# AppcentAkademi_Project

June 3, 2022

[56]:
```python
import random
class Student:
    def __init__(self,std_no,name,age,department):
        self.std_no = std_no
        self.name = name
        self.department = department


class Department(object):
    def __init__(self, department_name):
        self.department_name = department_name


class Lecture():

    def __init__(self):
        self.grades = {}


    def add_student(self, student): #add students to the lesson
        if student.std_no not in self.grades:
            self.grades[(student.std_no, student.department.
 ↪department_name,student.name)] = []
        else:
            print("Student already added to the list")


    def add_grade_random(self, student): #creating the student's course grade␣
 ↪randomly
        student_grade = random.sample(range(1,100),1)
        self.grades[(student.std_no, student.department.department_name,␣
 ↪student.name)].append(student_grade[0])


    def dept_average_grade(self, department): #find software and computer␣
 ↪department averages
        dept_grades = []
        for k,v in self.grades.items():
            if k[1] == department.department_name:
                dept_grades = dept_grades + v
        return "{:.2f}".format(sum(dept_grades)/len(dept_grades))


    def decreasing_sort_dept_grades(self, department): #Sort chapter grades␣
↪from successful to unsuccessful
        dept_grades = {}
        for k,v in self.grades.items():
            if k[1] == department.department_name:
                dept_grades[k[0]] = v[0]
        sort_dept_grades =sorted(dept_grades.items(),key=lambda x:␣
↪x[1],reverse= True)
        print("Sorted list of",department.department_name,
              " department students of grades in descending order :" ,end=" ")
               #[i[1] for i in sort_dept_grades]
        for i in sort_dept_grades:
            print(i[1], end=" ")
        return sort_dept_grades


    def faculty_average_grade(self): #GPA of the faculty
        sum_faculty_grades  = 0
        sort_faculty_grades = sorted(self.grades.items(), key=lambda x: x[1],␣
↪reverse= True)
        print("Sorted list of Engineer faculty students of grades in descending␣
↪order: ",end=" ")
        for i in sort_faculty_grades:
            print(i[1][0], end=" ")
            sum_faculty_grades = sum_faculty_grades + i[1][0]

        return "{:.2f}".format(sum_faculty_grades/len(sort_faculty_grades))


    def find_susccessful_std(self,department): #Finding the most successful␣
↪student of the department
        dept_grades = {}
        for k,v in self.grades.items():
            if k[1] == department.department_name:
                dept_grades[k[2]] = v[0]

        successful_student = max(dept_grades.items(), key = lambda k : k[1])
        print("The most successful student of the ", department.department_name␣
↪,
              "department and letter grade:",successful_student[0],
              "-", self.letter_grade(successful_student[1]))


    def find_unsuccessful_std(self, department): #Finding the most unsuccessful␣
 ↪student of the department
        dept_grades = {}
        for k,v in self.grades.items():
            if k[1] == department.department_name:
                dept_grades[k[2]] = v[0]

        unsuccessful_student = min(dept_grades.items(), key = lambda k : k[1])
        print("The most unsuccessful student of the ", department.
 ↪department_name ,
              "department and letter grade:",unsuccessful_student[0], "-",
              self.letter_grade(unsuccessful_student[1]))


    def find_susccessful_std_faculty(self): #Finding the most successful␣
 ↪student of the faculty
        faculty_grades  = {}

        for k,v in self.grades.items():
            faculty_grades[k[2]] = v[0]
        #print(faculty_grades)
        successful_student = max(faculty_grades.items(), key = lambda k : k[1])
        print("The most successful student of the engineer faculty and letter␣
 ↪grade:",
              successful_student[0], "-", self.
 ↪letter_grade(successful_student[1]))

    def find_unsusccessful_std_faculty(self): #Finding the most unsuccessful␣
 ↪student of the faculty
        faculty_grades  = {}

        for k,v in self.grades.items():
            faculty_grades[k[2]] = v[0]
        #print(faculty_grades)
        successful_student = min(faculty_grades.items(), key = lambda k : k[1])
        print("The most successful student of the engineer faculty and letter␣
 ↪grade:",
              successful_student[0], "-", self.
 ↪letter_grade(successful_student[1]))


    def letter_grade(self,std_grade): #letter grades
        std_letter_grade =""
```

```
        if std_grade >80 and std_grade<= 100:
            return "A"
        if std_grade >60 and std_grade<= 80:
            return "B"
        if std_grade >40 and std_grade<= 60:
            return "C";
        if std_grade >20 and std_grade<= 40:
            return "D"
        if std_grade >=0 and std_grade<= 20:
            return "F"
        return std_letter_grade
```

