

1.

19BCD7088

Mick's Wicks makes candles in various sizes. Create a class for the business named Candle that contains data fields for color, height, and price. Create get methods for all three fields. Create set methods for color and height, but not for price. Instead, when height is set, determine the price as \$2 per inch. Create a child class named ScentedCandle that contains an additional data field named scent and methods to get and set it. In the child class, override the parent's setHeight() method to set the price of a ScentedCandle object at \$3 per inch. Write an application that instantiates an object of each type and displays the details. Save the files as Candle.java, ScentedCandle.java, and DemoCandles.java.

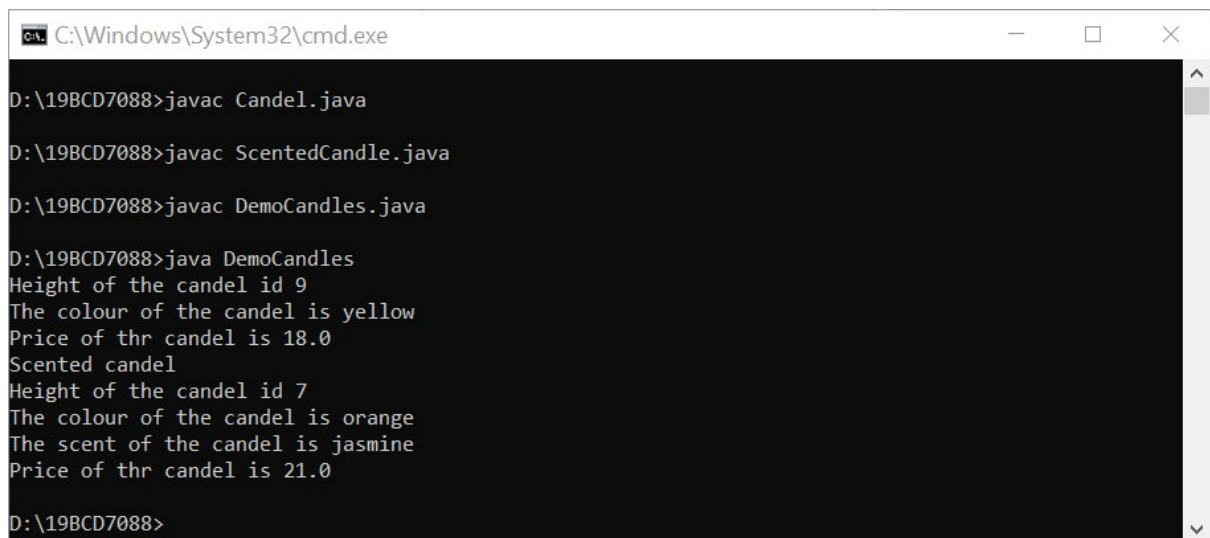
```
class Candle
{
String color;
int height;
double price;
void getColor(){
System.out.println ("The colour of the candel is " + color);
}
void getHeight()
{
System.out.println("Height of the candel id " + height);
}
void getPrice()
{
System.out.println("Price of thr candel is " + price);
}
void setColor(String c)
{
color = c;
}
void setHeight(int h)
{
height=h;
```

```

double n = 2;
price = h * n;
}
}
.....
class ScentedCandle extends Candle
{
    String scent;
    void getScent()
    {
        System.out.println("The scent of the candle is " + scent);
    }
    void setScent(String s)
    {
        scent = s;
    }
    void setHeight(int h)
    {
        double n = 3;
        super.setHeight(h);
        price = h * n;
    }
}
.....
public class DemoCandles
{
    public static void main(String args[])
    {
        Candle a = new Candle();
        ScentedCandle b = new ScentedCandle();
        a.setColor("yellow");
        a.setHeight(9);
        b.setColor("orange");
    }
}

```

```
b.setScent("jasmine");  
b.setHeight(7);  
a.getHeight();  
a.getColor();  
a.getPrice();  
System.out.println("Scented candel");  
b.getHeight();  
b.getColor();  
b.getScent();  
b.getPrice();  
}  
}
```



A screenshot of a Windows command prompt window titled "C:\Windows\System32\cmd.exe". The window shows the following commands and output:

```
D:\19BCD7088>javac Candel.java  
D:\19BCD7088>javac ScentedCandle.java  
D:\19BCD7088>javac DemoCandles.java  
D:\19BCD7088>java DemoCandles  
Height of the candel id 9  
The colour of the candel is yellow  
Price of thr candel is 18.0  
Scented candel  
Height of the candel id 7  
The colour of the candel is orange  
The scent of the candel is jasmine  
Price of thr candel is 21.0  
D:\19BCD7088>
```

2.

