**ALGORİTMA**

**VE**

**PROGRAMLAMA**

**Proje Adı**

**1- Alg ve Prog II, 2018- 2019 Proje-1**

**ÖĞRENCİ NO – AD SOYAD**

**1- 05160000018 - Umut GÖKBULUT**

**2- 05180000057 - Ahmet Furkan ERDEM**

**3- 05180000063 - Burak KIZILAY**

**Teslim Tarihi: 12.04.2019**

**İçindekiler**

1. **Tasarım............................................3**
   1. **Kullanılan Sınıf ve Metotlar.......3**
2. **Programcı Kataloğu..........................4**
   1. **Kaynak Kod Çıktısı....................4**
3. **Kullanıcı Kataloğu..............................7**

**3.1 Kısıtlamalar..................................11**

**1. TASARIM**

**1.1 Kullanılan Sınıf ve Metotlar**

* **public class Student {**

**// İçinde isim, soyisim, ID ve Counter değişkenini barındıran sınıftır. Constructer her çağrıldığında öğrenci sayısını belirten counter değeri bir artar.**

**}**

* **public class Graduate extends Student {**

**// Graduate sınıfı , Student sınıfının alt sınıfıdır. Student sınıfına ek olarak öğrencinin tez konusunu temsilen String tipinde “subject” değişkeni ile tez danışmanını temsilen String tipinde “advisor değişkenleri bulunmaktadır.**

**}**

* **public class UnderGraduate extends Student {**

**// UnderGraduate sınıfı da Student sınıfının alt sınıfıdır. Student sınıfına ek olarak öğrencinin eğitim kurumunda izlediği akademik patikayı (yazılım mühendisliği, yapay zeka, gömülü sistemler gibi) temsilen String tipinde “track” değişikeni bulunmaktadır. Buna ek olarak da Under Graduate öğrencinin sayısını tutan "UGCounter" adında bir sayaç bulunmaktadır.**

**}**

* **Bu saydığımız bütün sınıfların içinde parametresiz "constructor", tüm değişkenleri kullanan “constructor” ve “copy constructor” metotları ile “toString” metodu bulunmaktadır.**

**2. PROGRAMCI KATALOĞU**

Bu programda **tasarım** için harcadığımız süre yaklaşık olarak 2 saat 30 dakikadır. Tasarım kısmında hangi sınıfların oluşturulacağı ve nasıl düzenleneceği bir kağıt üzerinde tasarlanmıştır.

Bu programda **gerçekleştirim** için harcadığımız süre, verdiğimiz ufak aralarla, yaklaşık olarak 6 saattir. Gerçekleştirim kısmında da tasarladığımız sınıfları düzenli ve sıralı bir şekilde program olarak bilgisayara aktardık.

Bu programda **test** için harcadığımız süre yaklaşık olarak 1saat 30 dakikadır. Test aşamasında da farklı denemeler sonucunda karşılaşılan sorunlar düzeltilmiştir.

**2.1 Kaynak Kod**

**-Algo2-**

package algo2;

import java.io.File;

import java.io.FileNotFoundException;

import java.util.Scanner;

import java.util.StringTokenizer;

public class Algo2 {

public static void main(String[] args) {

Scanner input = null;

try {

input = new Scanner(new File("C:\\Users\\aycan kızılay\\Desktop\\Kurum.txt"));

} catch (FileNotFoundException e) {

System.out.println("File not found.");

System.exit(0);

}

StringTokenizer stringTokenizer = new StringTokenizer(input.nextLine(), ",");

int akademisyenler = Integer.parseInt(stringTokenizer.nextToken());

int akademisyenSayisi = akademisyenler;

String[] akademisyenIsımleri = new String[akademisyenSayisi];

int count = stringTokenizer.countTokens();

for (int i = 0; i < count; i++) {

akademisyenIsımleri[i] = stringTokenizer.nextToken();

}

int[][] genelListe = new int[200][akademisyenSayisi];

Student[] studentList = new Student[200];

while (input.hasNext()) {

Student newStudent = null;

stringTokenizer = new StringTokenizer(input.nextLine(), ",");

