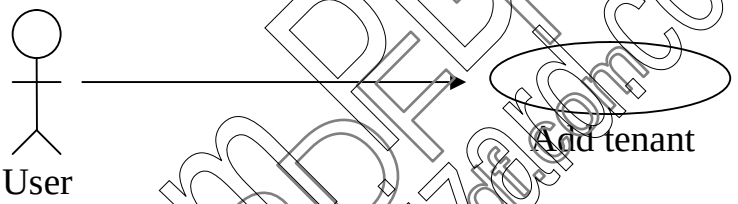
Abstract:

{ gz/R‡† œu ‘ „µ ••Ya’ •f/R
íÑåÔ Ó¢Ó¢ íáÚ Ñ¢Ì?¢ ÚŸí ì Ðá¢ Ï Í¢æ Ñ Ì¢ÊÓã äã åäæßÓã äæßÊä¢ äßã í¢åãã áß
Š|gz/RgS^SY RW„q ‡’g^SY †/RgS †Ur }Ya’ ‡RT^’ $S^ WZ„UT •Y†/R] †SgU/R
XS|e/R¾UŠ†QS• Š|gz/R..g• ãÓ?¢ íáÚ íæÊÎ ÊÝæÓÑÌ¢ÊÓã áß äÚ XS•„v†/R
`j †• W|S SU.edY †„/` †j ’] †SgU/R•f(, x„U/R gŒE/Rg^SY †„/Š’gŒE/R
XS|e/Ro gu g^SY † ¾f t|e/R¾^j Y ‡’g^SY †/SUŠ†QS•o gu ‡’g^SY †/R
. g^SY †„/Š’d’gSY/R

Use cases:

1- Use case**: add tenant**



الحاله البسيطه المباشره:

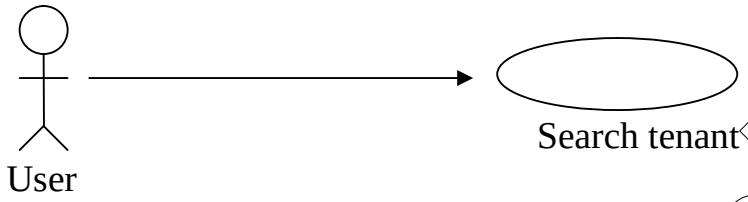
- المستخدم يضيف مستاجر جديد ،يرد النظام بتخزين بيانات المستخدم.
- المستخدم يكتب اسم المستاجر ورقم الغرفه ويضغط عل زر "add tenant"،فيقوم النظام بعرض عبارة "new tenant in room successfully add".

الحاله البديله:

- المستخدم يستفسر عن وجود غرفه فارغه، ويرد النظام برساله تحوي "true" او "false".
- المستخدم يدخل رقم الغرفه واسم المستاجر، فيقوم النظام بالتاكد من ادخال المستاجر والغرفه،اذاكان رقم الغرفه مدخل من قبل يرد النظام برسالة" room number is occupied"،او اذا كان رقم الغرفه اكبر من ارقام الغرف الموجوده يرد النظام رسالة" there are only n rooms".

DocuCom PDF Trial
Nitro PDF Trial
www.nitropdf.com

2-Use case: **search tenant**:



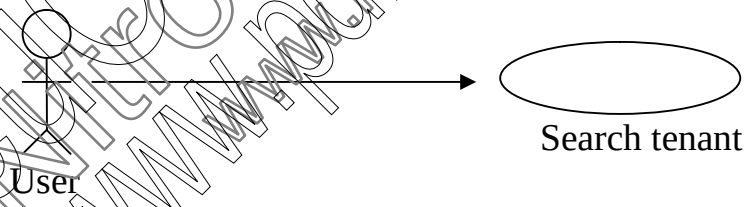
الحاله البسيطه المباشره:

- المستخدم يدخل رقم الغرفه، فيرد النظام بعرض رقم الغرفه واسم المستاجر.
- المستخدم يدخل رقم الغرفه وينقرعلى زر "search" ،فيقوم النظام بعرض اسم المستاجر ورقم الغرفه في "display area".

الحاله البديله:

- اذا ادخل المستخدم رقم غرفه خطا، يرد النظام برساله تفيد ان المستاجر غير موجود.
- المستخدم يستدعي "get payment method" ،يرد النظام بعرض رساله تحمل اخر دفعه للزبون.
- المستخدم يستدعي "calculate total method" ، يرد النظام بعرض رساله تحمل مجموع كل دفعات المستاجر كامله.

3-Use case: **remove tenant**:



الحاله البسيطه المباشره:

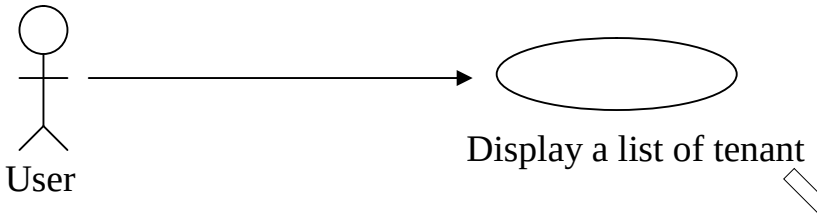
- المستخدم يدخل رقم الغرفه، فيرد النظام بعرض عبارة تم حذف المستاجر.
- المستخدم يدخل رقم الغرفه وينقرعلى زر "remove tenant" ،فيقوم النظام بعرض عبارة"tenant remove from room N"في"display area".

الحاله البديله:

- اذا ادخل المستخدم رقم غرفه وكانت الغرفه فارغه اصلا، يرد النظام بعرض عبارة " room number n is empty".
- المستخدم يتراجع عن الحذف المستاجر فينقر علي الزر"quit without save "،فيقوم النظام بارجاع المستخدم الي النموذج الرئيسي .

DocuCom PDF Trial
www.nitroPDF.com

4-Use case: **display a list of tenant**:



الحاله البسيطه المباشره:

- المستخدم وينقر على زر "display tenants" ،فيقوم النظام بعرض كل الغرف ومستاجريها في"display area".

الحاله البديله:

- المستخدم يقوم بالتاكد من حالة الغرف، يرد النظام بعرض عبارة " all room are empty ".

- المستخدم يقوم باستدعاء "total method"، يرد النظام برساله تحمل عدد المستاجرين.

5-Use case: **record payment**:

الحاله البسيطه المباشره:

- المستخدم يدخل رقم الغرفه والشهر والمبلغ وينقر زر "make payment"، يرد النظام بتسجيل الدفعه للمستاجر.

الحاله البديله:

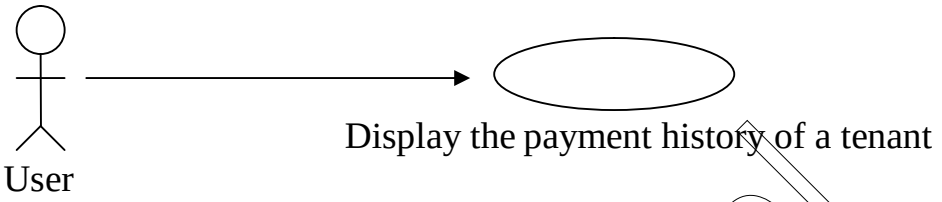
- المستخدم لم يدخل رقم الغرفه المبلغ والشهر,وضغط زر"make payment "، يرد النظام بعرض عبارة " room number,month,and amount must all be entered".

- المستخدم ادخل رقم غرفه غيرموجوده،يرد النظام بعرض عبارة " invalid room number"

- المستخدم يقوم باستدعاء "search method"، يرد النظام بعرض رقم الغرفه و اسم المستاجر.

DocuCom PDF Trial
NitroPDF Trial
www.nitropdf.com

6-Use case: **display the payment history of a tenant**:



الحاله البسيطه المباشره:

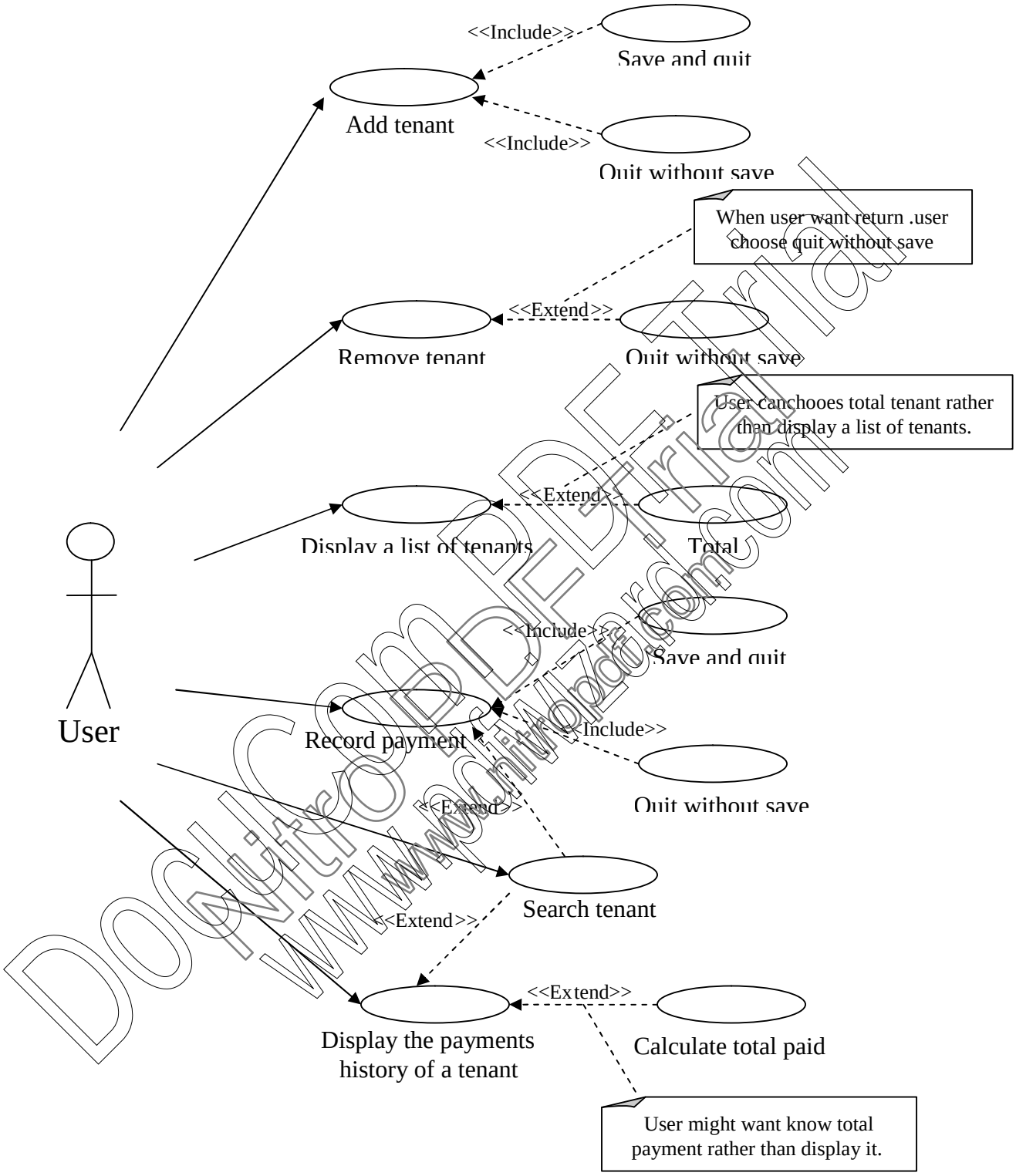
- المستخدم يدخل رقم الغرفه, وينقر علي زر "list payment"، يرد النظام بعرض الدفعات الشهريه للمستاجر كامله

الحاله البديله:

- المستخدم يستدعي "search method"، يرد النظام بعرض رقم الغرفه والمستاجر.
- المستخدم يستدعي "calculate total method"، يرد النظام بعرض رساله تحمل مجموع كل دفعات المستاجر كامله.
- المستخدم يدخل رقم الغرفه خطا ، يرد النظام بعرض رساله "invalid number room".
- المستخدم لم يدخل رقم الغرفه ، يرد النظام بعرض رساله "room number entered".