Create a class named Poem that contains fields for the name of the poem and the number of lines in it. Include a constructor that requires values for both fields. Also include get methods to retrieve field values. Create three subclasses: Couplet, Limerick, and Haiku. The constructor for each subclass requires only a title; the lines field is set using a constant value. A couplet has two lines, a limerick has five lines, and a haiku has three lines. Create an application that demonstrates usage of an object of each type. Save the files as Poem.java, Couplet.java, Limerick.java, Haiku.java, and DemoPoems.java.

```
class Poem
```

```
{
```

```
String name;
```

```
int lines;
```

```
Poem(String name, int lines)
```

```
{
```

```
this.name = name;
```

```
this.lines = lines;
```

```
}
```

```
String getPoemName()
```

```
{
```

```
return name;
```

```
}
```

```
int getLines()
```

```
{
```

```
return lines ;
```

```
}
```

```
}
```

```
.....
```

```
class Couplet extends Poem
```

```
{
```

```
Couplet(String name)
```

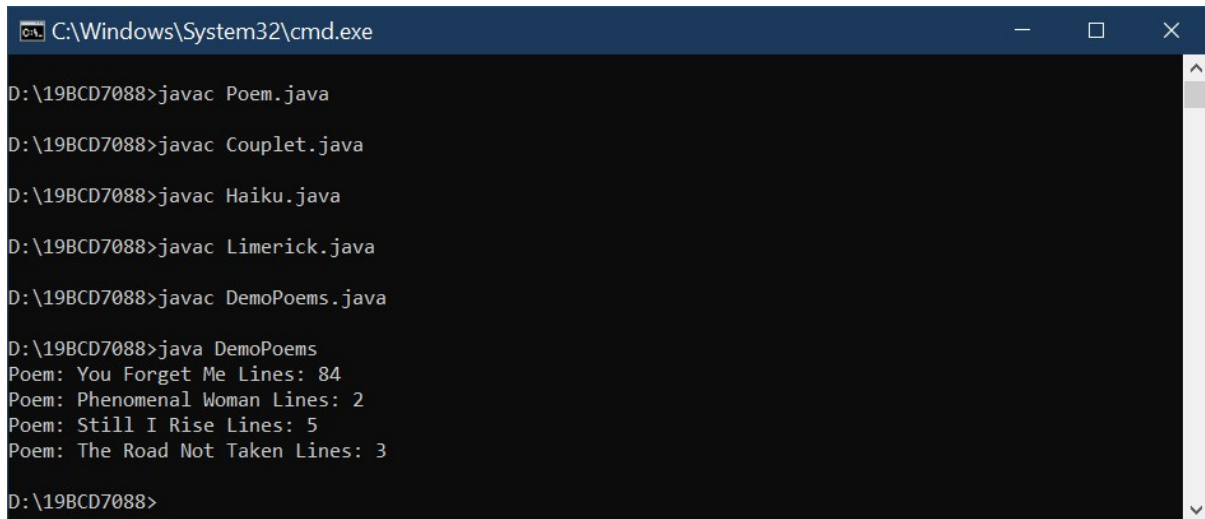
```

{
super(name,2);
}
}
.....
class Haiku extends Poem
{
Haiku(String name)
{
super(name,3);
}
}
.....
class Limerick extends Poem
{
Limerick(String name)
{
super(name,5);
}
}
.....
public class DemoPoems
{
public static void main(String[] args)
{
Poem p1 = new Poem("You Forget Me", 84);
Couplet p2 = new Couplet("Phenomenal Woman");
Limerick p3 = new Limerick("Still I Rise");
Haiku p4 = new Haiku("The Road Not Taken");

System.out.println("Poem: " + p1.getPoemName() +" Lines: " + p1.getLines());
System.out.println("Poem: " + p2.getPoemName() +" Lines: " + p2.getLines());
System.out.println("Poem: " + p3.getPoemName() +" Lines: " + p3.getLines());
System.out.println("Poem: " + p4.getPoemName() +" Lines: " + p4.getLines());
}
}

