

## חומרים שהוכנו על-ידי משתתפי קורס מורים מובילים תשע"א

ניתן להשתמש בחומרים לצורך הוראה בלבד.

**לא ניתן לפרסם את החומרים או לעשות בהם כל שימוש מסחרי**

ללא קבלת אישור מראש מצוות הפיתוח

היבטים שונים של ייצוג מבני נתונים בעזרת מערך של תורים / מערך של מחסניות

תוך שימת דגש על גנריות מבני הנתונים

**שאלה 5 – משרד התיווך "דירה לכל"**

**כתיבה ועריכה:**

**שרה וגנר**

**דורית ליקרמן**

**אהובה שפרלינג**

## שאלה 5

משרד התיווך "דירה לכל" עוסק בדירות להשכרה. למשרד מאגר דירות ומאגר לקוחות. אדם המעוניין לשכור דירה, חייב להיות לקוח או להירשם כלקוח. לקוח יכול לקבל פרטי דירה הקיימת כבר במאגר הדירות במשרד התיווך או פרטי דירה שרק התקבלו במשרד ועדיין לא הוכנסו למאגר. עבור קבלת פרטי הדירה בפעם הראשונה משלם הלקוח 100 ש"ח לבעלי המשרד. כל צפייה חוזרת בפרטי דירה היא חינם. לדוגמא: לקוח שצפה ב 3 דירות שונות ישלם 300 ש"ח ללא קשר למספר הפעמים שצפה בפרטי כל דירה. משרד התיווך שומר את פרטי הדירות וכן את המידע על לקוחותיו, באופן הבא:

לקוח - Customer	
name	שם
id	ת"ז
tel	טלפון
codeList	רשימת קודי הדירות השונות שהלקוח צפה בהן.
pay	תשלום לחברה

דירה - Flat	
Code	קוד הדירה (יחודי לדירה)
City	עיר
Region	אזור
Rooms	מספר חדרים
Details	פרטים נוספים

מאגר משרד התיווך "דירה לכל" מיוצג ע"י:

RentingFlats
סה"כ תקבולי המשרד
אוסף הלקוחות מטיפוס Customer
אוסף הדירות מטיפוס Flat