DocuCom PDF Trial
NitroPDF Trial
www.nitropdf.com

Use case diagrams:



DocuCom PDF Trial
Nitro PDF Trial
www.nitropdf.com

Discovering classes:

بعد قراءة الكود تم استخلاص 8 كلاسات من النظام المبني علي فكرة استئجار غرف في بيوت الطلبه ، وهي مكتوبه بلغة Java bluej. وهي كلاتي:

ObjectList:

وهو super class يحتوي methods التاليه، add() لإضافة كائن جديد الي المصفوفه object، isEmpty() لمعرفة هل المصفوفه فارغه، isFull() لمعرفة هل المصفوفه ممتلئه، getObject() للبحث عن كائن، remove() لحذف كائن .

TenantList:

وهو كلاس يرث من الكلاس objectlist يعمل علي تخزين قوائم المستأجرين، getTenant() ترجع كائن المستاجر من مصفوفة المستأجرين، searchTenant() تقوم بالبحث عن مستاجر، removeTenant() تقوم بحذف مستاجر.

PaymentList:

وهو ايضا يرث من الكلاس objectlist يقوم بتخزين الدفعات الخاصه بكل مستاجر، get Payment() ترجع اخر دفعه للمستأجر، calculateTotalPaid() يقوم بحساب جميع كامل الدفعات لمستاجر.

Tenant:

يمثل هذا الكلاس كائن المستاجر وله attributes التاليه name,room ، وله methods التاليه: getName() وتعيد اسم المستاجر، getRoom() تعيد رقم الغرفه، makePayment() تضيف دفعه جديده الي المستاجر.

Payment:

يمثل هذا الكلاس الدفعه وله attributes التاليه month, amount ، وله methods التاليه: getMonth() تعيد الشهر الدفعه، getAmount() تعيد قيمة الدفعه.

Hostel:

وهو الكلاس الذي يحمل interface الخاص بالنظام ويحتوي main method .

RunHostel:

وهو الكلاس الذي يقوم باستدعاء Interface من الكلاس Hostel .

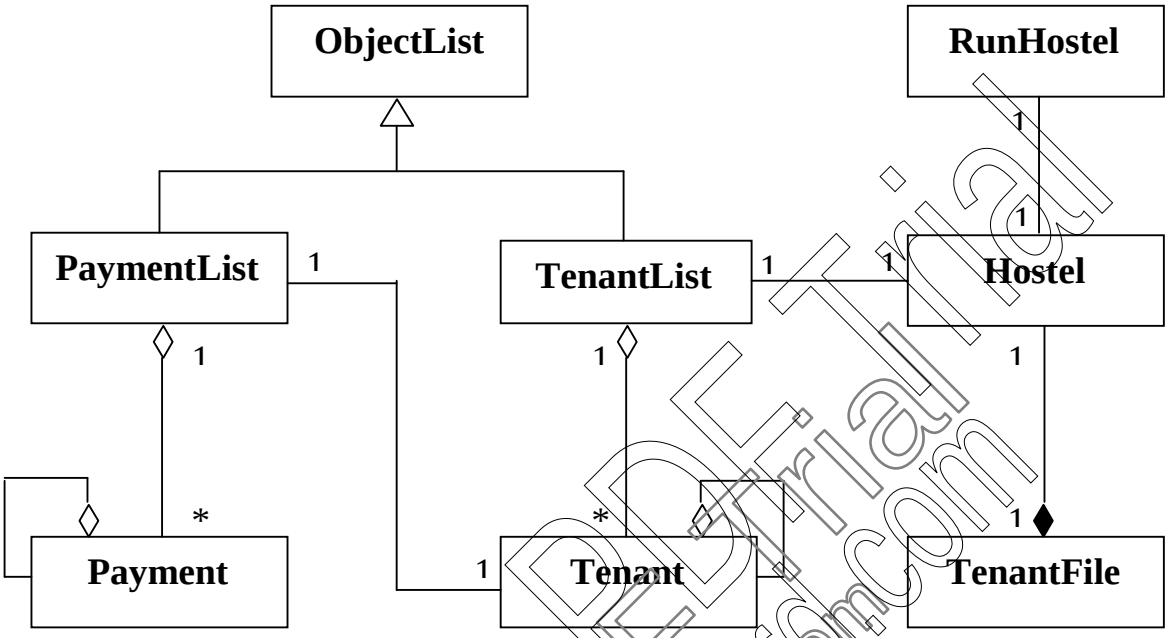
TenantFile:

وهو الكلاس الذي يقوم بالتخزين النهائي للملفات في النظام ،ويحوي methods التاليه: saveRecord() وهو يقوم بفتح الملف للتخزين فيه، readRecord() يفتح ملف المنظومه للقراءه منه.

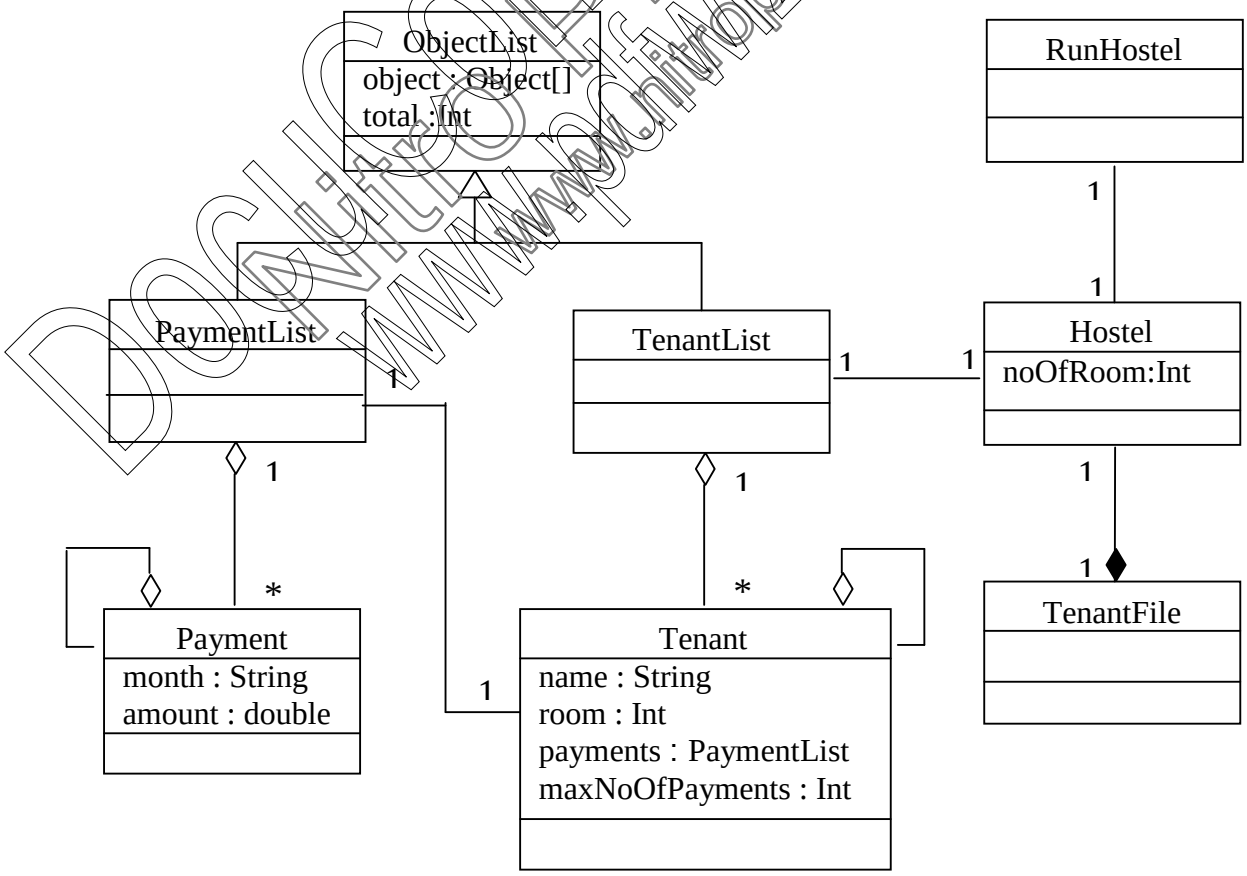
Doc1 PDF Trial
www.nitro-pdf.com

Class Diagrams :