```

```
}  
}
```



```
C:\Windows\System32\cmd.exe  
D:\19BCD7088>javac Poem.java  
D:\19BCD7088>javac Couplet.java  
D:\19BCD7088>javac Haiku.java  
D:\19BCD7088>javac Limerick.java  
D:\19BCD7088>javac DemoPoems.java  
D:\19BCD7088>java DemoPoems  
Poem: You Forget Me Lines: 84  
Poem: Phenomenal Woman Lines: 2  
Poem: Still I Rise Lines: 5  
Poem: The Road Not Taken Lines: 3  
D:\19BCD7088>
```

3

The developers of a free online game named "Sugar Smash" have asked you to develop a class named `SugarSmashPlayer` that holds data about a single player. The class contains the following fields: the player's integer ID number, a `String` screen name, and an array of integers that stores the highest score achieved in each of 10 game levels. Include get and set methods for each field. The get and set methods for the scores should each require two parameters—one that represents the score achieved and one that represents the game level to be retrieved or assigned. Display an error message if the user attempts to assign or retrieve a score from a level that is out of range for the array of scores. Additionally, no level except the first one should be set unless the user has earned at least 100 points at each previous level. If a user tries to set a score for a level that is not yet available, issue an error message. Create a class named `PremiumSugarSmashPlayer` that descends from `SugarSmashPlayer`. This class is instantiated when a user pays \$2.99 to have access to 40 additional levels of play. As in the free version of the game, a user cannot set a score for a level unless the user has earned at least 100 points at all previous levels. Create a program that instantiates several objects of each type and demonstrates the methods. Save the files as `SugarSmashPlayer.java`, `PremiumSugarSmashPlayer.java`, and `DemoSugarSmash.java`.

```
class SugarSmashPlayer  
{  
    int ID;  
    String screenName;  
    protected int[] scores = new int[10];  
    void setId(int num)  
    {  
        ID = num;  
    }  
}
```

```
void setName(String player)
{
    screenName = player;
}

void setScore(int score, int level)
{

    if (level == 0)
        { scores[level] = score;}
    else
    {
        if (scores[level - 1] >= 100 && level < scores.length)
            {scores[level] = score;}
        else
        {
            System.out.println("Invalid score");
        }
    }
}

int getId()
{
    return ID;
}

String getName(){
    return screenName;
}

int getScore(int level)
{
    if (level >= scores.length)
    {
        System.out.println("Invalid game level");
    }
}
```

```

        return 0;
    }
    else{
        return scores[level];

    }

}

}
.....
class PremiumSugarSmashPlayer extends SugarSmashPlayer
{
    protected int[] scores = new int[50];
    void setScore(int score, int level)
    {

        if (level == 0)
            { scores[level] = score;}
        else
        {
            if (scores[level - 1] >= 100 && level < scores.length)
                {scores[level] = score;}
            else
            {
                System.out.println("Invalid score");
            }
        }
    }

    int getScore(int level)
    {
        if (level >= scores.length)
        {

```



```

        System.out.println("Invalid game level");
        return -1;
    }
    else{
        return scores[level];
    }

}

}
}
.....
public class DemoSugarSmash
{
    public static void main(String[] args)
    {
        SugarSmashPlayer ss = new SugarSmashPlayer();
        PremiumSugarSmashPlayer ps = new PremiumSugarSmashPlayer();
        ss.setName("Jhon");
        ss.setld(1519);
        int a= 100;
        for(int i = 0;i<10;i++){
            ss.setScore(a,i);
            a=a+100;
        }
        String name = ss.getName();
        int n = ss.getld();
        System.out.print(name + " of player Id " + n + " Scores are ");
        int o;
        for(int j=0;j<10;j++){
            o=ss.getScore(j);
            System.out.print(o + " ");