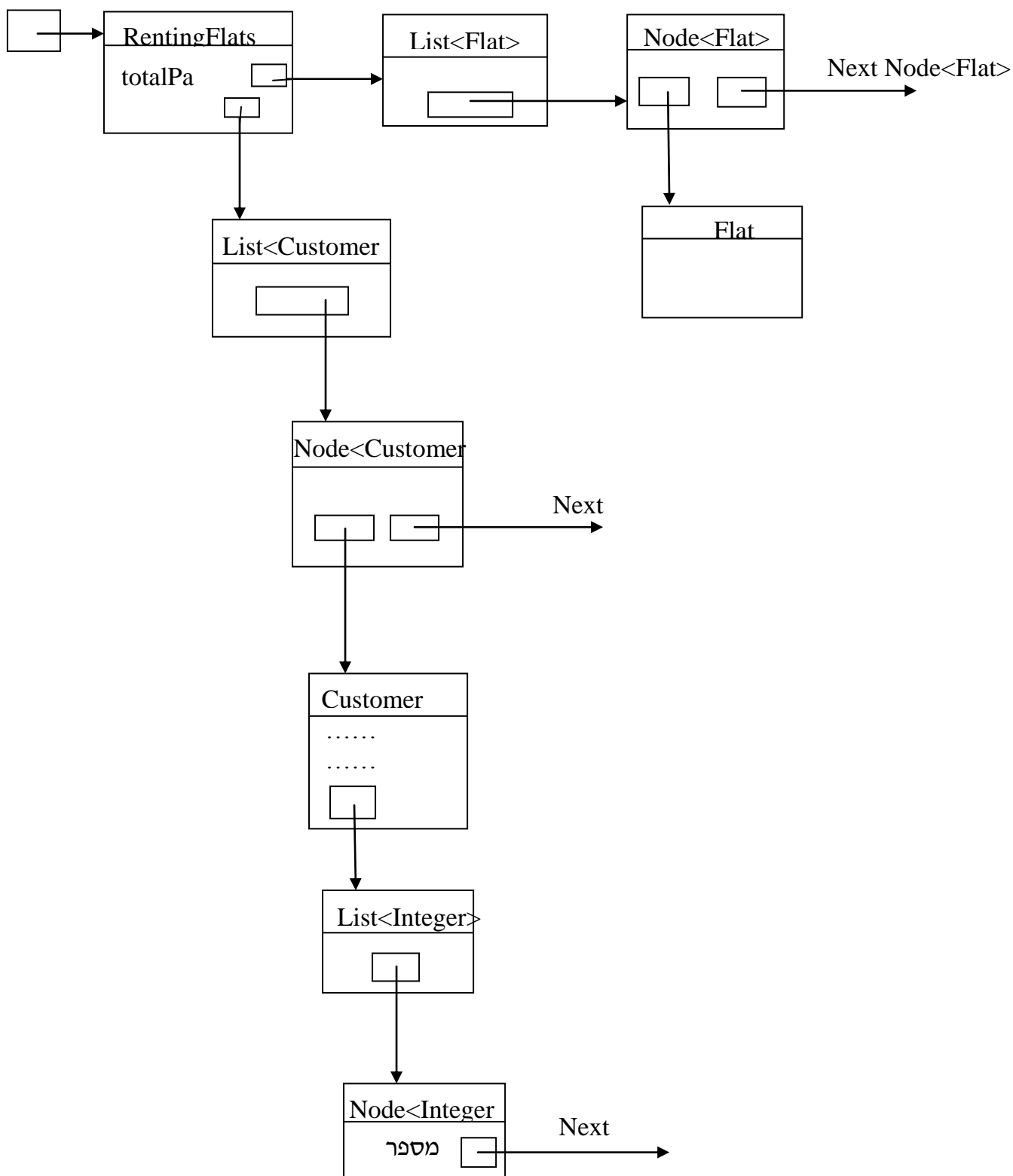
- א. (1) רשום את כותרת המחלקה "דירה" – Flat, את תכונותיה ורשום תיעוד ליד כל תכונה.  
(2) רשום את כותרת המחלקה "לקוח" – Customer, את תכונותיה ורשום תיעוד ליד כל תכונה.  
(3) רשום את כותרת המחלקה "דירה לכל" – RentingFlats, את תכונותיה ורשום תיעוד ליד כל תכונה.

הנח שכל פעולות ה-get/set במחלקות השונות מוגדרות.

אם אתה מגדיר פעולות עזר נוספות, עליך לממש אותן ולציין באיזו מחלקה נוספו.

- ב. כתוב פעולה המקבלת פרטי דירה המוצעת ללקוח ומעדכנת את משרד התיווך "דירה לכל" בהתאם לכללי המשרד הכתובים לעיל.  
(1) באיזו מחלקה יש להוסיף פעולה / ות?  
(2) ממש את הפעולה / ות שהגדרת בסעיף (1).
- ג. לצורך סקרים פנימיים של המשרד "דירה לכל" התעורר צורך במידע לגבי העיר בה יש דרישה גבוהה להשכרת דירות.  
כתוב פעולה popularTown המחזירה את שם העיר בה הביקוש להשכרה הוא הגבוה ביותר.  
(1) באיזו מחלקה יש להוסיף את הפעולה?  
(2) ממש את הפעולה.
- ד. מהי סיבוכיות הפעולה popularTown? נמק.