String cesit = stringTokenizer.nextToken();

if (cesit.equals("u")) {

newStudent = new UnderGraduate(Integer.parseInt(stringTokenizer.nextToken()),

stringTokenizer.nextToken(), stringTokenizer.nextToken(),

stringTokenizer.nextToken());

stringTokenizer = new StringTokenizer(input.nextLine(), ",");

for (int i = 0; i < akademisyenSayisi; i++) {

genelListe[Student.getCounter() - 1][i] = Integer.parseInt(stringTokenizer.nextToken());

}

} else {

newStudent = new Graduate(Integer.parseInt(stringTokenizer.nextToken()),

stringTokenizer.nextToken(), stringTokenizer.nextToken(), stringTokenizer.nextToken(),

stringTokenizer.nextToken());

stringTokenizer = new StringTokenizer(input.nextLine(), ",");

for (int i = 0; i < akademisyenSayisi; i++) {

genelListe[Student.getCounter() - 1][i] = Integer.parseInt(stringTokenizer.nextToken());

}

}

studentList[Student.getCounter() - 1] = newStudent;

}

// 1)

System.out.println("Her bir akademisyene ait ortalama dercelendirme puanları: \n");

Tools.ortalamaDerece(genelListe,akademisyenIsımleri,studentList,"S");

// 2)

System.out.println("\nHer bir akademisyen için lisans öğrencilerinin ortalama puanları: \n");

Tools.ortalamaDerece(genelListe,akademisyenIsımleri,studentList,"UG");

// 3)

System.out.println("\nHer bir akademisyen için lisansüstü öğrencilerin ortalama puanları: \n");

Tools.ortalamaDerece(genelListe,akademisyenIsımleri,studentList,"G");

// 4)

System.out.println("\nHer bir akademisyen için lisans öğrencilerinden izlediği patika yapay zeka olan öğrencilerin ortalama puanları: \n");

Tools.ortalamaDerece(genelListe,akademisyenIsımleri,studentList,"YZ");

System.out.println();

// 5)

Tools.ortalamaAlti(genelListe,akademisyenIsımleri,studentList,"UG");

// 6)

Tools.ortalamaAlti(genelListe,akademisyenIsımleri,studentList,"G");

int klavyeGiris = Student.getCounter();//Klavyeden girilecek Customer nesnelerinin başlangıç indeksi olacak.

// 7)

String devamMi = "r";

while(Student.getCounter() <200 && !devamMi.equalsIgnoreCase("h") ){

//g

// akademisyenSayisi = 4;

// a) Klavyeden yeni kullanıcılar oluşturulur, sonra her bir kullanıcı için ürün puanlamaları alınır:

int indeks = Student.getCounter();

studentList[indeks] = Tools.ogrenciOlustur();

System.out.println("Oluşturulan öğrenci için; ");

for (int i = 0; i < akademisyenSayisi - 1; i++){

System.out.print((i+1) + ". akademisyen puanlamasını giriniz: ");

genelListe[indeks][i] = Tools.getSpesifikNumara("Derece");

}

// b) En son ürün puanı önceden puan vermiş kullanıcıların puanları kullanılarak hesaplanır:

int min = 1000;

double aynıBenzerlikSayac = 0.0;

int esitSayisi = 0;

for (int i = 0; i < indeks; i++){

int benzerlikDegeri = 0;

for(int j = 0; j < akademisyenSayisi - 1; j++){

benzerlikDegeri += Tools.mutlakDeger(genelListe[i][j] , genelListe[indeks][j]);

}

if ( benzerlikDegeri < min){

min = benzerlikDegeri;

genelListe[indeks][akademisyenSayisi - 1] = genelListe[i][akademisyenSayisi - 1];

aynıBenzerlikSayac = genelListe[indeks][akademisyenSayisi - 1];

esitSayisi = 1;

}else if( benzerlikDegeri == min){

esitSayisi++;

aynıBenzerlikSayac += genelListe[i][akademisyenSayisi - 1];

genelListe[indeks][akademisyenSayisi -1] = (int) Math.round(aynıBenzerlikSayac/(double)esitSayisi);