[57]:
```
#Creating departments
computer_Engineer = Department("Computer Engineering")
software_Engineer = Department("Software Engineering")
```

[58]:
```
#Creating computer engineering students
merve = Student(20220001, "Merve Üstün", 23, computer_Engineer)
ali = Student(20220002, "Ali Bakar", 22, computer_Engineer)
nalan = Student(20220003, "Nalan Sarı", 25, computer_Engineer)
yagmur = Student(20220004, "Yagmur Kalaycıoglu", 20, computer_Engineer)
yasemin = Student(20220005, "Yasemin Tiryaki", 19, computer_Engineer)
baris = Student(20220006, "Barış Efe", 25, computer_Engineer)
berk = Student(20220007, "Berk Köseoğlu", 26, computer_Engineer)
alize = Student(20220008, "Alize Erkenci", 22, computer_Engineer)
tuna = Student(20220009, "Tuna Tuzlacı", 29, computer_Engineer)
deniz = Student(20220010, "Deniz ceylan", 23, computer_Engineer)
arda = Student(20220011, "Arda Şen", 21, computer_Engineer)
ayse = Student(202200012, "Ayşe Onay", 27, computer_Engineer)
hazal = Student(20220013, "Hazal Gencer", 25, computer_Engineer)
ozgur = Student(202200014, "Özgür Karadeniz", 21, computer_Engineer)
tugce = Student(202200014, "Tuğçe Koçak", 21, computer_Engineer)
mehmet= Student(202200016, "Mehmet Durmuşoğlu", 29, computer_Engineer)
ılgaz = Student(20220017, "Ilgaz Akagündüz", 22, computer_Engineer)
duru = Student(202200018, "Duru Öztürk", 19, computer_Engineer)

#Creating software engineering students

hakkı = Student(20220019, "Hakkı Topuz", 23, software_Engineer)
hasan = Student(20220020, "Hasan Dal", 24, software_Engineer)
ozgurmert = Student(20220021, "Özgür Mert", 22, software_Engineer)
emre = Student(20220022, "Emre Salık", 21, software_Engineer)
damla = Student(20220023, "Damla Akdeniz", 26, software_Engineer)
ece = Student(20220024, "Ece Çetinkaya", 20, software_Engineer)
```

```
sumru = Student(20220025, "Sumru Çiçek", 22, software_Engineer)
emrebulan = Student(20220026, "Emre Bulan", 21, software_Engineer)
can = Student(20220027, "Can Göktepe", 26, software_Engineer)
kutay = Student(20220028, "Kutay Talak", 20, software_Engineer)
aylin = Student(20220029, "Aylin Demirel", 24, software_Engineer)
ugur = Student(20220030, "Uğur Çalışkan", 21, software_Engineer)
```

[59]:
```
#Creating Lecture
software_Lecture = Lecture()

#Adding students to the course
software_Lecture.add_student(merve)
software_Lecture.add_student(ali)
software_Lecture.add_student(nalan)
software_Lecture.add_student(yagmur)
software_Lecture.add_student(yasemin)
software_Lecture.add_student(baris)
software_Lecture.add_student(berk)
software_Lecture.add_student(alize)
software_Lecture.add_student(tuna)
software_Lecture.add_student(deniz)
software_Lecture.add_student(arda)
software_Lecture.add_student(ayse)
software_Lecture.add_student(hazal)
software_Lecture.add_student(ozgur)
software_Lecture.add_student(tugce)
software_Lecture.add_student(mehmet)
software_Lecture.add_student(ılgaz)
software_Lecture.add_student(duru)
software_Lecture.add_student(hakkı)
software_Lecture.add_student(hasan)
software_Lecture.add_student(ozgurmert)
software_Lecture.add_student(emre)
software_Lecture.add_student(damla)
software_Lecture.add_student(ece)
software_Lecture.add_student(sumru)
software_Lecture.add_student(emrebulan)
software_Lecture.add_student(can)
software_Lecture.add_student(kutay)
software_Lecture.add_student(aylin)
software_Lecture.add_student(ugur)
```