```

```

    }

    System.out.println();

    ps.setId(9562);

    ps.setName("Vicky");

    int b= 100;

    for(int k = 0;k<50;k++){

        ps.setScore(b,k);

        b=b+100;

    }

    name = ps.getName();

    n=ps.getId();

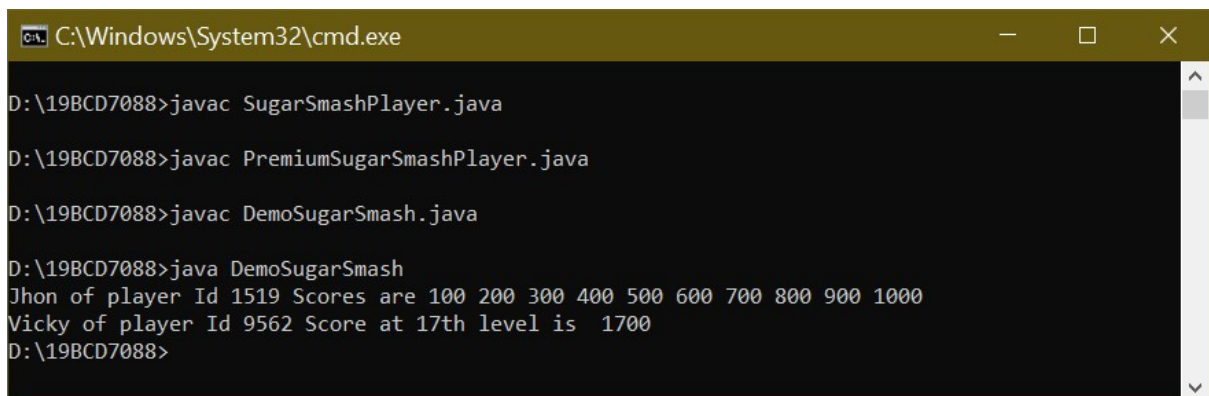
    System.out.print(name + " of player Id " + n + " Score at 17th level is ");

    System.out.print(ps.getScore(16));

}

}

```



The screenshot shows a Windows command prompt window titled "C:\Windows\System32\cmd.exe". The window contains the following text:

```

D:\19BCD7088>javac SugarSmashPlayer.java

D:\19BCD7088>javac PremiumSugarSmashPlayer.java

D:\19BCD7088>javac DemoSugarSmash.java

D:\19BCD7088>java DemoSugarSmash
Jhon of player Id 1519 Scores are 100 200 300 400 500 600 700 800 900 1000
Vicky of player Id 9562 Score at 17th level is 1700
D:\19BCD7088>

```

4.

Create a class named `CollegeCourse` that includes data fields that hold the department (for example, ENG), the course number (for example, 101), the credits (for example, 3), and the fee for the course (for example, \$360). All of the fields are required as arguments to the constructor, except for the fee, which is calculated at \$120 per credit hour. Include a `display()` method that displays the course data. Create a subclass named `LabCourse` that adds \$50 to the course fee. Override the parent class `display()` method to indicate that the course is a lab course and to display all the data. Write an application named `UseCourse` that prompts the user for course information. If the user enters a class in any of the following departments, create a `LabCourse`: BIO, CHM, CIS, or PHY. If the user enters any other department, create a `CollegeCourse` that does not include the lab fee. Then display the course data. Save the files as `CollegeCourse.java`, `LabCourse.java`, and `UseCourse.java`.

```
class CollegeCourse
{
    double Ch = 120.00;
    String dpt;
    int num;
    int c;
    double cf;
    CollegeCourse(String d, int n, int nc)
    {
        dpt = d.toUpperCase();
        num = n;
        c = nc;
        cf = Ch * c;
    }
    String getdpt()
    {
        return dpt;
    }
    int getcourseNo()
    {
        return num;
    }
}
```

```

int getCredits()
{
    return c;
}

double getcourseFee()
{
    return cf;
}

void display()
{
    System.out.println("department " + this.getDpt());
    System.out.println("Course number " + this.getcourseNo());
    System.out.println("Credit hours " + this.getCredits());
    System.out.println("Course fee " + this.getcourseFee());
}

```

```

}
.....

```

```

class LabCourse extends CollegeCourse

```

```

{
    double lf = 50.00;
    double cf;
    LabCourse(String dpt, int cn, int c)
    {
        super(dpt, cn, c);
        cf = super.getcourseFee() + lf;
    }
    double getLabCourseFee()
    {
        return cf;
    }
}

```

```

    }

    void display()
    {
        System.out.println("department " + super.getdpt());
        System.out.println("Course number " + super.getcourseNo());
        System.out.println("Credit hours " + super.getCredits());
        System.out.println("Course fee " + this.getLabCourseFee());
    }