}

}

devamMi = Tools.getString("Çıkmak İçin h'ye Devam Etmek İçin Herhangi Bir Tuşa Basınız:");

//if(devamMi.equals("h") || devamMi.equals("H")){

}

//}

// 8)

// a)Klavyeden griş yapılarak oluşturulan kullanıcıların bilgileri:

akademisyenSayisi = akademisyenler;

int[][] klavyedenGirilen = new int[Student.getCounter() - klavyeGiris + 1][akademisyenSayisi];

System.out.println("Klavyeden giriş yapılarak oluşturulan kullanıcıların bilgileri:\n");

int j = 0;

int studentNumber = Student.getCounter();

Student.setCounter(studentNumber - klavyeGiris);

for (int i = klavyeGiris; i < studentNumber; i++){

System.out.println(studentList[i]);

for (int k = 0; k < akademisyenSayisi; k++){

klavyedenGirilen[j][k] = genelListe[i][k];

}

j++;

}

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

// b) Klayveden girilen bilgilerle oluşturulan müşterilerin ürünlere verdiği not ortalamaları:

Tools.ortalamaDerece(klavyedenGirilen,akademisyenIsımleri,studentList,"S");

}

}

**-Tools-**

package algo2;

import java.util.InputMismatchException;

import java.util.Scanner;

public class Tools {

public static String getString(String s) {

System.out.print(s);

Scanner in = new Scanner(System.in);

return in.nextLine();

}

public static int getInt(String s) {

Scanner in = new Scanner(System.in);

System.out.print(s);

try {

return in.nextInt();

} catch (InputMismatchException e) {

//System.out.println("Lütfen sayısal bir değer giriniz.");

return getInt("Lütfen sayısal bir değer giriniz: ");

}

}

public static void ortalamaDerece(int[][] list, String[] akademisyenListesi, Student[] studentList, String mod) {

for (int j = 0; j < akademisyenListesi.length; j++) {

int counter = 0;

double ortalama = 0;

if (mod.equals("S")) {

for (int i = 0; i < Student.getCounter(); i++) {

counter += list[i][j];

}

ortalama = (double) counter / (double) Student.getCounter();

} else if (mod.equals("UG")) {

for (int i = 0; i < Student.getCounter(); i++) {

if (studentList[i] instanceof UnderGraduate) {

counter += list[i][j];

}

}

ortalama = (double) counter / (double) UnderGraduate.getUGCounter();

} else if (mod.equals("G")) {

for (int i = 0; i < Student.getCounter(); i++) {

if (studentList[i] instanceof Graduate) {

counter += list[i][j];

}

}

ortalama = (double) counter / (double) Graduate.getGCounter();

} else if (mod.equals("YZ")) {

int YZCounter = 0;

for (int i = 0; i < Student.getCounter(); i++) {

if (studentList[i] instanceof UnderGraduate) {

UnderGraduate UGStudent = (UnderGraduate) studentList[i];

if (UGStudent.getTrack().equalsIgnoreCase("Yapay zeka")) {

counter += list[i][j];

YZCounter++;

}

}

}

ortalama = (double) counter / (double) YZCounter;

}

System.out.printf("%s adlı öğretim elemanının ortalama puanı %.1f\n", akademisyenListesi[j], ortalama);

}

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

}

private static double ortalama(int[][] genelListe, int indeks) {

int counter = 0;

for (int i = 0; i < Student.getCounter(); i++) {

counter += genelListe[i][indeks];

}

return (double) counter / (double) Student.getCounter();

}

public static void ortalamaAlti(int[][] genelListe, String[] akademisyenListesi, Student[] studentList, String mod) {

for (int j = 0; j < akademisyenListesi.length; j++) {

int sayac = 0;

if (mod.equals("UG")){

System.out.println(akademisyenListesi[j] + " akademisyenine ortalamanın altında not veren lisans öğrencileri:\n");

System.out.println("");

}

else if (mod.equals("G")){

System.out.println(akademisyenListesi[j] + " akademisyenine ortalamanın altında not veren lisansüstü öğrencileri:\n");

System.out.println("");