מבנה הרשימות והקשרים:



```
package rentFlats;
import unit4.collectionsLib.List;
import unit4.collectionsLib.Node;

public class Customer
{
    private String name;
    private String id;
    private String phone;
    private List<Integer> codeList;
    private Double pay;

    public Customer(String name, String id, String tel)
    {
        this.name=name;
        this.id=id;
        this.phone=tel;
        this.codeList=new List<Integer>();
        this.pay=0.0;
    }

    public String getName()
    {
        return this.name;
    }

    public String getId()
    {
        return this.id;
    }

    public String getPhone()
    {
        return this.phone;
    }

    public List<Integer> getCodeList()
    {
        return this.codeList ;
    }

    public double getPay()
    {
        return this.pay;
    }

    public void setPay(double pay)
    {
        this.pay=pay;
    }

    public int addFlat(Flat f)
    {
        Node<Integer> pos=this.codeList.getFirst();
        while ((pos!=null) && (pos.getInfo()!=f.getCode()))
            pos=pos.getNext();
    }
}
```

```
        if (pos==null)
        {
            this.codeList.insert(pos, f.getCode());
            this.pay++;
            return 1;
        }
        return 0;
    }
    public String toString()
    {
        Return " \n"+this.name+", id="+this.id+", phone="+this.phone+
            ", Paymentr="+this.pay+"\n flats seen codes:\n"+this.codeList;
    }
}
package rentFlats;

public class Flat
{
    private int code;
    private String city;
    private String region;
    private int rooms;
    private String details;

    public Flat(int c, String city, String reg, int room, String det)
    {
        this.code=c;
        this.city=city;
        this.region=reg;
        this.details=det;
        this.rooms=room;
    }

    public int getCode()
    {
        return this.code;
    }

    public int getRooms()
    {
        return this.rooms;
    }

    public String getCity()
    {
        return this.city;
    }

    public String getRegion()
    {
        return this.region;
    }
}
```

```
public String getDetails()
{
    return this.details;
}

public String toString()
{
    Return "\n"+this.code+", "+this.city+", "+this.region+", rooms="
        +this.rooms+this.details;
}
}
```

```
package rentFlats;
```

```
import unit4.collectionsLib.List;
import unit4.collectionsLib.Node;
```

```
public class RentingFlats
{
    private double totalPay;
    private List<Customer> customers;
    private List<Flat> flats;

    public RentingFlats()
    {
        this.totalPay=0.0;
        this.flats=new List<Flat>();
        this.customers=new List<Customer>();
    }

    public List<Flat> getFlats()
    {
        return this.flats;
    }

    public List<Customer> getCustomers()
    {
        return this.customers;
    }

    public double getTotalPay()
    {
        return this.totalPay;
    }

    public void insertFlat(Flat f, Customer c)
    {
        Node<Customer> posC=insertCustomer(c);
        addFlat(f);
        this.totalPay=this.totalPay+posC.getInfo().addFlat(f);
    }
}
```

```
public void addFlat(Flat f)
{
    Node<Flat> pos=this.flats.getFirst();
    while ((pos!=null) && (pos.getInfo().getCode()!=f.getCode()))
        pos=pos.getNext();
    if (pos==null)
        this.flats.insert(pos, f);
}

public Node<Customer> insertCustomer(Customer c)
{
    Node<Customer> pos=this.customers.getFirst();
    while ((pos!=null) && (pos.getInfo().getId().compareTo(c.getId())!=0))
