

# RELAZIONE DI INFORMATICA

**“Esercitazione CondominiumManagement SWING”**

# Indice

<b>Parte Teorica</b>	<b>3</b>
Sommaro	3
<b>Parte Tecnica</b>	<b>4</b>
UML	4
Package data	5
-Classe Apartament	6
-Classe Condominium	6
-Classe CondominiumManagement	6
Package frames	7
-Classe MainMenu	7
-Classe CreateCond	7
-Classe CondMenu	8
-Classe ModApartment	8
Schermate di esecuzione	9
Schermate Semplificate	9
Schermate Effettive	14
Collegamenti utili	19

# Parte Teorica

## Sommario

Il programma Java ha richiesto molto impegno e molta ricerca su Internet, ed infine sono rimasto piuttosto soddisfatto del risultato. Inizialmente era più faticosa la scrittura, ma con il tempo ho chiaramente imparato certe sintassi/meccaniche e quindi il tempo passato a ricercare era sempre minore.

Cose che ho ricercato:

- Molte classi Swing
- Molti listener, tra cui interfacce e adapter
- Organizzazione della business class e del package dei dati

Il punto che ho sempre tenuto a mente è uno: "l'internazionalità" del programma.

Ho cercato di rendere la rielaborazione del programma più facile possibile.

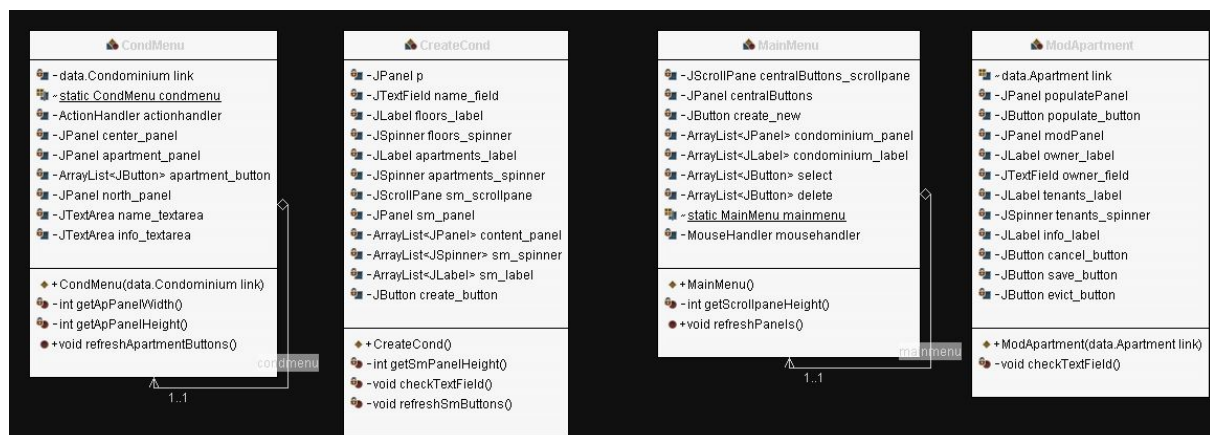
Per fare ciò ho fatto un sistema che consiste nel creare un package separato con tutti i dati del programma (package **data**), con la classe **DataAccess** che gestisce tutto il flusso input/output dei dati. Facendo così, spero di aver facilitato il lavoro agli sviluppatori dopo di me che sapranno di dover semplicemente richiamare staticamente **DataAccess** per accedere ai dati interni.