}

double akademisyenOrt = ortalama(genelListe, j);

for (int i = 0; i < studentList.length; i++) {

if ((double) genelListe[i][j] < akademisyenOrt && studentList[i] != null) {

if (mod.equals("UG")) {

if (studentList[i] instanceof UnderGraduate) {

System.out.println(studentList[i]);

sayac ++;

}

} else if (mod.equals("G")) {

if (studentList[i] instanceof Graduate) {

System.out.println(studentList[i]);

sayac ++;

}

}

}

}

if (sayac == 0){

System.out.println("Hiç Bir Kullanıcı Bulunamadı.");

}

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

}

}

public static int mutlakDeger(int a, int b){

if(a > b)

return a - b;

else

return b - a;

}

public static int getSpesifikNumara(String mod){

int deger;

if (mod.equals("Derece")) {

deger = getInt("");

if (deger < 0 || deger > 5) {

System.out.print("Lütfen 0-5 arası değerler giriniz: ");

return getSpesifikNumara("Derece");

} else {

return deger;

}

}

return 0;

}

public static Student ogrenciOlustur(){

Student newStudent = null;

String ogrenciTipi = getString("Lütfen ogrenci tipini giriniz (u/g)");

if (ogrenciTipi.equalsIgnoreCase("u")||(ogrenciTipi.equalsIgnoreCase("ug"))){

newStudent = new UnderGraduate(getInt("Öğrenci ID si giriniz: "),getString("Öğrenci adı giriniz: "),

getString("Öğrenci soyadı giriniz: "),

getString("Öğrencinin izlediği akademik patikayı giriniz: "));

}else if (ogrenciTipi.equalsIgnoreCase("g")){

newStudent = new Graduate(getInt("Öğrenci ID si giriniz: "),getString("Öğrenci adı giriniz: "),

getString("Öğrenci soyadı giriniz: "), getString("Öğrencinin tez konusunu giriniz: "),

getString("Öğrencinin akademik danışmanını giriniz: "));

}

else{

System.out.println("Yanlış Giriş Yaptınız Lütfen Tekrar Deneyin...");

ogrenciOlustur();

}

return newStudent;

}

}

**-Student-**

package algo2;

public class Student {

private int studentID;

private String name;

private String surname;

private static int counter = 0; // Ortalama hesaplamalarında kullanılmak üzere statik sınıf sayacı

public Student(int studentID, String name, String surname) {

this.studentID = studentID;

this.name = name;

this.surname = surname;

this.counter++;

}

public Student() {

this.studentID = 0;

this.name = null;

this.surname = null;

this.counter++;

}

public Student(Student student){

this.studentID = student.studentID;

this.name = student.name;

this.surname = student.surname;

this.counter++;

}

@Override

public String toString(){

return "StudentID: " + studentID + " Name: " + name + " Surname: " + surname;

}

public int getStudentID() {

return studentID;

}

public void setStudentID(int studentID) {

this.studentID = studentID;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getSurname() {

return surname;

}

public void setSurname(String surname) {

this.surname = surname;

}

public static int getCounter() {

return counter;

}

public static void setCounter(int counter) {

Student.counter = counter;

}

}

**-Graduate-**

package algo2;

public class Graduate extends Student {

private String subject;

private String advisor;

private static int GCounter = 0; // Ortalama hesaplamalarında kullanılmak üzere statik sınıf sayacı

public Graduate(int studentID, String name, String surname, String subject, String advisor) {

super(studentID, name, surname);

this.subject = subject;

this.advisor = advisor;

this.GCounter++;

}

public Graduate() {

super();

this.subject = null;

this.advisor = null;

this.GCounter++;

}

public Graduate(Graduate G) {

super(G.getStudentID(),G.getName(),G.getSurname());

this.subject = G.subject;

this.advisor = G.advisor;

this.GCounter++;

}

@Override

public String toString(){

return super.toString() + " Subject: " + subject + " Advisor: " + advisor;

}

public String getSubject() {

return subject;

}

public void setSubject(String subject) {

this.subject = subject;

}

public String getAdvisor() {

return advisor;

}

public void setAdvisor(String advisor) {

this.advisor = advisor;