- **Domain-level class diagrams:**

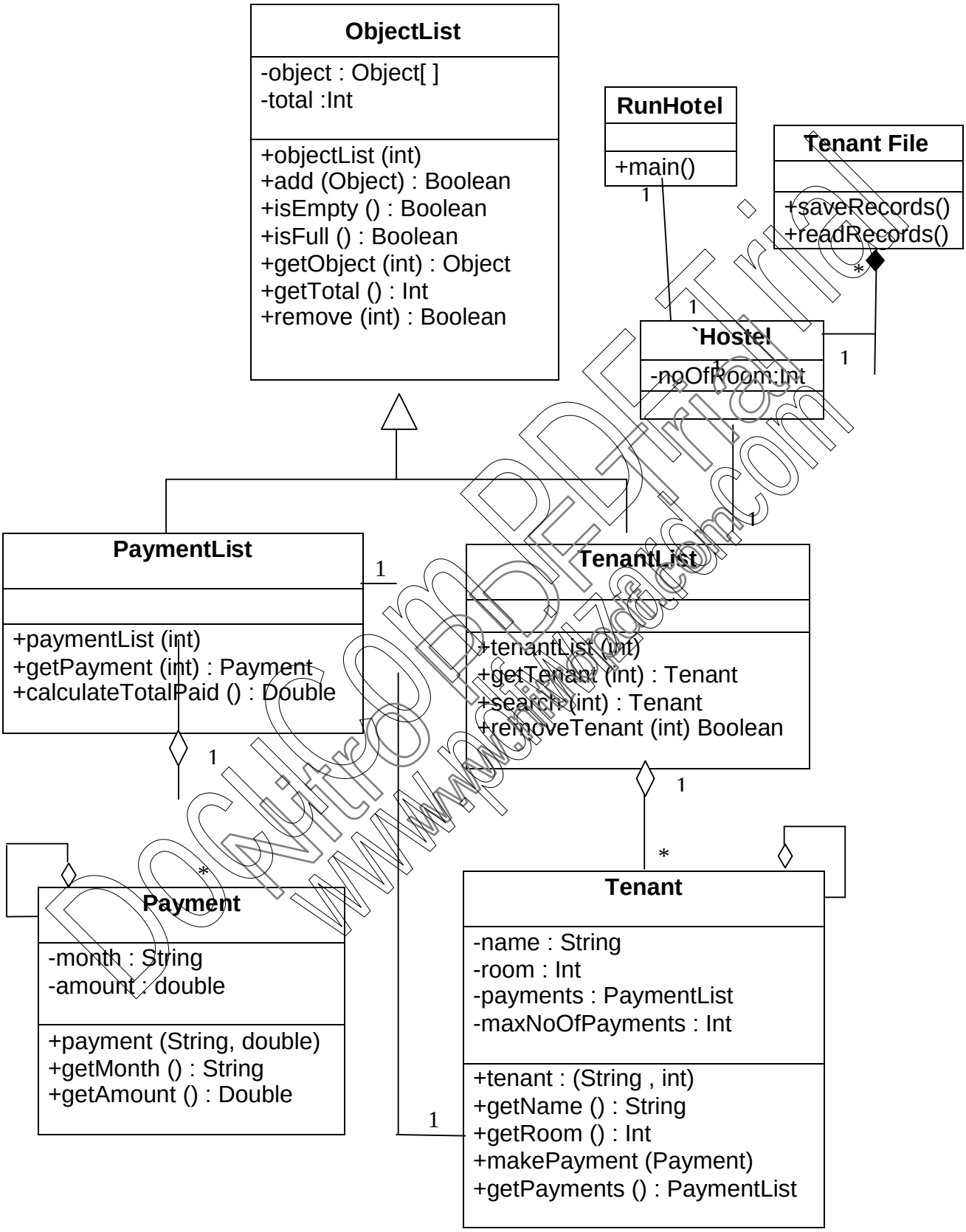


- **Analysis-level class diagrams:**



DocuPrintPDF Trial
www.nitrozdf.com

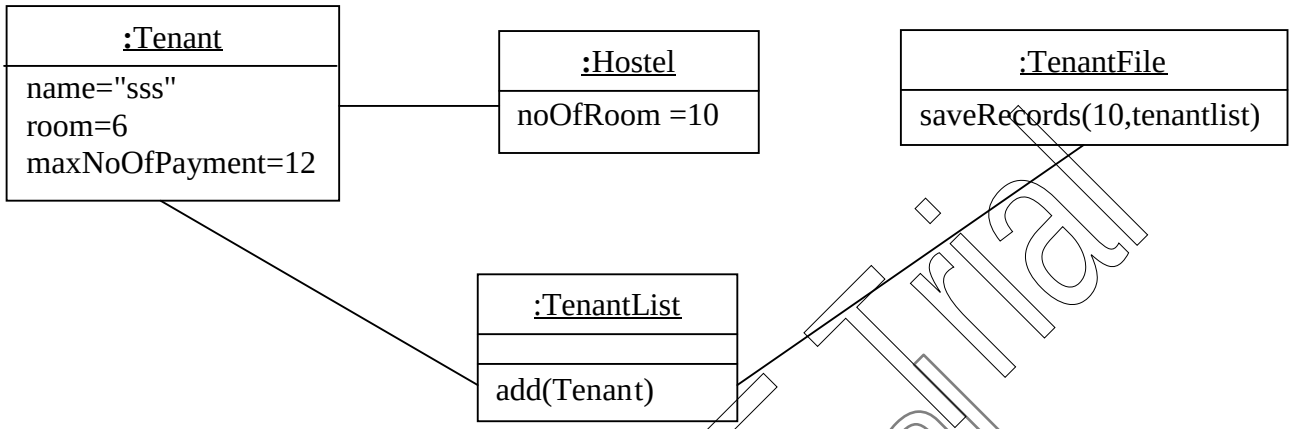
- **Design-level class diagrams:**



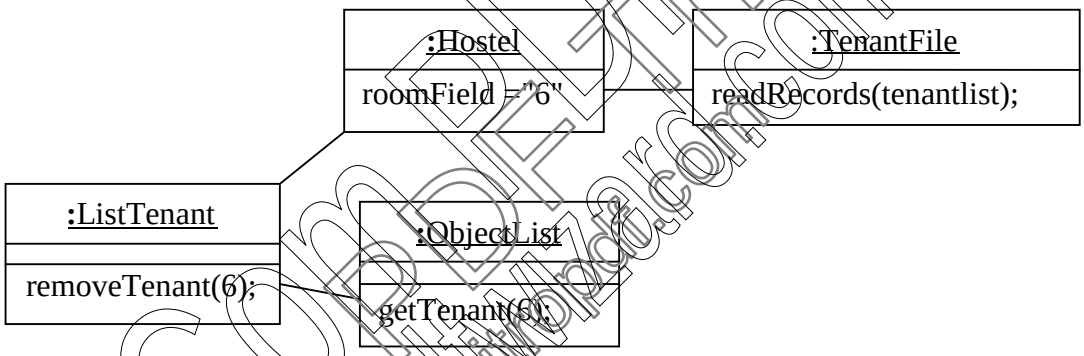
DocuPDF Trial
NitroPDF Trial
www.nitropdf.com

Object Diagrams:

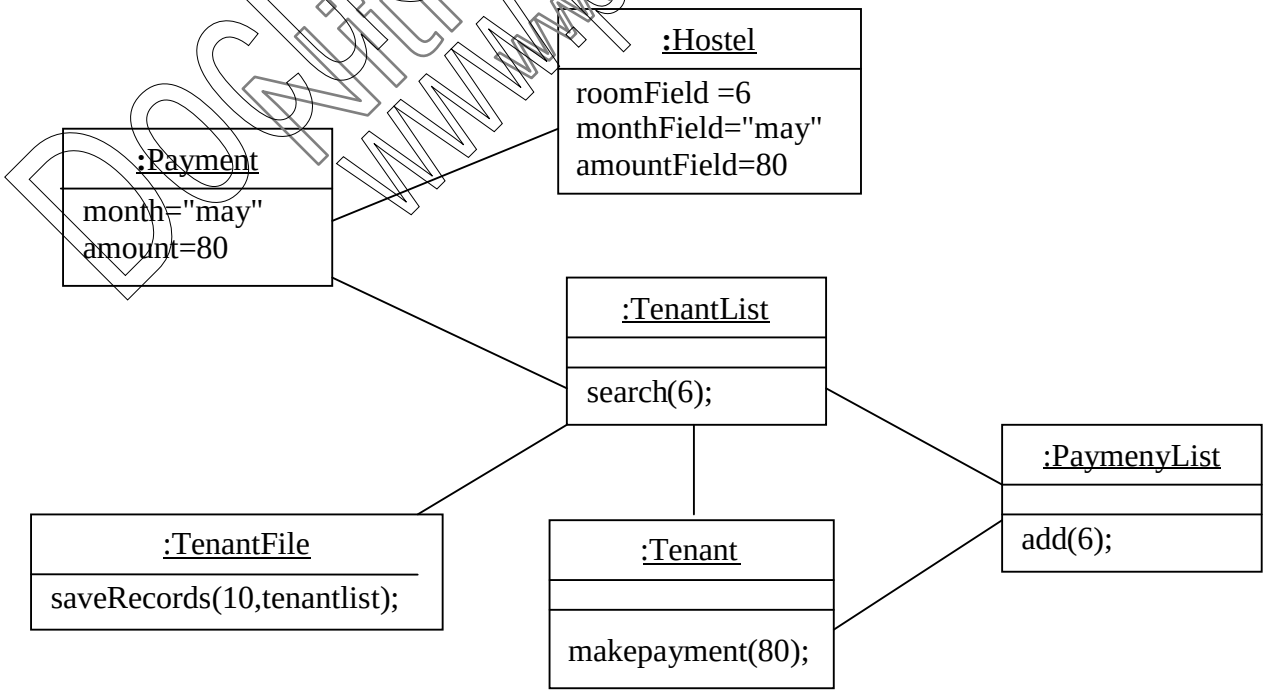
1-object diagram of add tenant:



2-object diagram of remove tenant:

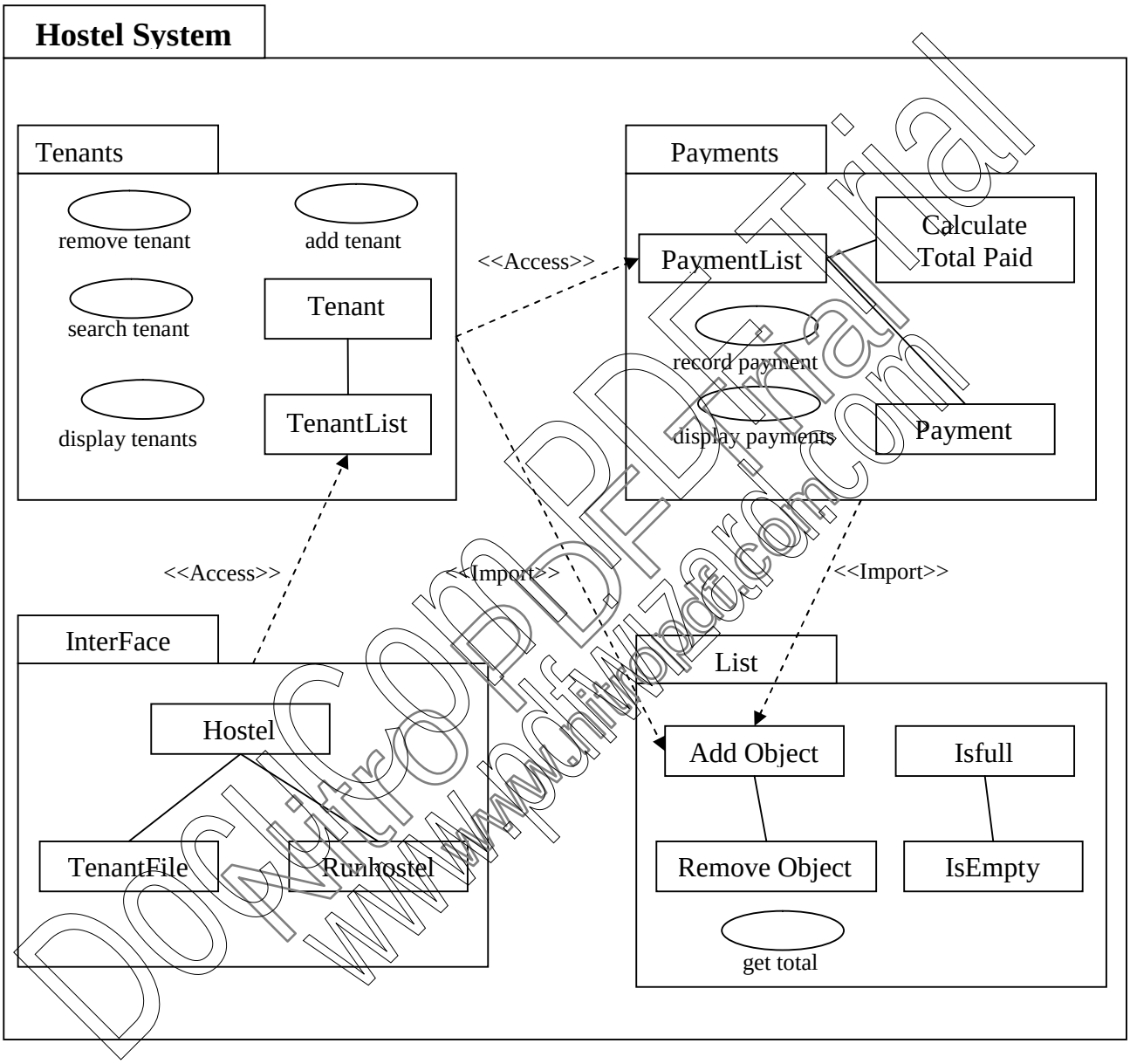


3-object diagram of record payment:



DocuCom PDF Trial
www.nitroPDF.com

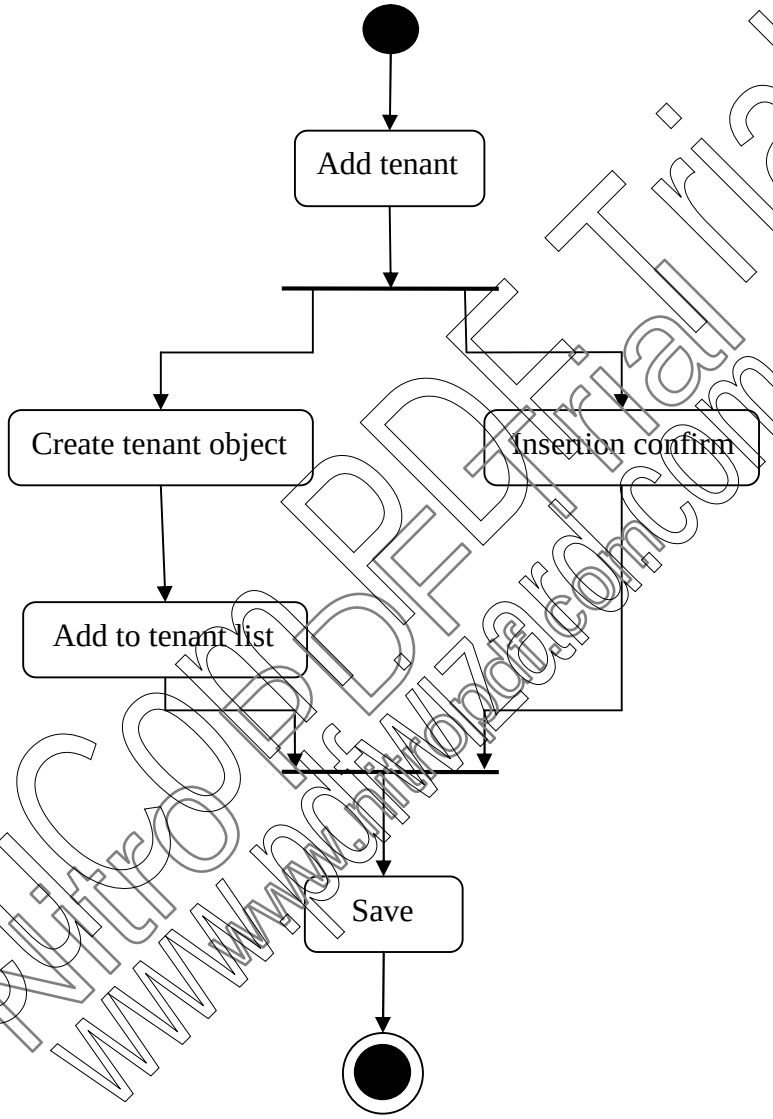
Package Diagrams:



DoNitroPDFTrial
www.nitrozdf.com

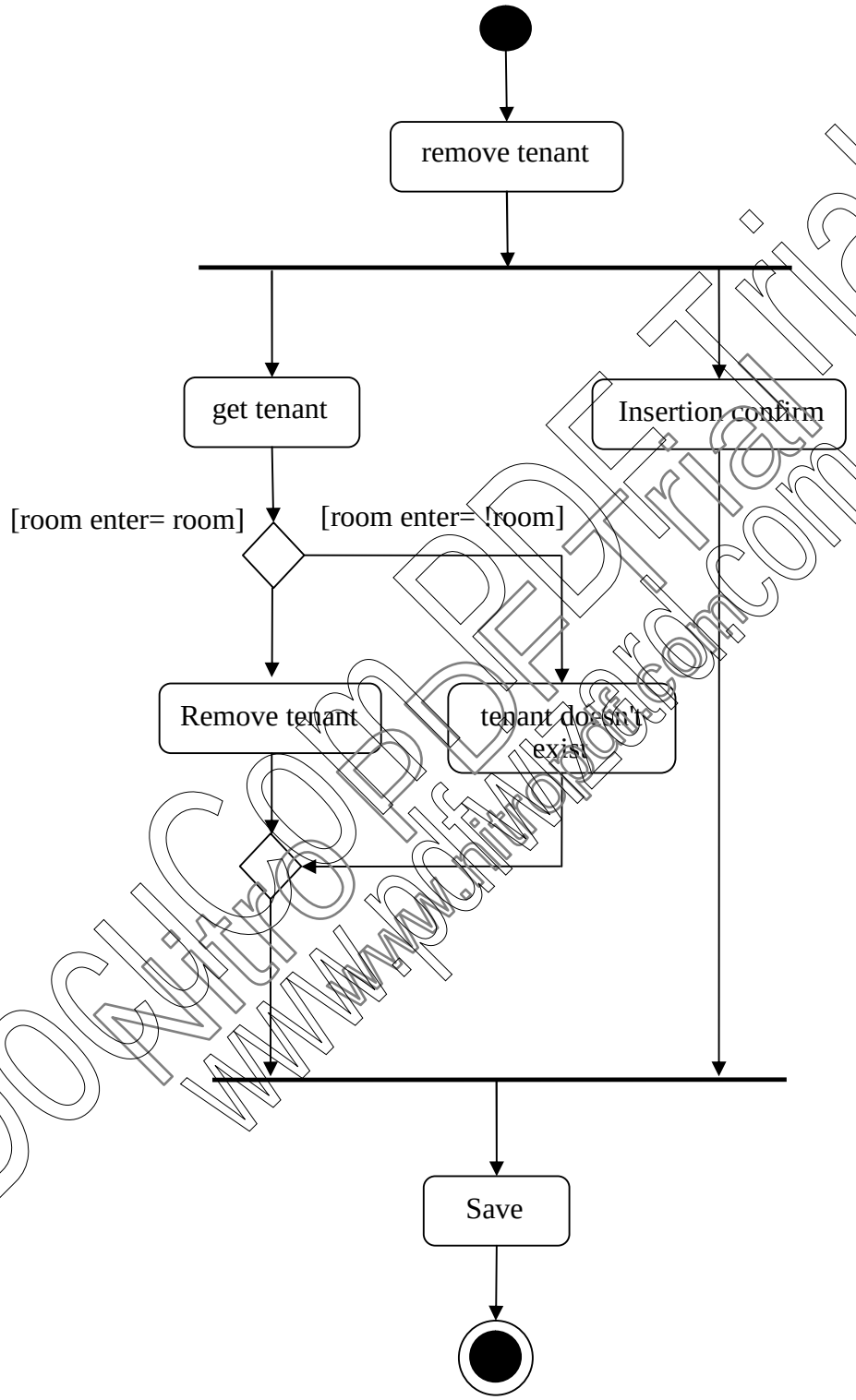
Activity diagrams:

- **activity diagram of add tenant**



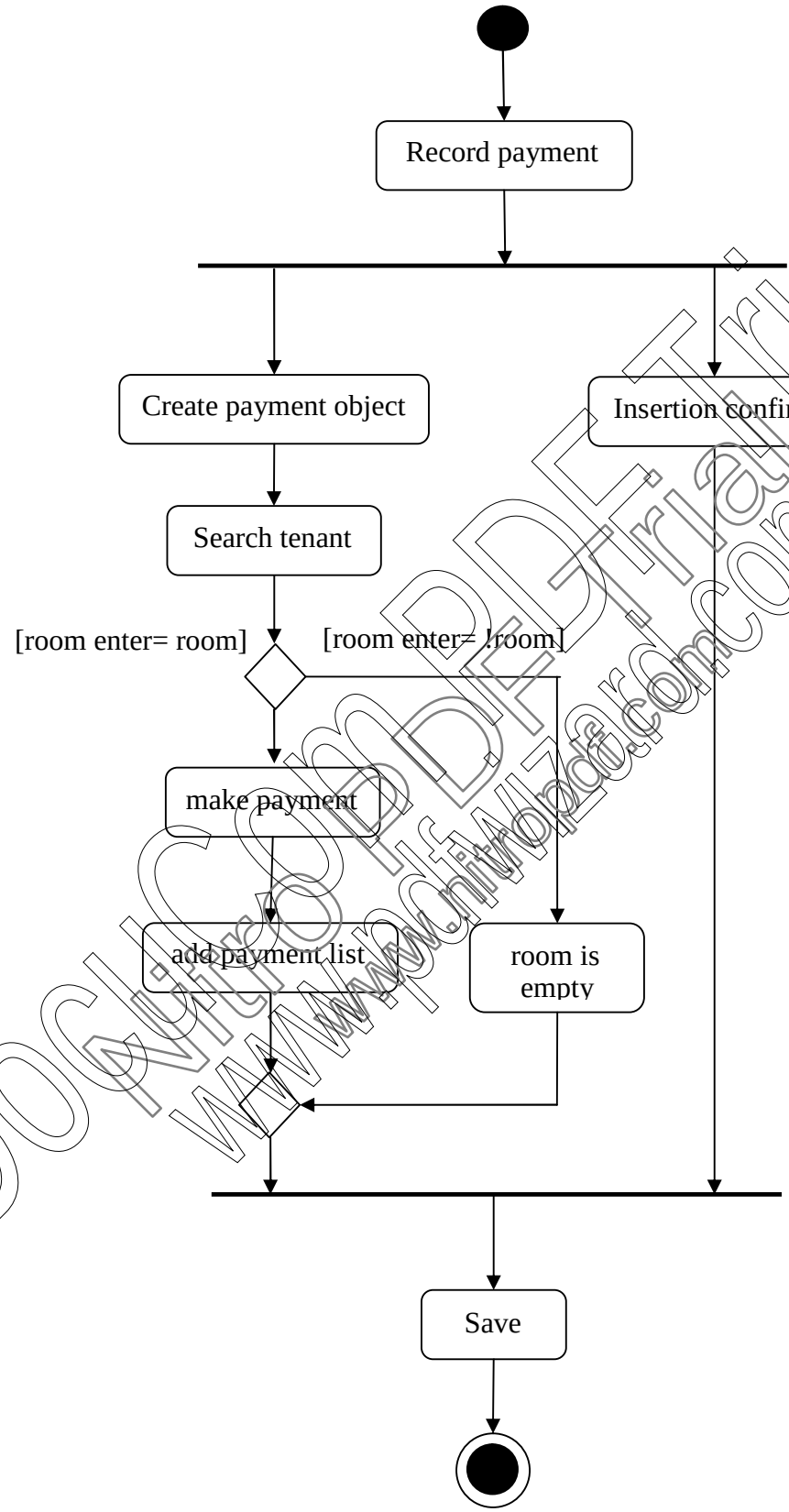
DocuCom PDF Trial
NitroPDF Trial
www.nitropdf.com