# Parte Tecnica

## UML

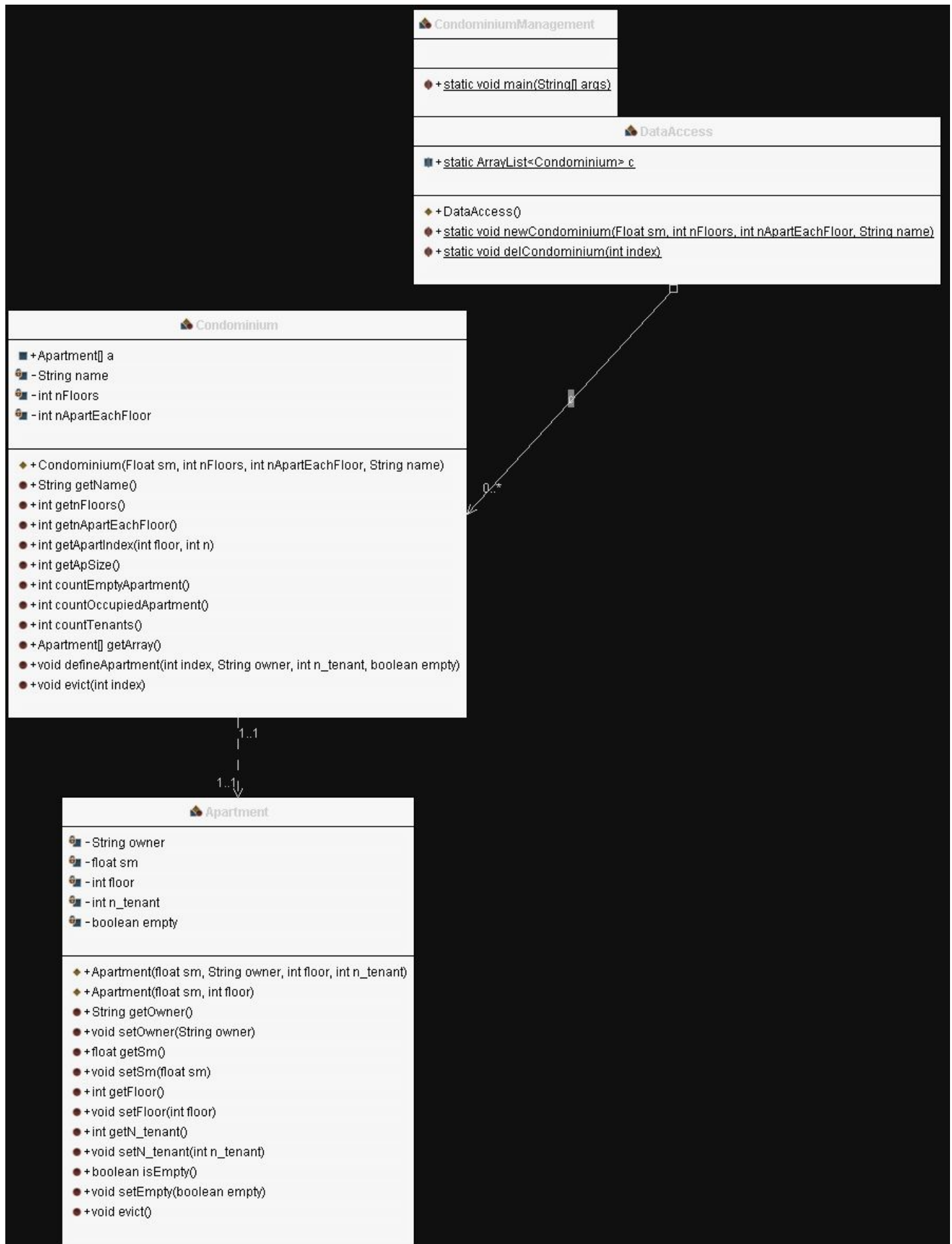


## Package data



## Package frames

## Package data



Apartment
<ul style="list-style-type: none"> <li>- String owner</li> <li>- float sm</li> <li>- int floor</li> <li>- int n_tenant</li> <li>- boolean empty</li> </ul>
<ul style="list-style-type: none"> <li>+ Apartment(float sm, String owner, int floor, int n_tenant)</li> <li>+ Apartment(float sm, int floor)</li> <li>+ String getOwner()</li> <li>+ void setOwner(String owner)</li> <li>+ float getSm()</li> <li>+ void setSm(float sm)</li> <li>+ int getFloor()</li> <li>+ void setFloor(int floor)</li> <li>+ int getN_tenant()</li> <li>+ void setN_tenant(int n_tenant)</li> <li>+ boolean isEmpty()</li> <li>+ void setEmpty(boolean empty)</li> <li>+ void evict()</li> </ul>

## -Classe Apartment

Contiene tutti i dati di un singolo appartamento come il nome del proprietario, i metri quadri, il piano, il numero di coinquilini, ed il boolean empty per riconoscere se è occupato o vuoto.