```

```

}
.....
import java.util.Scanner;

public class UseCourse{

    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter the department of the course: ");

        String dept = sc.nextLine();

        System.out.print("Enter the number of the courses ");

        int number = sc.nextInt();

        System.out.print("Enter the credit hours of the courses ");

        int hours = sc.nextInt();

        if(dept.equals("BIO") || dept.equals("CHM") || dept.equals("CIS") || dept.equals("PHY"))
        {
            LabCourse l = new LabCourse(dept, number, hours);

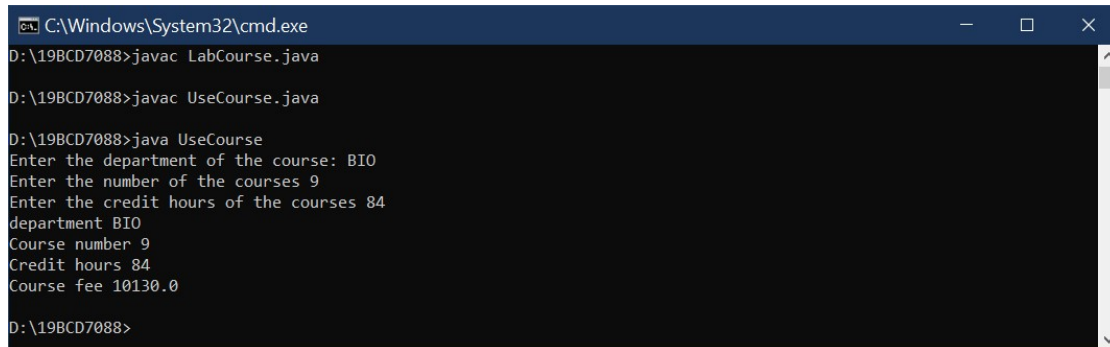
            l.display();
        }
        else
        {
            CollegeCourse c = new CollegeCourse(dept, number, hours);

            c.display();
        }
    }
}

```

}

}



```
C:\Windows\System32\cmd.exe
D:\19BCD7088>javac LabCourse.java
D:\19BCD7088>javac UseCourse.java
D:\19BCD7088>java UseCourse
Enter the department of the course: BIO
Enter the number of the courses 9
Enter the credit hours of the courses 84
department BIO
Course number 9
Credit hours 84
Course fee 10130.0
D:\19BCD7088>
```

5.

Develop a set of classes for a college to use in various student service and personnel applications. Classes you need to design include the following:

- Person—A Person contains a first name, last name, street address, zip code, and phone number. The class also includes a method that sets each data field, using a series of dialog boxes and a display method that displays all of a Person's information on a single line at the command line on the screen.
- CollegeEmployee—CollegeEmployee descends from Person. A CollegeEmployee also includes a Social Security number, an annual salary, and a department name, as well as methods that override the Person methods to accept and display all CollegeEmployee data.
- Faculty—Faculty descends from CollegeEmployee. This class also includes Boolean field that indicates whether the Faculty member is tenured, as well as methods that override the CollegeEmployee methods to accept and display this additional piece of information.
- Student—Student descends from Person. In addition to the fields available in Person, a Student contains a major field of study and a grade point average as well as methods that override the Person methods to accept and display these additional facts.

Write an application named CollegeList that declares an array of four "regular" CollegeEmployees, three Faculty, and seven Students. Prompt the user to specify which type of person's data will be entered (C, F, or S), or allow the user to quit (Q). While the user chooses to continue (that is, does not quit), accept data entry for the appropriate type of Person. If the user attempts to enter data for more than four CollegeEmployees, three Faculty, or seven Students, display an error message. When the user quits, display a report on the screen listing each group of Persons under the appropriate heading of "College Employees," "Faculty," or "Students." If the user has not entered data for one or more types of Persons during a session, display an appropriate message under the appropriate heading. Save the files as Person.java, CollegeEmployee.java, Faculty.java, Student.java, and CollegeList.java.

```
class Person{  
    String firstName;  
    String lastName;  
    String address;  
    int pincode;  
    long phonenumber;  
    void setFirstName(String firstName){  
        this.firstName= firstName;  
    }  
    void setLastName(String lastName){  
        this.lastName=lastName;  
    }  
}
```

```

    }

    void setAddress(String address){
        this.address = address;
    }

    void setPincode(int pincode){
        this.pincode = pincode;
    }

    void setphonenum(long phonenum){
        this.phonenum= phonenum;
    }

    void display(){
        System.out.println(this.firstName + " " + this.lastName + "'s " + "Address is " +
this.address + "," + this.pincode + ".Mobile number is " + this.phonenum);
    }

```

```

}

```

```

class CollegeEmployee extends Person{

```

```

    int sn;

    double a;

    String dept;

    void setSnum(int n){
        this.sn=n;
    }

    void setAsal(double b){
        this.a=b;
    }

    void setdept(String dept){
        this.dept=dept;
    }

```



```

        void display(){
            super.display();

            System.out.println("His Security number is "+this.sn+" and Annual salary is
"+this.a+". He belongs to Department of "+this.dept+".");
        }

```

```

}
.....