        pos=pos.getNext();
    if (pos==null)
        pos=this.customers.insert(pos, c);
    return pos;
}

private List<String> townList()
{
    List<String> towns=new List<String>();
    Node<Flat> posF=this.flats.getFirst();
    while (posF!=null)
    {
        Node<String> posT=towns.getFirst();
        while ((posT!=null) &&
            (posT.getInfo().compareTo(posF.getInfo().getCity())!=0))
            posT=posT.getNext();
        if (posT==null)
            posT=towns.insert(posT, posF.getInfo().getCity());
        posF=posF.getNext();
    }
    return towns;
}

public String popularTown()
{
    List<String> towns=townList();
    Node<String> posT=towns.getFirst();
    int maxTown=0, currentTown;
    String maxTownName="";
    while (posT!=null)
    {
        currentTown=0;
        Node<Flat> posF=this.flats.getFirst();
        while (posF!=null)
        {
            if (posF.getInfo().getCity().compareTo(posT.getInfo())==0)
            {
```

```
int code=posF.getInfo().getCode();
Node<Customer> posC=this.customers.getFirst();
while (posC!=null)
{
    Node<Integer> pos= posC.getInfo().getCodeList().getFirst();
    while ((pos!=null) &&(pos.getInfo()!=code))
        pos=pos.getNext();
    if (pos!=null)
        currentTown++;
        posC=posC.getNext();
    }
}
posF=posF.getNext();
}
if (currentTown>maxTown)
{
    maxTown=currentTown;
    maxTownName=posT.getInfo();
}
posT=posT.getNext();
}
return maxTownName;
}
}

public String toString()
{
    return "Total Pay="+this.totalPay+
        "\n"+"Customers:\n"+this.customers+"\nFlats:\n"+this.flats;
}
}
```

```
package rentFlats;
//import unit4.collectionsLib.List;
public class TestRent
{
    /**
    @ * param args
    */
    public static void main(String[] args)
    {
        Flat f1=new Flat (1,"Haifa","Carmel",4,"");
        Flat f2=new Flat (2,"Tel Aviv","Ramat Aviv",3,"");
        Flat f3=new Flat (3,"Haifa","Bat Galim",3,"");
        Flat f4=new Flat (4,"Jerusalem","Ramot",5,"");
        Flat f5=new Flat (5,"Tel Aviv","",4,"");
        Flat f6=new Flat (6,"Haifa","Ahuza",6,"");
        Flat f7=new Flat (7,"Jerusalem","Gilo",5,"");
        Flat f8=new Flat (8,"Jerusalem","Bait Vagan",5,"");
    }
}
```



```
Flat f9=new Flat (9,"Haifa","Carmel",3,"");
Flat f10=new Flat (10,"Haifa","Hadar",2,"");
Flat f11=new Flat (11,"Haifa","Carmel",5,"");

Customer c1=new Customer("Rachel Ron","011222","048345678");
Customer c2=new Customer("Ron Magen","0221222","048120088");
Customer c3=new Customer("Sapir Ruth","5311222","048244555");
Customer c4=new Customer("Sharon Gad","005389","048110022");
Customer c5=new Customer("Mayer Or","212121","048200444");

RentingFlats RF=new RentingFlats();
RF.addFlat(f1);
RF.addFlat(f2);
RF.addFlat(f3);
RF.addFlat(f4);
RF.addFlat(f5);
RF.addFlat(f6);
RF.addFlat(f7);
RF.addFlat(f8);
RF.addFlat(f9);
RF.addFlat(f10);
RF.addFlat(f11);

RF.insertCustomer(c1);
RF.insertCustomer(c2);
RF.insertCustomer(c3);
RF.insertCustomer(c4);
RF.insertCustomer(c5);

RF.insertFlat(f1, c1);
RF.insertFlat(f2, c1);
RF.insertFlat(f3, c2);
RF.insertFlat(f11, c2);
RF.insertFlat(f10, c3);
RF.insertFlat(f5, c3);
RF.insertFlat(f1, c4);
RF.insertFlat(f8, c4);
RF.insertFlat(f7, c5);
RF.insertFlat(f6, c5);
RF.insertFlat(f8, c1);
RF.insertFlat(f3, c2);
RF.insertFlat(f9, c3);
RF.insertFlat(f4, c4);
RF.insertFlat(f2, c1);
RF.insertFlat(f11, c1);
System.out.println(RF);
System.out.println(RF.popularTown());
```

```
}
}
```