Practical no:06

Expert System: Employee performance evaluation

Program:

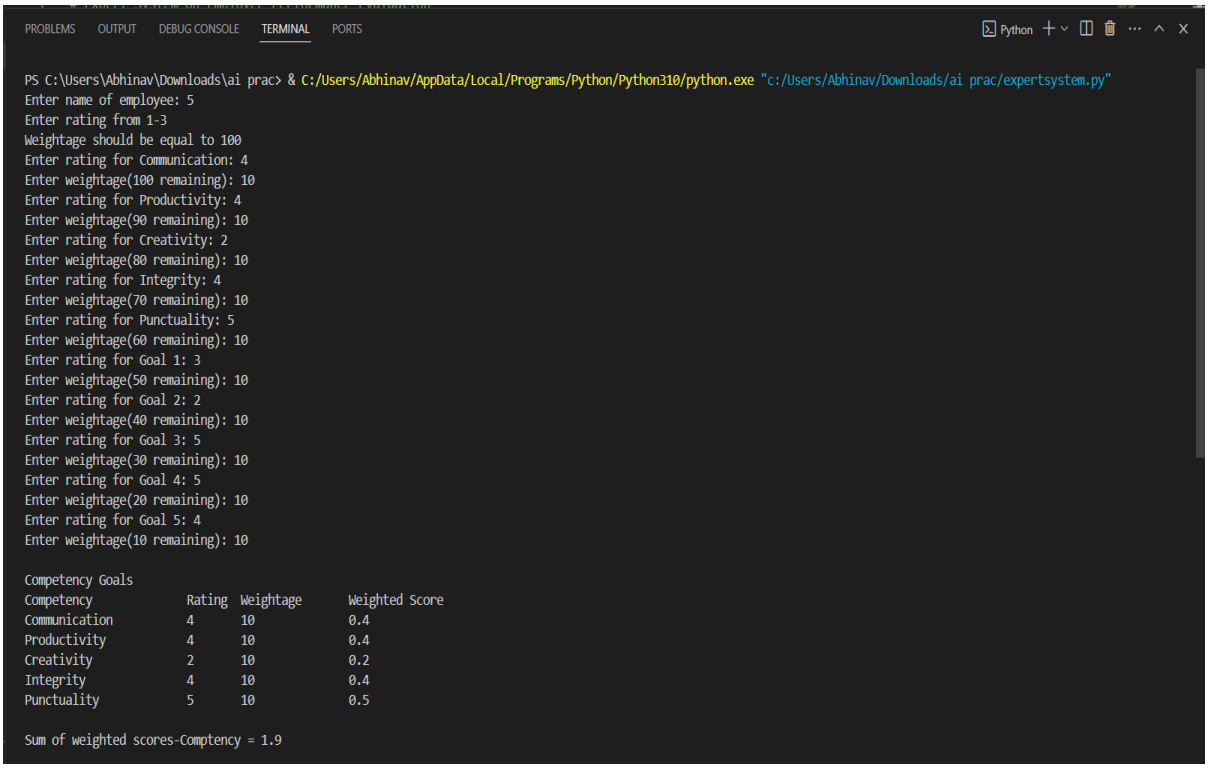
```
# Expert System on Employee Performance Evaluation
class Evaluation:
    def __init__(self) -> None:
        self.name = input("Enter name of employee: ")
        self.competencies = {
            "Communication" : [0,0,0],
            "Productivity" : [0,0,0],
            "Creativity" : [0,0,0],
            "Integrity" : [0,0,0],
            "Punctuality" : [0,0,0]
        }
        self.performance = {
            "Goal 1" : [0,0,0],
            "Goal 2" : [0,0,0],
            "Goal 3" : [0,0,0],
            "Goal 4" : [0,0,0],
            "Goal 5" : [0,0,0]
        }



    def printTable(self,hashMap : dict):
        if hashMap == self.competencies:
            print("Competency Goals")
```

```
        print("Competency\t\tRating\tWeightage\tWeighted Score")
        for key,value in self.competencies.items():
            print(f"{key}\t\t{value[0]}\t{value[1]}\t\t{value[2]}")
        print()
    else:
        print("Performance Goals")
        print("Goals\t\tRating\tWeightage\tWeighted Score")
        for key,value in self.performance.items():
            print(f"{key}\t\t{value[0]}\t{value[1]}\t\t{value[2]}")
        print()


  def input(self):
    print("Enter rating from 1-3")
    print("Weightage should be equal to 100")
    weightTotal = 0
    for key in self.competencies.keys():
        self.competencies[key][0] = int(input(f"Enter rating for {key}: "))
        self.competencies[key][1] = int(input(f"Enter weightage({100 - weightTotal}
remaining): "))
        weightTotal += self.competencies[key][1]
    for key in self.performance.keys():
        self.performance[key][0] = int(input(f"Enter rating for {key}: "))
        self.performance[key][1] = int(input(f"Enter weightage({100 - weightTotal}
remaining): "))
        weightTotal += self.performance[key][1]


  def calcScore(self):
```