}

public static int getGCounter() {

return GCounter;

}

public static void setGCounter(int GCounter) {

Graduate.GCounter = GCounter;

}

}

**-UnderGraduate-**

package algo2;

public class UnderGraduate extends Student{

private String track;

private static int UGCounter;

public UnderGraduate(int studentID, String name, String surname, String track) {

super(studentID, name, surname);

this.track = track;

this.UGCounter++;

}

public UnderGraduate() {

super();

this.track = null;

this.UGCounter++;

}

public UnderGraduate(UnderGraduate UG) {

super(UG.getStudentID(),UG.getName(),UG.getSurname());

this.track = UG.track;

this.UGCounter++; // Ortalama hesaplamalarında kullanılmak üzere statik sınıf sayacı

}

@Override

public String toString(){

return super.toString() + " Track: " + getTrack();

}

public String getTrack() {

return track;

}

public void setTrack(String track) {

this.track = track;

}

public static int getUGCounter() {

return UGCounter;

}

public static void setNCCounter(int UGCounter) {

UnderGraduate.UGCounter = UGCounter;

}

}

**2.2 Kaynak Kod Çıktısı**

Aşağıda, önceden hazırlanmış olan Kurum.txt dosyası ile çalıştırılmış olan programın çıktısı bulunmaktadır.

Her bir akademisyene ait ortalama dercelendirme puanları:

A adlı öğretim elemanının ortalama puanı 3,8

B adlı öğretim elemanının ortalama puanı 2,4

C adlı öğretim elemanının ortalama puanı 3,0

D adlı öğretim elemanının ortalama puanı 3,6

E adlı öğretim elemanının ortalama puanı 1,8

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

Her bir akademisyen için lisans öğrencilerinin ortalama puanları:

A adlı öğretim elemanının ortalama puanı 3,3

B adlı öğretim elemanının ortalama puanı 3,0

C adlı öğretim elemanının ortalama puanı 3,7

D adlı öğretim elemanının ortalama puanı 4,0

E adlı öğretim elemanının ortalama puanı 1,7

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

Her bir akademisyen için lisansüstü öğrencilerin ortalama puanları:

A adlı öğretim elemanının ortalama puanı 4,5

B adlı öğretim elemanının ortalama puanı 1,5

C adlı öğretim elemanının ortalama puanı 2,0

D adlı öğretim elemanının ortalama puanı 3,0

E adlı öğretim elemanının ortalama puanı 2,0

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

Her bir akademisyen için lisans öğrencilerinden izlediği patika yapay zeka olan öğrencilerin ortalama puanları:

A adlı öğretim elemanının ortalama puanı 3,0

B adlı öğretim elemanının ortalama puanı 3,0

C adlı öğretim elemanının ortalama puanı 3,5

D adlı öğretim elemanının ortalama puanı 4,0

E adlı öğretim elemanının ortalama puanı 1,5

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

A akademisyenine ortalamanın altında not veren lisans öğrencileri:

StudentID: 101 Name: Ali Surname: Ceviz Track: Yapay zeka

StudentID: 103 Name: Zeynep Surname: Demir Track: Yapay zeka

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

B akademisyenine ortalamanın altında not veren lisans öğrencileri:

StudentID: 103 Name: Zeynep Surname: Demir Track: Yapay zeka

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

C akademisyenine ortalamanın altında not veren lisans öğrencileri:

Hiç Bir Kullanıcı Bulunamadı.

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

D akademisyenine ortalamanın altında not veren lisans öğrencileri:

StudentID: 103 Name: Zeynep Surname: Demir Track: Yapay zeka

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

E akademisyenine ortalamanın altında not veren lisans öğrencileri:

StudentID: 101 Name: Ali Surname: Ceviz Track: Yapay zeka

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

A akademisyenine ortalamanın altında not veren lisansüstü öğrencileri:

Hiç Bir Kullanıcı Bulunamadı.