```

```

class Faculty extends CollegeEmployee

```

```

{
    boolean tenured;
    void setTenured(boolean t){
        this.tenured=t;
    }

    public void display()
    {
        super.display();
        if(tenured){
            System.out.println("Faculty member is tenured");
        }
        else{
            System.out.println("Faculty member is not tenured");
        }
    }
}

```

```

.....

```

```

class Student extends Person

```

```

{
    String major;

```

```

double avg;

void setMajor(String major)
{
    this.major = major;
}

void setAvg(double avg)
{
    this.avg = avg;
}

public String getMajor()
{
    return major;
}

public double getAvg()
{
    return avg;
}

public void display()
{
    super.display();

    System.out.println("His major is " + getMajor() + " and his average is " + getAvg());
}
}
.....
import java.util.Scanner;

public class CollegeList
{
    public static void main(String[] args)

```

```

{
    CollegeEmployee[] c = new CollegeEmployee[4];
    Faculty[] f = new Faculty[3];
    Student[] s = new Student[7];
    Scanner sc = new Scanner(System.in);
    String response, fname, lname, address, dept, major;
    int pin, securitynum;
    double salary, avg;
    long phone;
    boolean tenured;
    String cont = "y";
    String QUIT = "Q";
    int i=0;
    int j=0;
    int k=0;

    System.out.println("Enter C for CollegeEmployee entry or F for Faculty entry or S for Student
entry");
    response = sc.nextLine();
    while(response!="Q"){
        if(response.equals("C")){
            while(cont.equals("y"))
            {
                c[i]=new CollegeEmployee();
                System.out.println("Enter first name");
                fname = sc.nextLine();
                c[i].setFirstName(fname);
                System.out.println("Enter last name");
                lname = sc.nextLine();
                c[i].setLastName(lname);
                System.out.println("Enter address");
                address = sc.nextLine();

```

```

c[i].setAddress(address);

System.out.println("Enter pin code");

pin = sc.nextInt();

c[i].setPincode(pin);

System.out.println("Enter phone number");

phone = sc.nextLong();

c[i].setphonenum(phone);

System.out.println("Enter security number");

securitynum = sc.nextInt();

c[i].setSnum(securitynum);

System.out.println("Enter Anual salary");

salary = sc.nextDouble();

sc.nextLine();

c[i].setAsal(salary);

System.out.println("Enter department name");

dept = sc.nextLine();

c[i].setdept(dept);

System.out.println("Enter more entries? (y/n)");

cont = sc.nextLine();

if(i==3){

    System.out.println("Entered maxium number of entries");

    cont="n";

}

i++;

}

System.out.println("Enter C for CollegeEmployee entry or F fot Faculty entry or S for Student
entry");

response = sc.nextLine();

cont ="y";

}

if(response.equals("F")){

```

```
while(cont.equals("y"))
{
    f[j]=new Faculty();
    System.out.println("Enter first name: ");
    fname = sc.nextLine();
    f[j].setFirstName(fname);
    System.out.println("Enter last name");
    lname = sc.nextLine();
    f[j].setLastName(lname);
    System.out.println("Enter address");
    address = sc.nextLine();
    f[j].setAddress(address);
    System.out.println("Enter pin code");
    pin = sc.nextInt();
    f[j].setPincode(pin);
    System.out.println("Enter phone number");
    phone = sc.nextLong();
    f[j].setphonenum(phone);
    System.out.println("Enter security number");
    securitynum = sc.nextInt();
    f[j].setSnum(securitynum);
    System.out.println("Enter Anual salary");
    salary = sc.nextDouble();
    f[j].setAsal(salary);
    System.out.println("Enter department name");
    dept = sc.nextLine();
    sc.nextLine();
    f[j].setdept(dept);
    System.out.println("Enter true if tenured or enter false if not tenured");
    tenured=sc.nextBoolean();
    sc.nextLine();
}
```

```

f[j].setTenured(tenured);

System.out.println("Enter more entries ? (y/n)");

cont = sc.nextLine();

if(j==2){

    System.out.println("Entered maxium number of entries");

    cont="n";

}

j++;

}

System.out.println("Enter C for CollegeEmployee entry or F fot Faculty entry or S for Student
entry");

response = sc.nextLine();

cont ="y";

}

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

while(cont.equals("y"))

{

    s[k]=new Student();

    System.out.println("Enter first name");

    fname = sc.nextLine();

    s[k].setFirstName(fname);

    System.out.println("Enter last name");

    lname = sc.nextLine();

    s[k].setLastName(lname);

    System.out.println("Enter address");

    address = sc.nextLine();

    s[k].setAddress(address);

    System.out.println("Enter pin code");

    pin = sc.nextInt();

    s[k].setPincode(pin);

    System.out.println("Enter phone number");

