allow the user to exit the system.

PROGRAM :

from functools import reduce

l = []

def order(\*item):

  l.append([item[0],int(item[1]),float(item[2])])

  print("Item added to the order successfully!")

def show():

  print("Order Details:")

  for i in l: print("Item: ",i[0],", Quantity: ",i[1]," Total Cost: $",i[2])

def calc(\*items):

  print("Total Cost of the Order: $", reduce(lambda x,y: x+y[1]\*y[2] , l, 0))

def remove(\*items):

    item = input("Enter the item name to remove from the order: ")

    removed=False

    for i in items:

        if i[0]==item:

            l.remove(i)

            print("Item removed from the order.")

            removed=True

    if removed==False:

        print("Item Not Found")

while(True):

  print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

  print("Welcome to the Restaurant Order Management System!")

  print("1. Place Order")

  print("2. Display Order Details")

  print("3. Calculate Total Cost")

  print("4. Remove Items from Order")

  print("5. exit")

  c = int(input("Enter ur choice: "))

  if(c==1):

    i = input("Enter item details (name, quantity, price - comma-separated): ").split(',')

    # list expansion/destructuring

    order(\*i)

  elif c==2:

    show()

  elif c==3:

    calc(\*l)

  elif c==4:

    remove(\*l)

  elif c==5:

    print("Exit")

    break

  else:

    print("invalid input")

OUTPUT:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Restaurant Order Management System!

1. Place Order

2. Display Order Details

3. Calculate Total Cost

4. Remove Items from Order

5. exit

Enter ur choice: 1

Enter item details (name, quantity, price - comma-separated): Burger,2,5.99

Item added to the order successfully!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Restaurant Order Management System!

1. Place Order

2. Display Order Details

3. Calculate Total Cost

4. Remove Items from Order

5. exit

Enter ur choice: 1

Enter item details (name, quantity, price - comma-separated): Soda,1,1.99

Item added to the order successfully!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Restaurant Order Management System!

1. Place Order

2. Display Order Details

3. Calculate Total Cost

4. Remove Items from Order

5. exit

Enter ur choice: 2

Order Details:

Item: Burger , Quantity: 2 Total Cost: $ 5.99

Item: Soda , Quantity: 1 Total Cost: $ 1.99

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Restaurant Order Management System!

1. Place Order

2. Display Order Details

3. Calculate Total Cost

4. Remove Items from Order

5. exit

Enter ur choice: 3

Total Cost of the Order: $ 13.97

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Restaurant Order Management System!

1. Place Order

2. Display Order Details

3. Calculate Total Cost

4. Remove Items from Order

5. exit

Enter ur choice: 4

Enter the item name to remove from the order: Soda

Item removed from the order.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Restaurant Order Management System!

1. Place Order

2. Display Order Details

3. Calculate Total Cost

4. Remove Items from Order

5. exit

Enter ur choice: 2

Order Details:

Item: Burger , Quantity: 2 Total Cost: $ 5.99

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the Restaurant Order Management System!

1. Place Order

2. Display Order Details

3. Calculate Total Cost

4. Remove Items from Order

5. exit

Enter ur choice: 5

Exit