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

B akademisyenine ortalamanın altında not veren lisansüstü öğrencileri:

StudentID: 201 Name: Ayse Surname: Can Subject: Makine ogrenmesi Advisor: Ahmet Celik

StudentID: 202 Name: Ahmet Surname: Tunc Subject: Veri madenciligi Advisor: Can Durmaz

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

C akademisyenine ortalamanın altında not veren lisansüstü öğrencileri:

StudentID: 201 Name: Ayse Surname: Can Subject: Makine ogrenmesi Advisor: Ahmet Celik

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

D akademisyenine ortalamanın altında not veren lisansüstü öğrencileri:

StudentID: 202 Name: Ahmet Surname: Tunc Subject: Veri madenciligi Advisor: Can Durmaz

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

E akademisyenine ortalamanın altında not veren lisansüstü öğrencileri:

StudentID: 202 Name: Ahmet Surname: Tunc Subject: Veri madenciligi Advisor: Can Durmaz

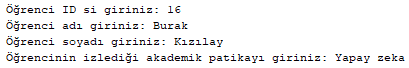
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Lütfen ogrenci tipini giriniz (u/g)

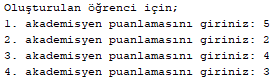
**\*\*Buradan sonra kullanıcı girişi beklenmektedir. Örnek kullanıcı girişi, kullanıcı kataloğunda bulunmaktadır.**

**3. KULLANICI KATALOĞU**

**1.** İlk olarak kullanıcıdan öğrenci tipi, yani Graduate(g) mi yoksa UnderGraduate(u) mi olduğunun girdisi istenmektedir.1.png

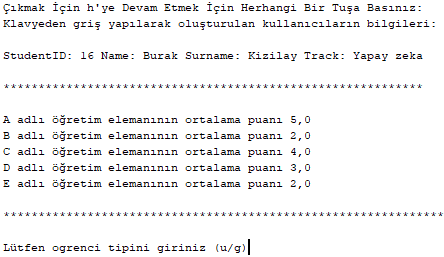
**2.** Örnek olarak u girişi yapıldığını var sayalım. Bu durumda karşımıza aşağıdaki görselde görüldüğü gibi ID, ad, soyad ve akademik patika bilgileri istenmektedir.

**3.** Yukarıdaki örnekteki girdiler yapıldıktan sonra kullanıcıdan akademisyen puanlamalarını istemektedir. Örnek kullanıcı girdisi aşağıdadır.

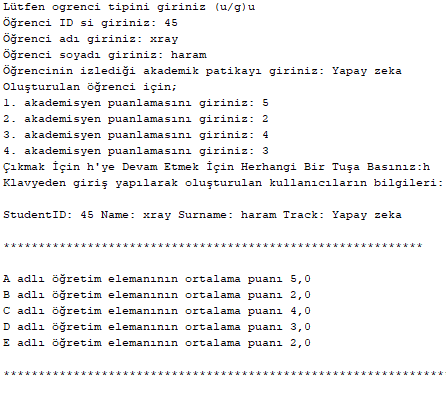


**4.** Bu girişler de yapıldıktan sonra kullanıcıya devam etmek isteyip istemediği sorulur. Devam etmek istemiyorsanız "h" tuşuna basarak programdan çıkış yapabilirsiniz. Devam etmek istiyorsanız herhangi bir tuşa basmanız gerekmektedir. Aşağıdaki görsellerde iki durum da gösterilmiştir.

**Devam edilirse oluşacak çıktı aşağıdaki gibidir.**



**Devam edilmeyecekse oluşacak çıktı aşağıdaki gibidir.**

****

**\*\*Örnek bir çalıştırma yukarıda gösterildiği gibidir. Program kullanılırken dikkat edilmesi gereken hususlar "Kısıtlamalar" başlığı altında bulunmaktadır.**

**3.1 Kısıtlamalar**

* Program maksimum 200 öğrenci kapasitesindedir. Lütfen daha fazla veri girişi yapmayınız.
* Girilecek akademisyen puanlamaları 0'dan büyük 5'ten küçük veya eşit olmalıdır.
* Öğrenci tipi u veya g olarak girilmelidir. Başka değerler girilmemesi gerekmektedir.
* Sayısal girilmesi gereken değerler sayısal; yazılı(string tipinde) girilmesi gereken değerler yazılı şekilde girilmelidir.