Condominium
<ul style="list-style-type: none"> <li>+ Apartment[] a</li> <li>- String name</li> <li>- int nFloors</li> <li>- int nApartmentEachFloor</li> </ul>
<ul style="list-style-type: none"> <li>+ Condominium(float sm, int nFloors, int nApartmentEachFloor, String name)</li> <li>+ String getName()</li> <li>+ int getnFloors()</li> <li>+ int getnApartmentEachFloor()</li> <li>+ int getApartmentIndex(int floor, int n)</li> <li>+ int getApSize()</li> <li>+ int countEmptyApartment()</li> <li>+ int countOccupiedApartment()</li> <li>+ int countTenants()</li> <li>+ Apartment[] getArray()</li> <li>+ void defineApartment(int index, String owner, int n_tenant, boolean empty)</li> </ul>

## -Classe Condominium

Contiene tutti i dati di un condominio come il nome dell'edificio, il numero di piani, il numero di piano in un piano. Infine contiene un array di **Apartment** che rappresenta gli appartamenti nell'edificio. E' previsto di metodi per agire sugli appartamenti che invocheranno a loro volta i loro metodi per eseguire la stessa medesima cosa

.DataAccess
<ul style="list-style-type: none"> <li>+ static ArrayList&lt;Condominium&gt; c</li> </ul>
<ul style="list-style-type: none"> <li>+ DataAccess()</li> <li>+ static void newCondominium(float sm, int nFloors, int nApartmentEachFloor, String name)</li> <li>+ static void delCondominium(int index)</li> </ul>

## -Classe DataAccess

E' la classe che permette a tutti di interfacciarsi con i dati interni del programma. Contiene l'ArrayList dei condomini(e tutto quello che contengono).

Per accedere ai dati basta richiamare

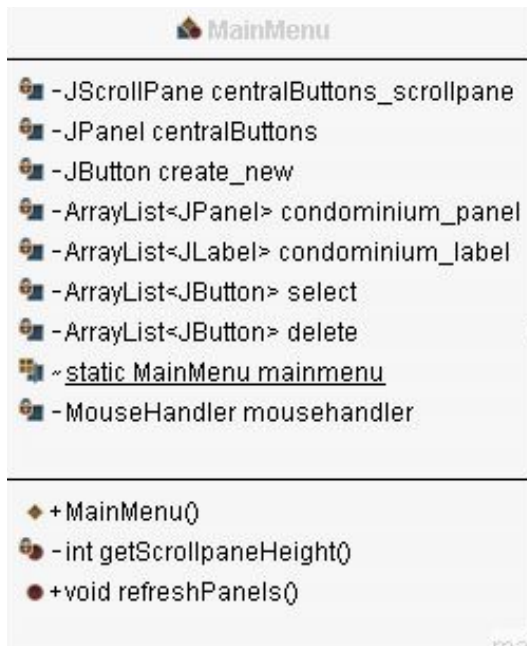
**DataAccess.c** proprio perché il modificatore di accesso è impostato su static.

CondominiumManagement
<ul style="list-style-type: none"> <li>+ static void main(String[] args)</li> </ul>

## -Classe CondominiumManagement

Contiene il main, che prima istanzia il **DataAccess**, poi il **mainMenu**

## Package frames



### -Classe **MainMenu**

La classe grafica che viene istanziata per prima, è il menù principale. Permette di creare ed eliminare condomini illimitati e di poter scegliere quale gestire



### -Classe **CreateCond**

Una volta premuto sul bottone “+” viene aperta questa finestra che permette di creare un nuovo condominio.

I dati inseriti sono il nome del condominio, il numero di piani, il numero di appartamenti per ogni piano.. Inoltre nella mini finestrella appaiono quanti appartamenti quanti sono quelli impostati sul JNumberModel spinner. Il controllo sulla correttezza dei dati inseriti è continuo, ed in caso di errore non valido, i tasti importanti per verranno disabilitati, impedendo all’utente di avanzare.