[60]:
```
#add lecture notes to students
software_Lecture.add_grade_random(merve)
software_Lecture.add_grade_random(ali)
software_Lecture.add_grade_random(nalan)
software_Lecture.add_grade_random(yagmur)
software_Lecture.add_grade_random(yasemin)
software_Lecture.add_grade_random(baris)
software_Lecture.add_grade_random(berk)
software_Lecture.add_grade_random(alize)
software_Lecture.add_grade_random(tuna)
software_Lecture.add_grade_random(deniz)
software_Lecture.add_grade_random(arda)
software_Lecture.add_grade_random(ayse)
software_Lecture.add_grade_random(hazal)
software_Lecture.add_grade_random(ozgur)
software_Lecture.add_grade_random(tugce)
software_Lecture.add_grade_random(mehmet)
software_Lecture.add_grade_random(ılgaz)
software_Lecture.add_grade_random(duru)

software_Lecture.add_grade_random(hakkı)
software_Lecture.add_grade_random(hasan)
software_Lecture.add_grade_random(ozgurmert)
software_Lecture.add_grade_random(emre)
software_Lecture.add_grade_random(damla)
software_Lecture.add_grade_random(ece)
software_Lecture.add_grade_random(sumru)
software_Lecture.add_grade_random(emrebulan)
software_Lecture.add_grade_random(can)
software_Lecture.add_grade_random(kutay)
software_Lecture.add_grade_random(aylin)
software_Lecture.add_grade_random(ugur)
```

[61]:
```
computer_Engineer_grades= software_Lecture.
  ↪decreasing_sort_dept_grades(computer_Engineer)
```

```
Sorted list of Computer Engineering  department students of grades in descending
order : 94 93 90 89 84 84 78 67 65 65 58 48 35 26 26 7 4
```

[62]:
```
software_Engineer_grades = software_Lecture.
  ↪decreasing_sort_dept_grades(software_Engineer)
```

```
Sorted list of Software Engineering  department students of grades in descending
order : 87 82 71 61 59 52 49 48 45 42 41 32
```

[63]:
```
software_Lecture.find_susccessful_std(computer_Engineer)
```

```
The most successful student of the  Computer Engineering department and letter
grade: Arda Şen - A
```

[64]:
```
software_Lecture.find_susccessful_std(software_Engineer)
```

```
The most successful student of the  Software Engineering department and letter
```

```
grade: Uğur Çalışkan - A
```

[65]:
```python
software_Lecture.find_unsuccessful_std(computer_Engineer)
```

```
The most unsuccessful student of the  Computer Engineering department and letter
grade: Hazal Gencer - F
```

[72]:
```python
software_Lecture.find_unsuccessful_std(software_Engineer)
```

```
The most unsuccessful student of the  Software Engineering department and letter
grade: Emre Salık - D
```

[73]:
```python
print("Computer Engineer average grade: ",software_Lecture.
 ↪dept_average_grade(computer_Engineer))
```

```
Computer Engineer average grade:  58.00
```

[74]:
```python
print("Software Engineer average grade: ", software_Lecture.
 ↪dept_average_grade(software_Engineer))
```

```
Software Engineer average grade:  55.75
```

[75]:
```python
faculty_average_grade= software_Lecture.faculty_average_grade()
print("\nEngineer Faculty  average grade",faculty_average_grade)
```

```
Sorted list of Engineer faculty students of grades in descending order:  94 93
90 89 87 84 84 82 78 71 67 65 65 61 59 58 52 49 48 48 45 42 41 35 32 31 26 26 7
4
Engineer Faculty  average grade 57.10
```

[76]:
```python
software_Lecture.find_susccessful_std_faculty()
```

```
The most successful student of the engineer faculty and letter grade: Arda Şen -
A
```

[77]:
```python
software_Lecture.find_unsusccessful_std_faculty()
```

```
The most successful student of the engineer faculty and letter grade: Hazal
Gencer - F
```