- **activity diagram of remove tenant**



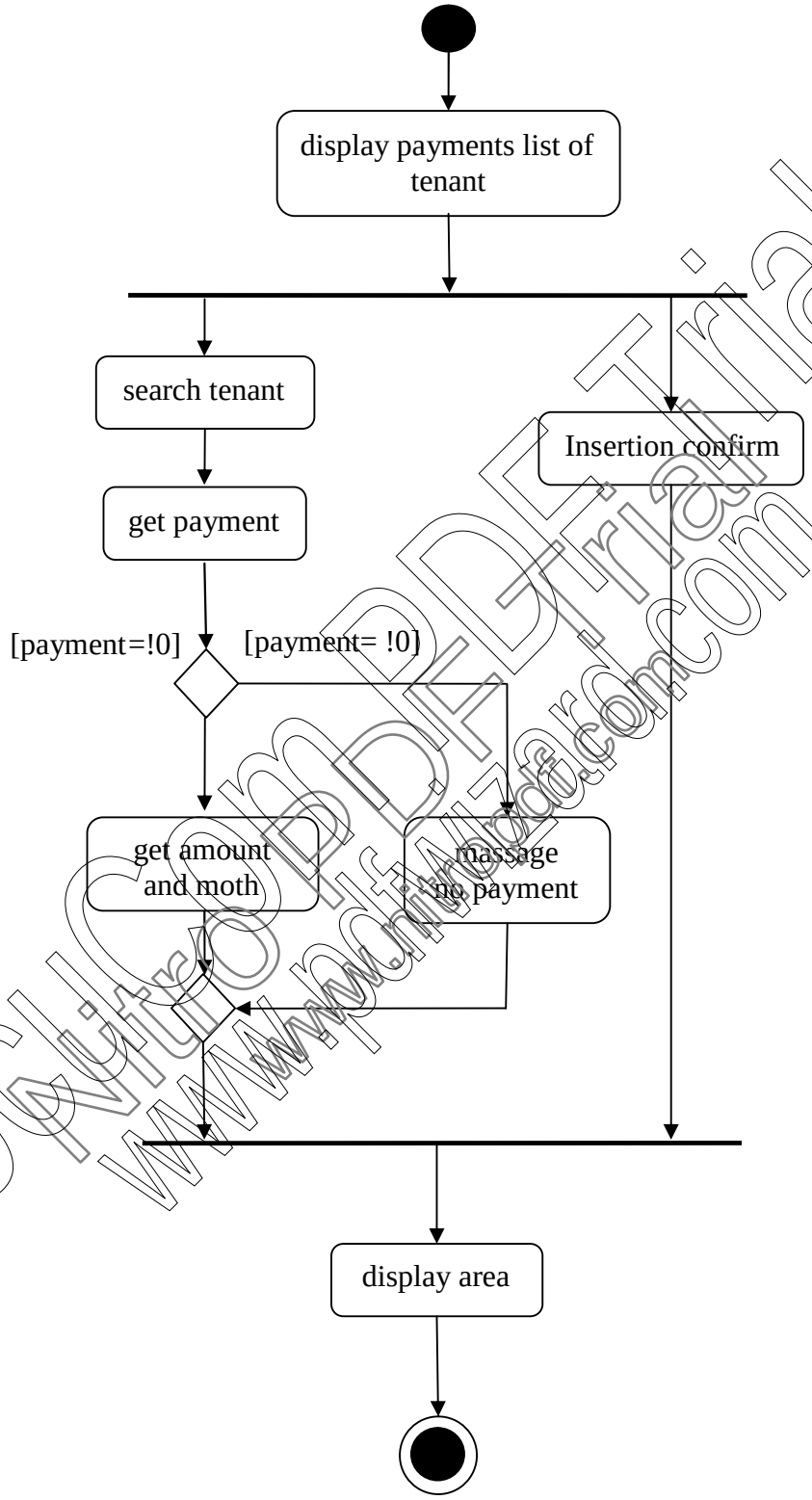
DocuCom PDF Trial
NitroPDF Trial
www.nitropdf.com

- **activity diagram of record payment**



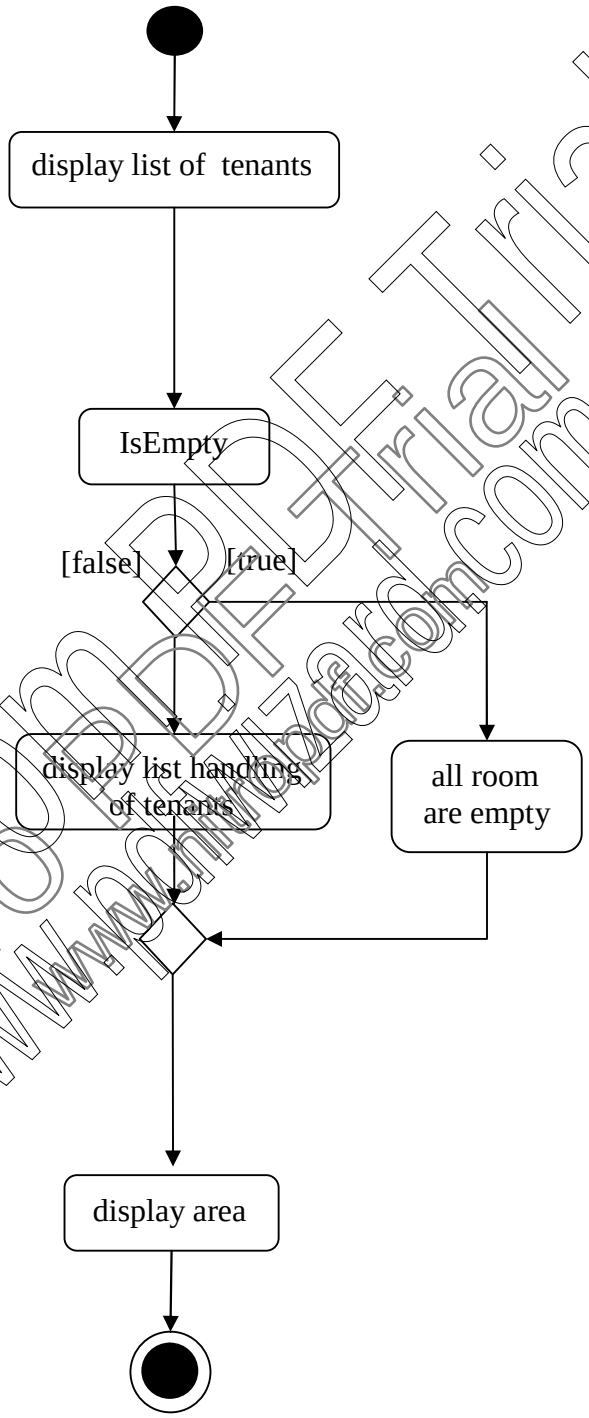
DocuCom PDF Trial
Nitro PDF Trial
www.nitropdf.com

- **activity diagram of display payments list of tenant**



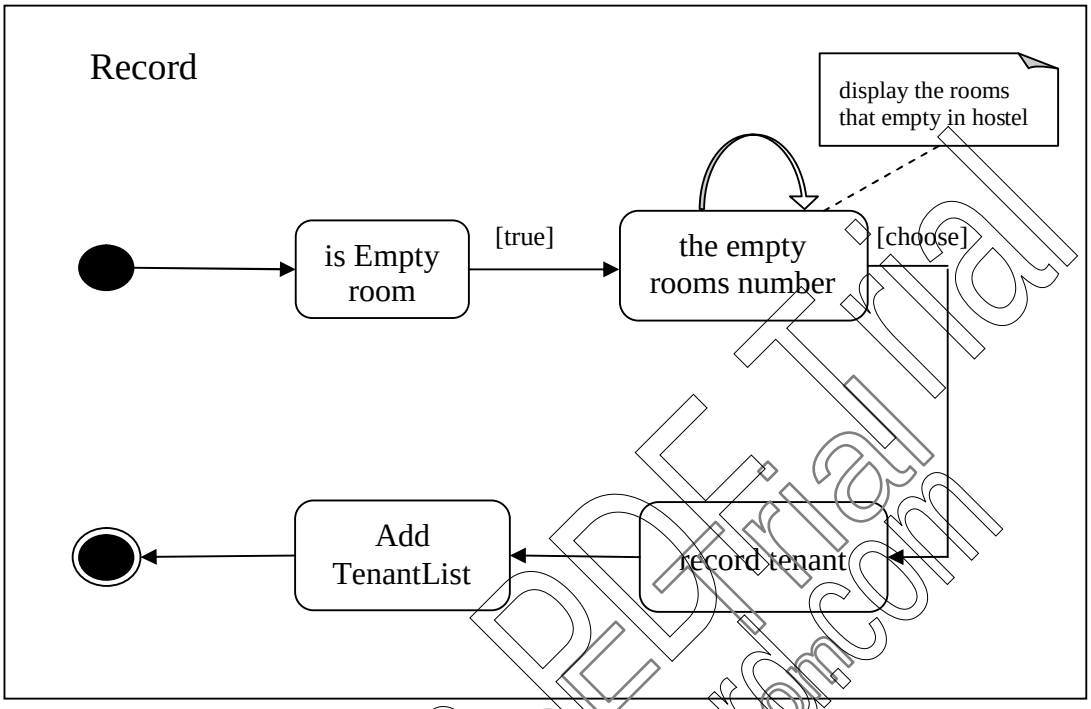
DocuCom PDF Trial
NitroPDF Trial
www.nitropdf.com

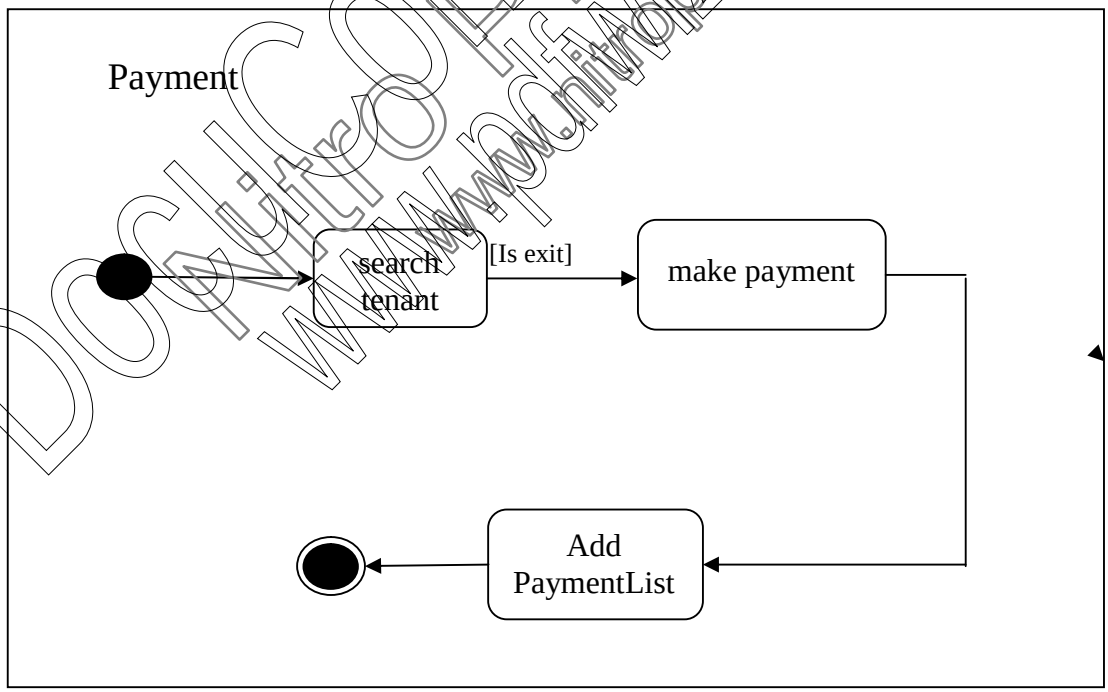
- **activity diagram of display list of tenants**