CondMenu
<ul style="list-style-type: none"> <li>- data.Condominium link</li> <li>~static CondMenu condmenu</li> <li>-ActionHandler actionhandler</li> <li>-JPanel center_panel</li> <li>-JPanel apartment_panel</li> <li>-ArrayList&lt;JButton&gt; apartment_button</li> <li>-JPanel north_panel</li> <li>-JTextArea name_textarea</li> <li>-JTextArea info_textarea</li> </ul>
<ul style="list-style-type: none"> <li>+ CondMenu(data.Condominium link)</li> <li>-int getApPanelWidth()</li> <li>-int getApPanelHeight()</li> <li>+void refreshApartmentButtons()</li> </ul>

## -Classe CondMenu

Dopo avere selezionato il Condominio da modificare, questa finestra viene istanziata per permettere la modifica dei singoli appartamenti nel condominio selezionato prima. al centro presenta il pannello che contiene tutti i JButton indicanti gli appartamenti. nella zona nord è presente un pannello contenente un JLabel e una JTextArea. il JLabel indica il nome del condominio mentre la JTextArea mostra vari dati. Le dimensioni del pannello centrale saranno impostate in base al numero di appartamenti così che la disposizione dei bottoni vada a capo tutte le volte che inizia la raffigurazione di un nuovo piano. I bottoni Una volta premuti portano alla finestra di modifica dell'appartamento.

ModApartment
<ul style="list-style-type: none"> <li>~ data.Apartment link</li> <li>-JPanel populatePanel</li> <li>-JButton populate_button</li> <li>-JPanel modPanel</li> <li>-JLabel owner_label</li> <li>-JTextField owner_field</li> <li>-JLabel tenants_label</li> <li>-JSpinner tenants_spinner</li> <li>-JLabel info_label</li> <li>-JButton cancel_button</li> <li>-JButton save_button</li> <li>-JButton evict_button</li> </ul>
<ul style="list-style-type: none"> <li>+ ModApartment(data.Apartment link)</li> <li>-void checkTextField()</li> </ul>

## -Classe ModApartment

Dopo aver premuto il bottone di un appartamento questa finestra viene istanziata e si apre. vengono creati un JTextField, un JSpinner per modificare/ visualizzare il numero di coinquilini, un JLabel per visualizzare i dati, un tasto per salvare, un tasto per annullare, ed un tasto per sfrattare l'appartamento. Viene inoltre istanziato un bottone che verrà visualizzato da solo solo in caso in cui l'appartamento sia vuoto: una volta cliccato farà sì che l'appartamento sia affittato. Vengono fatti controlli continui sulla validità dei dati inseriti, nel caso almeno uno non sia corretto, il tasto salva viene disabilitato fino ad avvenuta correzione dell'utente. se affittiamo un appartamento che prima era vuoto possiamo vedere come lo slot dell'appartamento nella finestra precedente cambia colore da verde al rosso, indicando che l'appartamento è stato occupato.

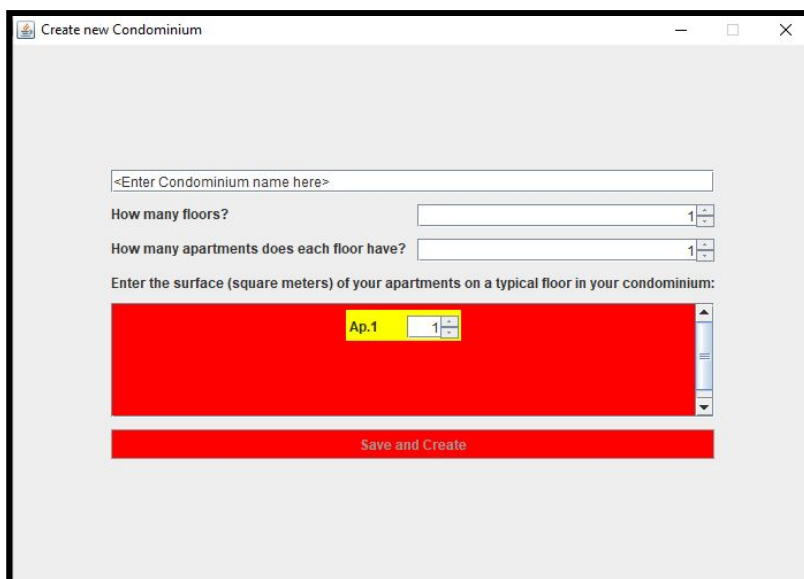


## Schermate di esecuzione

Le schermate si dividono in: **Schermate semplificate** e **Schermate effettive**  
Gli screenshot si susseguono per ordine di esecuzione

### Schermate Semplificate

Utili per capire la dimensione dei vari Panel e l'organizzazione in generale

A screenshot of a web application window titled "Create new Condominium". The window has a light gray background and a black border. At the top, there is a text input field with the placeholder text "<Enter Condominium name here>". Below this, there are two rows of labels and numeric inputs: "How many floors?" with a value of "1", and "How many apartments does each floor have?" with a value of "1". Both numeric inputs have small up and down arrows. Below these, there is a label "Enter the surface (square meters) of your apartments on a typical floor in your condominium:". Underneath this label is a table with a red background. The table has one visible row labeled "Ap.1" in a yellow cell, followed by a numeric input with the value "1". To the right of the table is a vertical scrollbar. At the bottom of the form, there is a red button with the text "Save and Create" in white.