```

```

        phone = sc.nextLong();
        sc.nextLine();
        s[k].setphonenum(phone);
        System.out.println("Enter major");
        major =sc.nextLine();
        s[k].setMajor(major);
        System.out.println("Enter average");
        avg=sc.nextDouble();
        sc.nextLine();
        s[k].setAvg(avg);
        System.out.println("Enter more entries? (y/n)");
        cont = sc.nextLine();
        if(k==6){
            System.out.println("Entered maxium number of entries");
            cont="n";
        }
        k++;
    }
    System.out.println("quiting");
    response="Q";

}

}

if(response.equals("Q")){

    for (int p =0;p<i;p++){
        c[p].display();
    }
    for(int q=0;q<j;q++){
        f[q].display();
    }
}

```

```
    }  
    for(int r=0;r<k;r++){  
        s[r].display();  
    }  
}  
int h = 4-i;  
if(h!=0){  
    System.out.println(h +" CollegeEmployee data not Entered");  
}  
h=3-j;  
if(h!=0){  
    System.out.println(h+ " Faculty data not Entered");  
}  
h=7-k;  
if(h!=0){  
    System.out.println(h + " Student data not Entered");  
}  
  
}  
}
```



C:\Windows\System32\cmd.exe

```
D:\19BCD7088>javac Person.java
D:\19BCD7088>javac CollegeEmployee.java
D:\19BCD7088>javac Faculty.java
D:\19BCD7088>javac Student.java
D:\19BCD7088>javac CollegeList.java
D:\19BCD7088>java Collegelist
Enter C for CollegeEmployee entry or F for Faculty entry or S for Student entry
C
Enter first name
lokesh
Enter last name
nara
Enter address
hyderabad
Enter pin code
500001
Enter phone number
9555560461
Enter security number
25
Enter Annual salary
1000000
Enter department name
CSE
Enter more entries? (y/n)
y
Enter first name
nilesh
Enter last name
kota
Enter address
guntur
Enter pin code
522001
Enter phone number
9755552147
Enter security number
64
Enter Annual salary
800000
Enter department name
ECE
Enter more entries? (y/n)
n
Enter C for CollegeEmployee entry or F for Faculty entry or S for Student entry
F
Enter first name:
ron
Enter last name
weasley
Enter address
vijayawada
Enter pin code
502355
Enter phone number
7555799528
Enter security number
54
Enter Annual salary
1100000
Enter department name
CSE
Enter true if tenured or enter false if not tenured
true
```

```
true
Enter more entries ? (y/n)
n
Enter C for CollegeEmployee entry or F for Faculty entry or S for Student entry
S
Enter first name
bhuvanesh
Enter last name
valiveti
Enter address
agiripalli
Enter pin code
521211
Enter phone number
9014914993
Enter major
CSE
Enter average
8.89
Enter more entries? (y/n)
n
quitting
lokesh nara's Address is hyderabad,500001.Mobile number is 9555560461
His Security number is 25 and Annual salary is 1000000.0. He belongs to Department of CSE.
nilesh kota's Address is guntur,522001.Mobile number is 9755552147
His Security number is 64 and Annual salary is 800000.0. He belongs to Department of ECE.
ron weasley's Address is vijayawada,502355.Mobile number is 7555799528
His Security number is 54 and Annual salary is 1100000.0. He belongs to Department of .
Faculty member is tenured
bhuvanesh valiveti's Address is agiripalli,521211.Mobile number is 9014914993
His major is CSE and his average is 8.89
2 CollegeEmployee data not Entered
2 Faculty data not Entered
6 Student data not Entered

D:\19BCD7088>
```