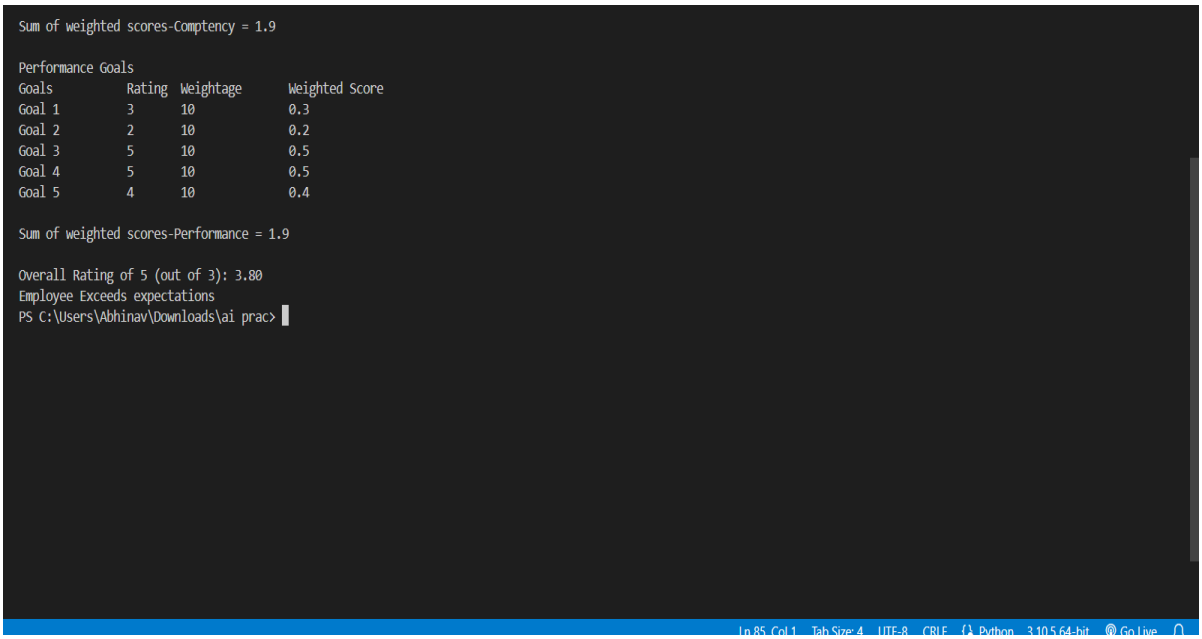
```
        for key in self.competencies.keys():
            self.competencies[key][2] = self.competencies[key][0] *
self.competencies[key][1] / 100
        for key in self.performance.keys():
            self.performance[key][2] = self.performance[key][0] *
self.performance[key][1] / 100


    def calculate(self):
        self.input()
        print()
        self.calcScore()
        self.printTable(self.competencies)


        sumCompetancy = 0
        for key in self.competencies.keys():
            sumCompetancy += self.competencies[key][2]
        print(f"Sum of weighted scores-Comptency = {sumCompetancy}")
        print()
        sumPerformance = 0
        self.printTable(self.performance)
        for key in self.performance.keys():
            sumPerformance += self.performance[key][2]
        print(f"Sum of weighted scores-Performance = {sumPerformance}")
        print()
        total = (sumCompetancy + sumPerformance)


        if total >= 2.7:
```

```
            total = "{:.2f}".format(total)
            print(f"Overall Rating of {self.name} (out of 3): {total}")
            print("Employee Exceeds expectations")
        elif total >= 1.7 and total < 2.7:
            total = "{:.2f}".format(total)
            print(f"Overall Rating of {self.name} (out of 3): {total}")
            print("Employee meets expectations")
        else:
            total = "{:.2f}".format(total)
            print(f"Overall Rating of {self.name} (out of 3): {total}")
            print("Employee fails expectations")
obj = Evaluation()
obj.calculate()
```

Output:



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                                   Python

PS C:\Users\Abhinav\Downloads\ai prac> & C:/Users/Abhinav/AppData/Local/Programs/Python/Python310/python.exe "c:/Users/Abhinav/Downloads/ai prac/expertsystem.py"
Enter name of employee: 5
Enter rating from 1-3
Weightage should be equal to 100
Enter rating for Communication: 4
Enter weightage(100 remaining): 10
Enter rating for Productivity: 4
Enter weightage(90 remaining): 10
Enter rating for Creativity: 2
Enter weightage(80 remaining): 10
Enter rating for Integrity: 4
Enter weightage(70 remaining): 10
Enter rating for Punctuality: 5
Enter weightage(60 remaining): 10
Enter rating for Goal 1: 3
Enter weightage(50 remaining): 10
Enter rating for Goal 2: 2
Enter weightage(40 remaining): 10
Enter rating for Goal 3: 5
Enter weightage(30 remaining): 10
Enter rating for Goal 4: 5
Enter weightage(20 remaining): 10
Enter rating for Goal 5: 4
Enter weightage(10 remaining): 10

Competency Goals
Competency             Rating  Weightage       Weighted Score
Communication          4       10              0.4
Productivity           4       10              0.4
Creativity             2       10              0.2
Integrity              4       10              0.4
Punctuality            5       10              0.5

Sum of weighted scores-Comptency = 1.9
```



```
Sum of weighted scores-Comptency = 1.9

Performance Goals
Goals          Rating  Weightage       Weighted Score
Goal 1         3       10              0.3
Goal 2         2       10              0.2
Goal 3         5       10              0.5
Goal 4         5       10              0.5
Goal 5         4       10              0.4

Sum of weighted scores-Performance = 1.9

Overall Rating of 5 (out of 3): 3.80
Employee Exceeds expectations
PS C:\Users\Abhinav\Downloads\ai prac>

Ln 85, Col 1   Tab Size: 4   UTF-8   CRLF   {} Python   3.10.5 64-bit   Go Live
```