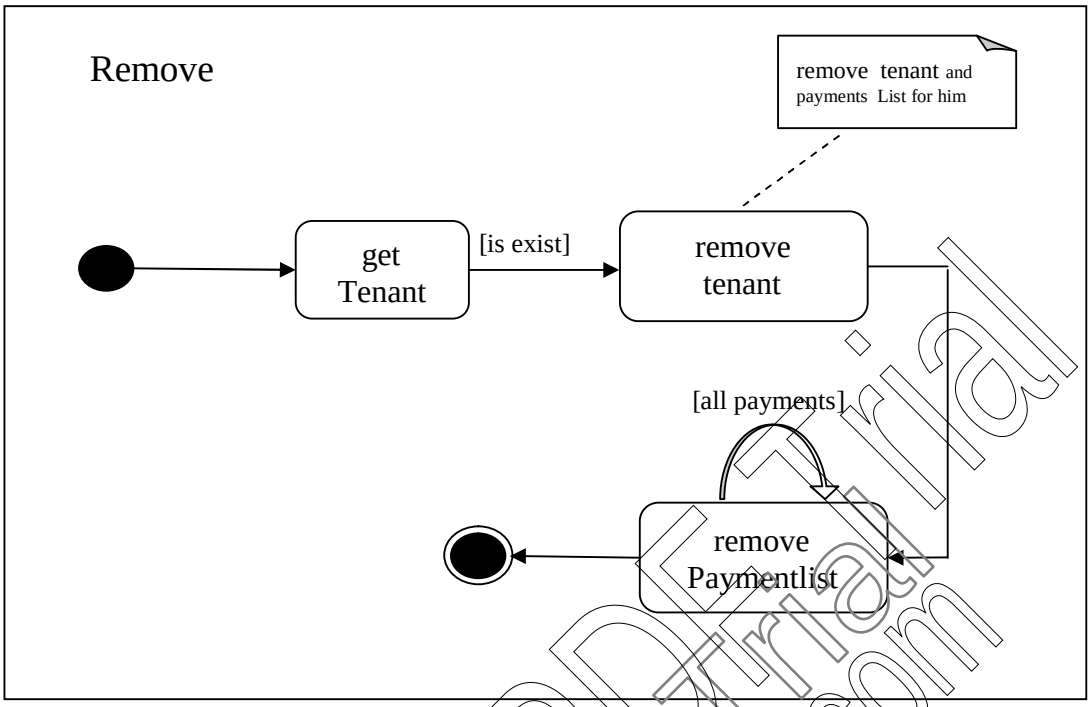
DocuCom PDF Trial
NitroPDF Trial
www.nitropdf.com

State machine diagrams:





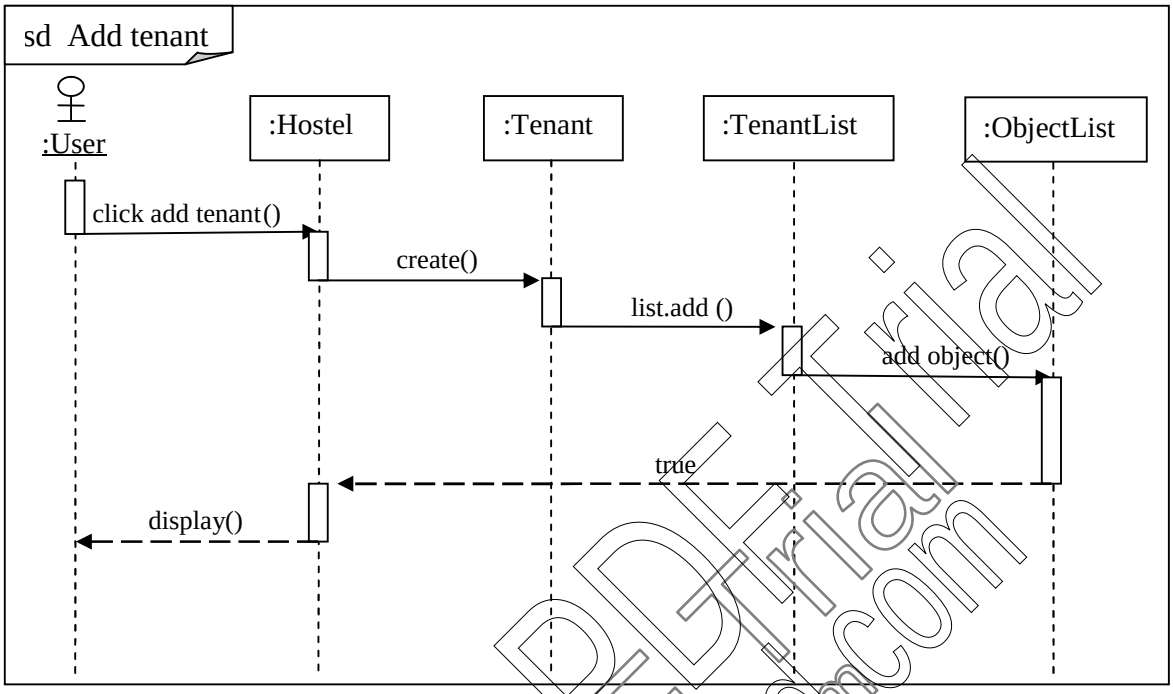
DocuCom PDF Trial
www.nitrodz.com



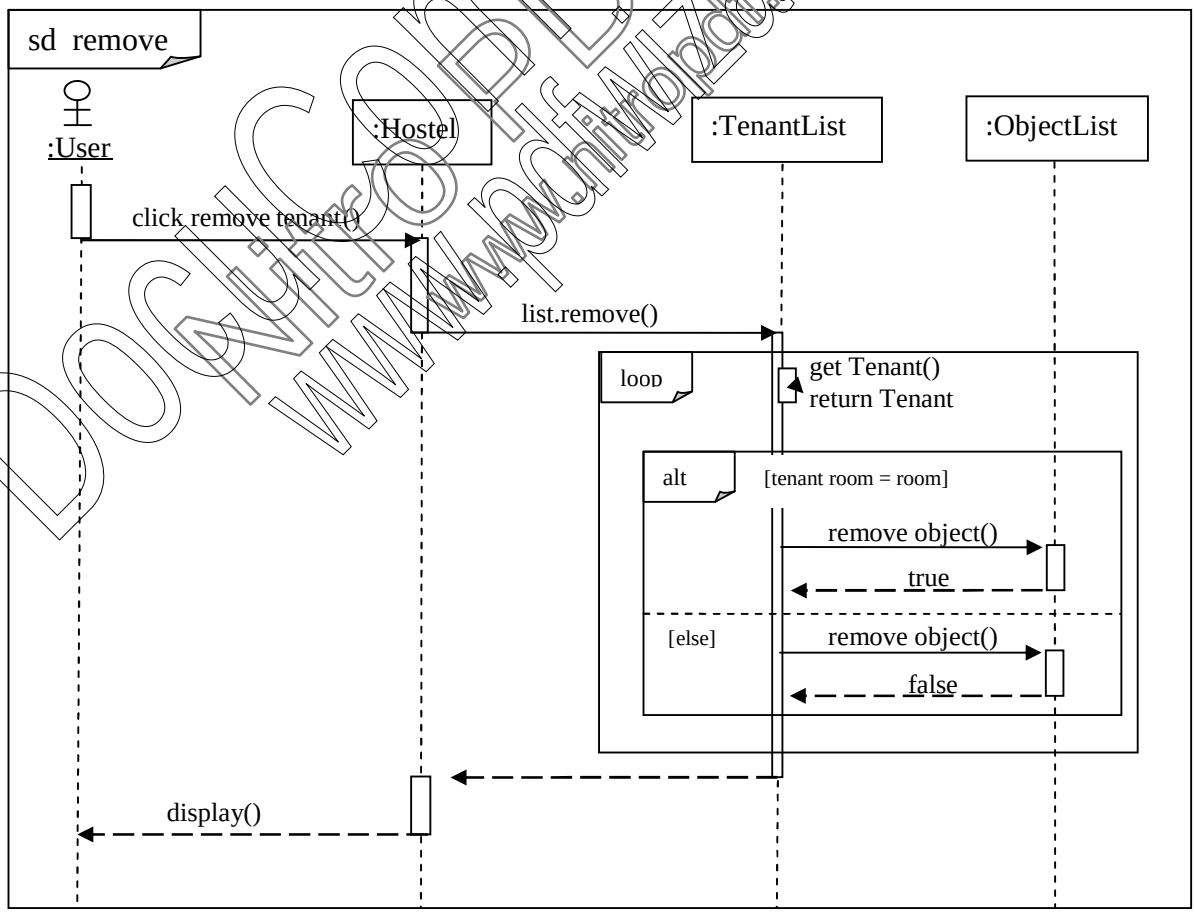
DocuCom PDF Trial
NitroPDF Trial
www.nitropdf.com

Interaction Diagram:

- **sequence diagram of add tenant:**

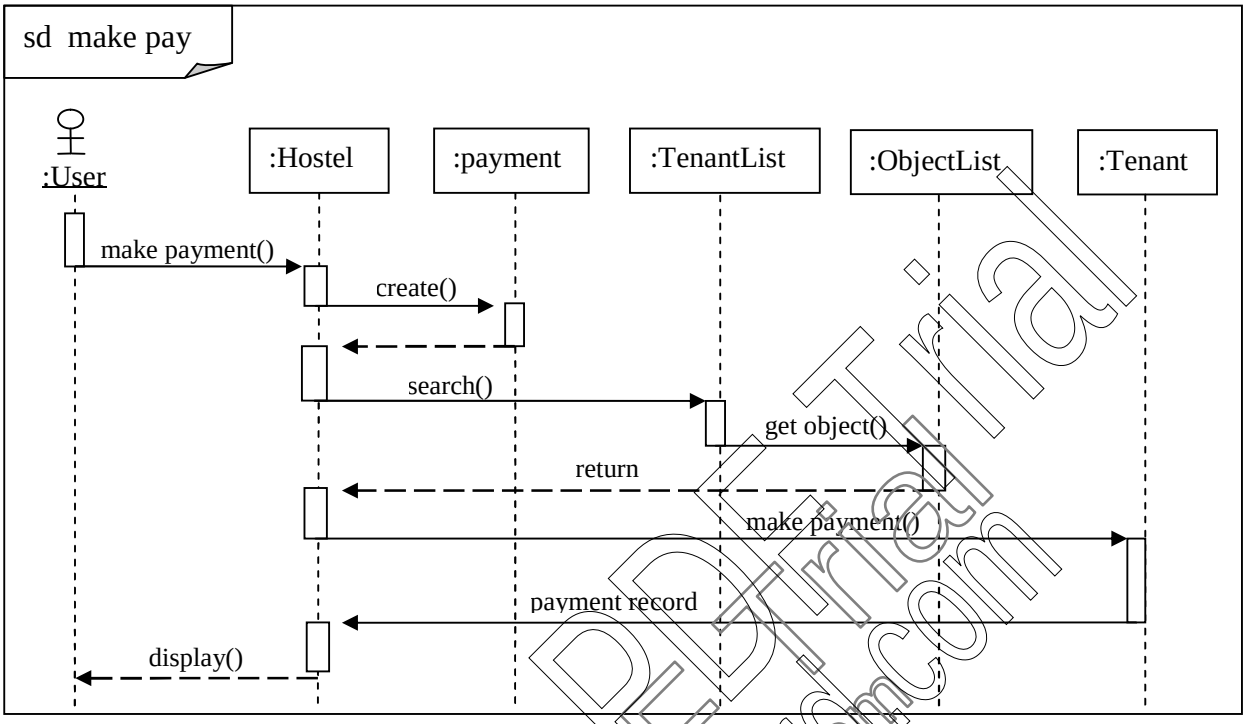


- **sequence diagram of remove tenant:**

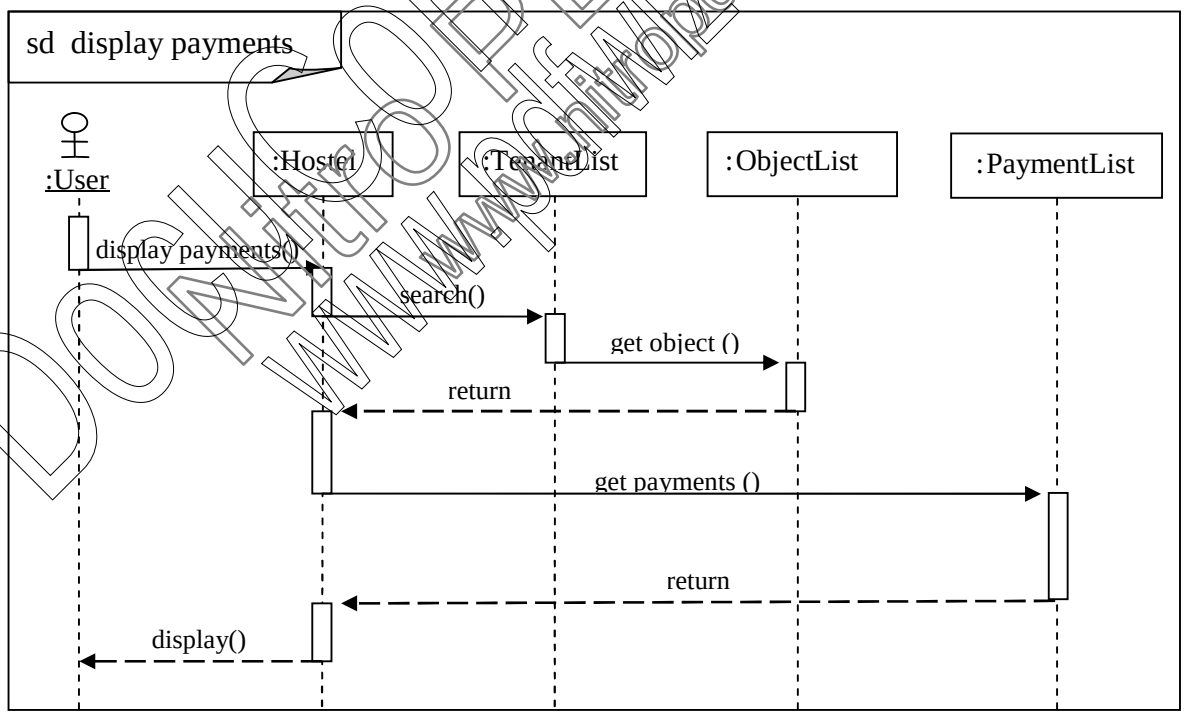


DocuCom PDF Trial
Nitro PDF Trial
www.nitropdf.com

- **sequence diagram of make payment:**

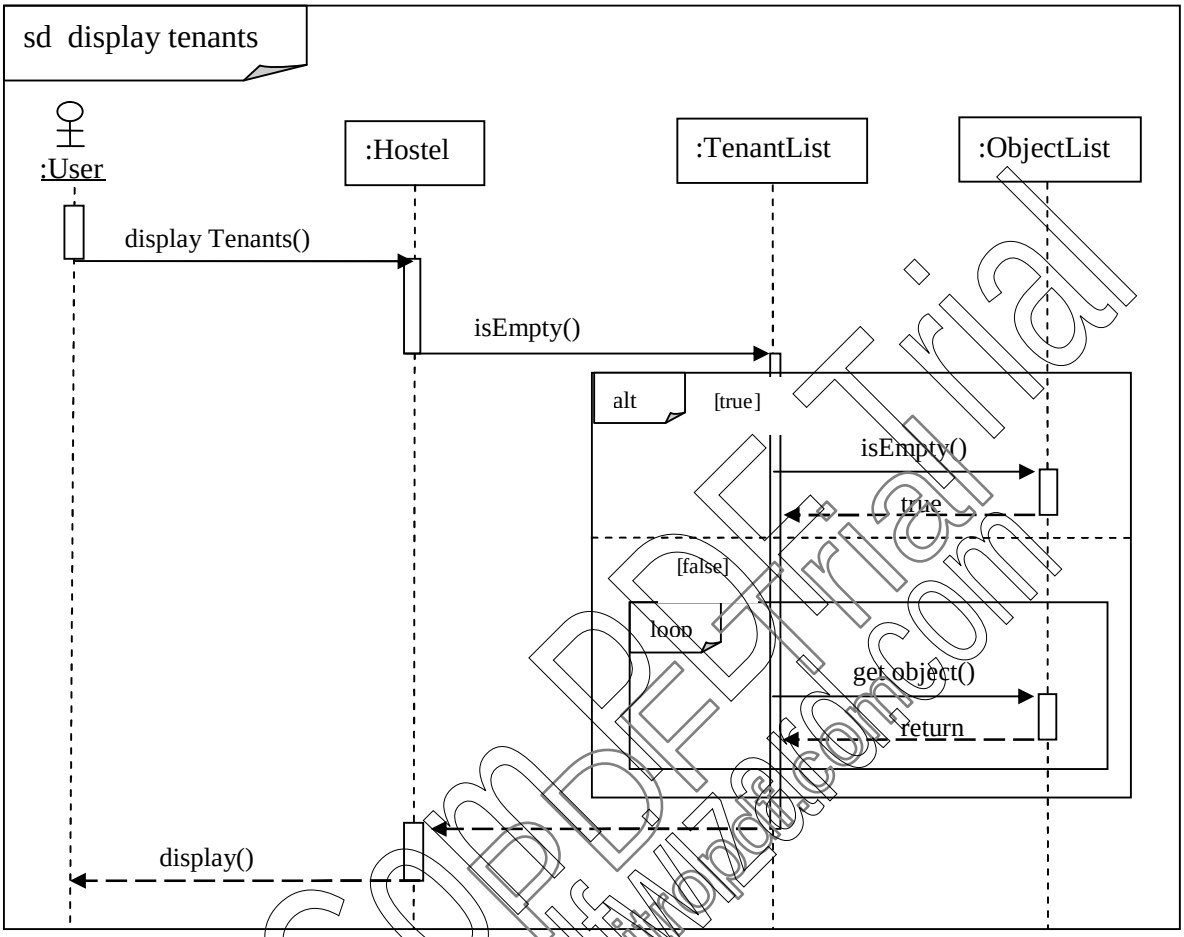


- **sequence diagram of display list of payments:**



DocuPDF Trial
www.ahinozdf.com

- **sequence diagram of display list of tenants:**



DocuCom PDF Trial
NitroPDF Trial
www.nitropdf.com

Appendix :

programming code that use in the analysis and design this project . is following:

```
import java.awt.*;
class RunHostel
{
  public static void main(String[ ] args)
  {
    Frame frame = new Frame();
    frame.setTitle("Student Hostel");
    // assume only 5 rooms available
    Hostel property = new Hostel(5);
    frame.setSize(550,500);
    frame.setBackground(Color.lightGray);
    frame.add(property);
    frame.setVisible(true);
  }
}
```

```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
import java.text.*;
class Hostel extends Applet implements ActionListener
{
 // the attributes
 private int noOfRooms;
 private TenantList list;
 private Button addButton = new Button("Add Tenant");
 private Button displayButton = new Button("Display Tenants");
 private Button removeButton = new Button("Remove Tenant");
 private Label roomLabel = new Label("Room");
 private TextField roomField = new TextField(4);
 private Label nameLabel = new Label("Name");
 private TextField nameField = new TextField(40);
 private Button saveAndQuitButton = new Button("Save and Quit");
 private Button quitButton = new Button("Quit Without Saving");
 private TextArea displayArea = new TextArea(8,75);
 private TextArea displayArea2 = new TextArea(12,75);
 private Label roomLabel2 = new Label("Room");
 private TextField roomField2 = new TextField(4);
 private Label monthLabel = new Label("Month");
 private TextField monthField = new TextField(15);
 private Label amountLabel = new Label("Amount");
 private TextField amountField = new TextField(10);
 private Button paymentButton = new Button("Make Payment");
 private Button listButton = new Button("List Payments");
```

Docuto PDF Trial
www.chinwdf.com

```
// the constructor
public Hostel(int numberIn)
{
  noOfRooms = numberIn;
  list = new TenantList(noOfRooms);
  add(roomLabel);
  roomLabel.setAlignment(Label.RIGHT);
  add(roomField);
  add(nameLabel);
  nameLabel.setAlignment(Label.RIGHT);
  add(nameField);
  add(addButton);
  add(displayButton);
  add(removeButton);
  add(saveAndQuitButton);
  add(quitButton);
  add(displayArea);
  add(roomLabel2);
  roomLabel2.setAlignment(Label.RIGHT);
  add(roomField2);
  add(monthLabel);
  monthLabel.setAlignment(Label.RIGHT);
  add(monthField);
  add(amountLabel);
  amountLabel.setAlignment(Label.RIGHT);
  add(amountField);
  add(paymentButton);
  add(listButton);
  add(displayArea2);
  // add ActionListeners to the buttons
  addButton.addActionListener(this);
  displayButton.addActionListener(this);
  paymentButton.addActionListener(this);
  listButton.addActionListener(this);
  removeButton.addActionListener(this);
  listButton.addActionListener(this);
  saveAndQuitButton.addActionListener(this);
  quitButton.addActionListener(this);
  // read the records file from disk
  TenantFile.readRecords(list);
}
// the event-handlers
public void actionPerformed(ActionEvent e)
{
  if(e.getSource() == addButton)
  {
     String roomEntered = roomField.getText();
     String nameEntered = nameField.getText();
        // if room number or name not entered
  if(roomEntered.length()==0 || nameEntered.length()==0)
```

Docu Converter PDF Trial
www.nitrodf.com

```
    {
      displayArea.setText("Room number and name must be entered");
    }
    // if room number is out of range
    else if(Integer.parseInt(roomEntered)< 1 || Integer.parseInt(roomEntered) >
noOfRooms)
    {
        displayArea.setText("There are only " + noOfRooms + " rooms");
    }
    // if the room is occupied
    else if(list.search(Integer.parseInt(roomEntered))!= null)
    {
 displayArea.setText("Room number " +  Integer.parseInt(roomEntered) + " is
occupied");
    }
    // if everything is okay then add new tenant
    else
    {
        Tenant t = new Tenant(nameEntered,Integer.parseInt(roomEntered));
        list.add(t);
        roomField.setText("");
        nameField.setText("");
        displayArea.setText("New tenant in room "+ roomEntered + " successfully
added");
     }
  }
  if(e.getSource() == displayButton)
  {
    int i;
    // if the list is empty
    if(list.isEmpty())
    {
      displayArea.setText("All rooms are empty");
    }
    // if the list is not empty then display the tenants
    else
    {
      // display a heading
      displayArea.setText("Room" + "\t" + "Name" + "\n");
      // display each tenant in turn
      for(i= 1; i <= list.getTotal(); i++)
      {
  displayArea.append(list.getTenant(i).getRoom() + "\t" + list.getTenant(i).getName() +
"\n");
      }
    }
  }
  if(e.getSource() == removeButton)
  {
    String roomEntered = roomField.getText();
```

DocuCom PDF Trial
NitroPDF Trial
www.nitropdf.com

```
    // if the room number is not entered
    if(roomEntered.length()==0)
    {
      displayArea.setText("Room number must be entered");
    }
    // if the room number is out of range
    else if(Integer.parseInt(roomEntered) < 1 || Integer.parseInt(roomEntered)>
noOfRooms)
    {
      displayArea.setText("Invalid room number");
    }
    // if the room is empty
    else if(list.search(Integer.parseInt(roomEntered)) == null)
    {
      displayArea.setText("Room number " + roomEntered + " is empty");
    }
    // if everything is okay then remove the tenant
    else
    {
      list.removeTenant(Integer.parseInt(roomEntered));
      displayArea.setText("Tenant removed from room " +
Integer.parseInt(roomEntered));
    }
  }
  if(e.getSource() == paymentButton)
  {
    String roomEntered = roomField2.getText();
    String monthEntered = monthField.getText();
    String amountEntered = amountField.getText();

    // if room number, month or amount not entered
    if(roomEntered.length()==0 || monthEntered.length()==0 ||
amountEntered.length()==0)
    {
      displayArea2.setText("Room number, month and amount must all be entered");
    }
    // if room number out of range
    else if(Integer.parseInt(roomEntered) < 1 || Integer.parseInt(roomEntered) >
noOfRooms)
    {
      displayArea2.setText("Invalid room number");
    }
    // if room is empty
    else if(list.search(Integer.parseInt(roomEntered)) == null)
    {
      displayArea2.setText("Room number " + roomEntered + " is empty");
    }
    // if everything is okay then add the new payment
    else
    {
```

Document PDF Trial
Nitro PDF Trial
www.nitropdf.com

```
      Payment p = new
Payment(monthEntered,Double.valueOf(amountEntered).doubleValue());
      list.search(Integer.parseInt(roomEntered)).makePayment(p);
      displayArea2.setText("Payment recorded");
    }
  }
  if(e.getSource() == listButton)
  {
    int i;
    String roomEntered = roomField2.getText();
    // if room number not entered
    if(roomEntered.length()==0)
    {
      displayArea2.setText("Room number must be entered");
    }
    // if room number out of range
    else if(Integer.parseInt(roomEntered) < 1 || Integer.parseInt(roomEntered) >
noOfRooms)
    {
      displayArea2.setText("Invalid room number");
    }
    // if room is empty
    else if(list.search(Integer.parseInt(roomEntered)) == null)
    {
     displayArea2.setText("Room number " + Integer.parseInt(roomEntered) + " is
empty");
    }
    // if everything is okay then list the payments
    else
    {
      Tenant t =
            list.search(Integer.parseInt(roomEntered));
      PaymentList p = t.getPayments();
      if(t.getPayments().getTotal() == 0)
      {
        displayArea2.setText("No payments made for this tenant");
      }
      else
      {
        NumberFormat nf = NumberFormat.getCurrencyInstance();
        String s;
        // display a heading
        displayArea2.setText("Month"  + "\t\t" + "Amount" + "\n");
       // display each payment in turn
        for(i = 1; i <= p.getTotal(); i++)
        {
          s = nf.format(p.getPayment(i).getAmount());
          displayArea2.append("" + p.getPayment(i).getMonth() + "\t\t" + s +    "\n");
        }
```

DocuCom PDF Trial
www.nitrodocom.com

```
        displayArea2.append("\n" + "Total paid so far : " +
nf.format(p.calculateTotalPaid()));
        monthField.setText("");
        amountField.setText("");
      }
    }
  }
  if(e.getSource() == saveAndQuitButton)
  {
    // save the records to file
    TenantFile.saveRecords(noOfRooms,list);
    // terminate the program
    System.exit(0);
  }

  if(e.getSource() == quitButton)
  {
    // terminate the program without saving the records
    System.exit(0);
  }
 }
}
```

```
class ObjectList
{
 private Object[] object ;
 private int total ;
 public ObjectList(int sizeIn)
 {
    object = new Object[sizeIn];
    total = 0;
 }
 public boolean add(Object objectIn)
 {
    if(!isFull())
    {
       object[total] = objectIn;
       total++;
       return true;
    }
    else
    {
       return false;
    }
 }
 public boolean isEmpty()
 {
    if(total==0)
    {
       return true;
```

Doc1Com PDF Trial
Nitro PDF Trial
www.nitrodf.com

```
    }
    else
    {
        return false;
    }
 }
 public boolean isFull()
 {
    if(total==object.length)
    {
        return true;
    }
    else
    {
        return false;
    }
 }
 public Object getObject(int i)
 {
    return object[i-1];
 }
 public int getTotal()
 {
    return total;
 }
 public boolean remove(int numberIn)
 {
    // check if index is valid before removing
    if(numberIn >= 1 && numberIn <= total)
    {   // overwrite object by shifting following objects along
        for(int i = numberIn-1; i <= total-2; i++)
        {
            object[i] = object[i+1];
        }
        total--; // Decrement total number of objects
        return true;
    }
    else // remove was unsuccessful
    {
        return false;
    }
 }
}
```

```
class PaymentList extends ObjectList // inherit from ObjectList
{
 public PaymentList(int sizeIn)
 {   // call ObjectList constructor
    super(sizeIn);
 }
```

Doc to PDF Trial
Nitro PDF Trial
www.pdfwizard.com

```
 public Payment getPayment(int indexIn)
 {   // call inherited method and type cast
    return (Payment) getObject(indexIn);
 }
 public double calculateTotalPaid()
 {
    double totalPaid = 0; // initialize totalPaid
    // loop through all payments
    for (int i=1; i<= getTotal();i++)
    {   // add current payment to running total
       totalPaid = totalPaid + getPayment(i).getAmount();
    }
    return totalPaid;
 }
}
```

```
class TenantList extends ObjectList
{
 public TenantList(int sizeIn)
 {   // call ObjectList constructor
    super(sizeIn);
 }
 public Tenant getTenant(int indexIn)
 {   // call inherited method and type cast
    return (Tenant)getObject(indexIn);
 }
 public Tenant search(int roomIn)
 {
    for(int i=1;i<=getTotal();i++)
    {     // find tenant with given room number
                if(getTenant(i).getRoom() == roomIn)
                {
                    return getTenant(i);
                }
    }
    return null; // no tenant found with given room number
 }
 public boolean removeTenant(int roomIn)
 {
    for(int i=1;i<=getTotal();i++)
    {    // remove tenant with given room number
                if(getTenant(i).getRoom() == roomIn)
                {
             remove(i);
             return true;
                }
    }
    return false; // no tenant found with given room number
 }
}
```

Doc Nitro PDF Trial
www.nitrodf.com

```
---------------------------------------------------------------------------------------------------
class Tenant
{
  private String name;
  private int room;
  private PaymentList payments;
  private static final int maxNoOfPayments = 12;
  Tenant(String nameIn, int roomIn)
  {
    name = new String(nameIn);
    room = roomIn;
    payments = new PaymentList(maxNoOfPayments);
  }
  public String getName()
  {
    return name;
  }
  public int getRoom()
  {
    return room;
  }
  public void makePayment(Payment paymentIn)
  { // call PaymentList method
    payments.add(paymentIn);
  }
  public PaymentList getPayments()
  {
    return payments;
  }
}
---------------------------------------------------------------------------------------------------
class Payment
{
 private String month;
 private double amount;

 public Payment(String monthIn, double amountIn)
 {
    month = monthIn;
    amount=amountIn;
 }
 public String getMonth()
 {
    return month;
 }
 public double getAmount()
 {
    return amount;
 }
}
```

DocuCom PDF Trial
www.nitroPDFWizard.com

```
import java.io.*;
class TenantFile
{
  public static void saveRecords(int noOfRoomsIn, TenantList listIn)
  {
    try
    {
      FileOutputStream ostream = new FileOutputStream("Tenants.txt");
      PrintWriter pw = new PrintWriter(ostream);
      pw.println(listIn.getTotal());
      for(int i=1; i <= noOfRoomsIn; i++)
      {
        if(listIn.getTenant(i) != null)
        {
          pw.println(listIn.getTenant(i).getRoom());
          pw.println("\"" + listIn.getTenant(i).getName() + "\"");
          pw.println(listIn.getTenant(i).getPayments().getTotal());
          for(int j = 1;
               j<= listIn.getTenant(i).getPayments().getTotal(); j++)
          {
            pw.println("\"" + listIn.getTenant(i).getPayments().getPayment(j).getMonth()
            + "\"");
            pw.println(listIn.getTenant(i).getPayments().getPayment(j).getAmount());
          }
       }
      }
      pw.flush();
      ostream.close();
    }
    catch(IOException ioe)
    {
      System.out.println("Error writing file");
    }
  }
  public static void readRecords(TenantList listIn)
  {
    try
    {
      FileInputStream istream = new FileInputStream("Tenants.txt");
      InputStreamReader ir = new InputStreamReader(istream);
      BufferedReader br = new BufferedReader(ir);
      StreamTokenizer st = new StreamTokenizer(br);
      int tempRoom = 0;
      String tempName = new String("");
      String tempMonth = new String("");
      double tempAmount = 0 ;
      Tenant tempTenant = null;
      Payment tempPayment =  null;
      int tot = 0;
      int totpay = 0;
```

DocuCom PDF Trial
Nitro PDF Trial
www.nitroPDF.com

```
        st.nextToken();
        tot = (int) st.nval;
        for(int j = 1; j<=tot; j++)
        {
           st.nextToken();
           tempRoom = (int) st.nval;
           st.nextToken();
           tempName = st.sval;
           tempTenant = new Tenant(tempName,tempRoom);
           st.nextToken();
           totpay = (int) st.nval;
           for(int k = 1; k <= totpay; k++)
           {
               st.nextToken();
               tempMonth = st.sval;
               st.nextToken();
               tempAmount = st.nval;
               tempPayment = new Payment(tempMonth, tempAmount);
               tempTenant.makePayment(tempPayment);
            }
           listIn.add(tempTenant);
         }
      }
      catch(IOException ioe)
      {
        System.out.println("No records found");
      }
    }
  }
```

DocuCom PDF Trial
NitroPDF Trial
www.pdfwizard.com