Create new Condominium

Invalid Name

How many floors?1

How many apartments does each floor have?6

Enter the surface (square meters) of your apartments on a typical floor in your condominium:

Ap.11Ap.21Ap.31Ap.41

Ap.51Ap.61

Save and Create

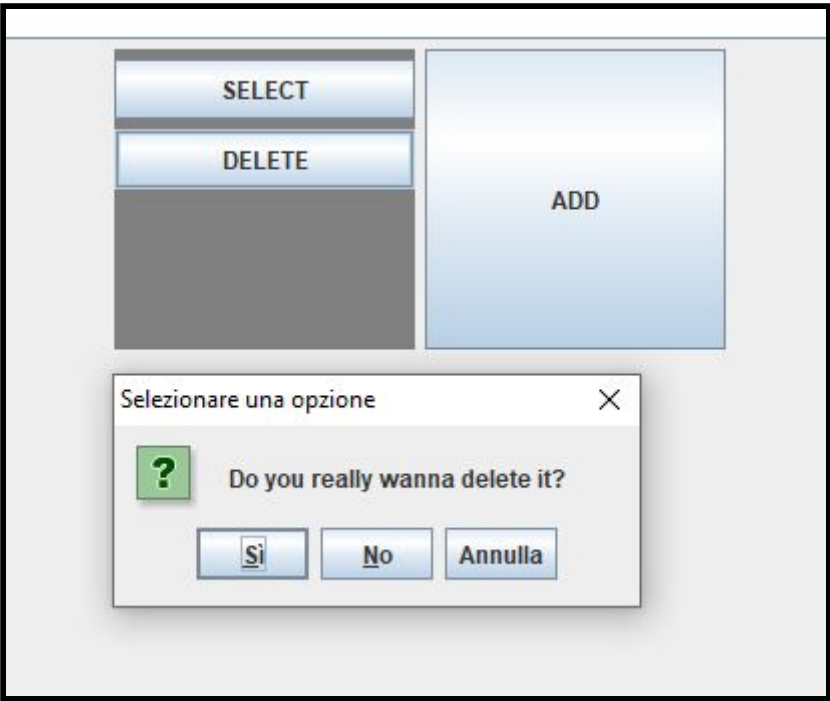
'asdsda'

ADD

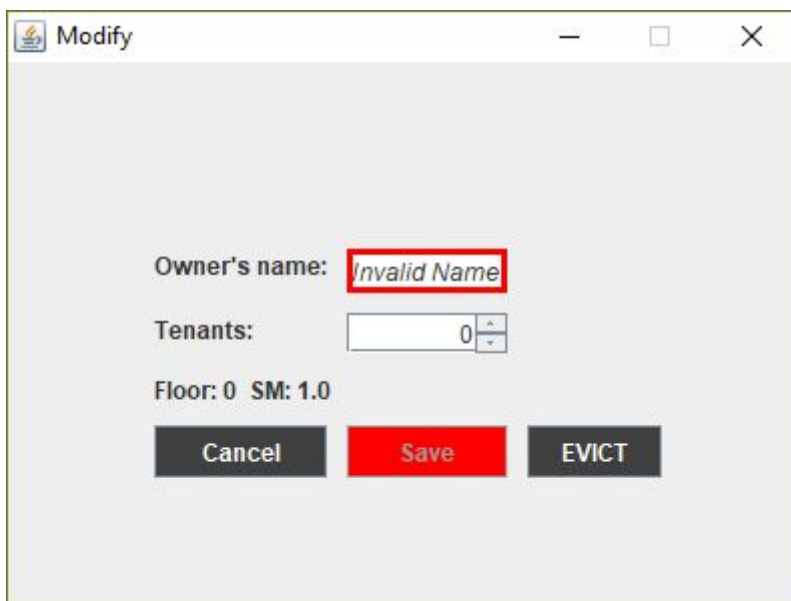
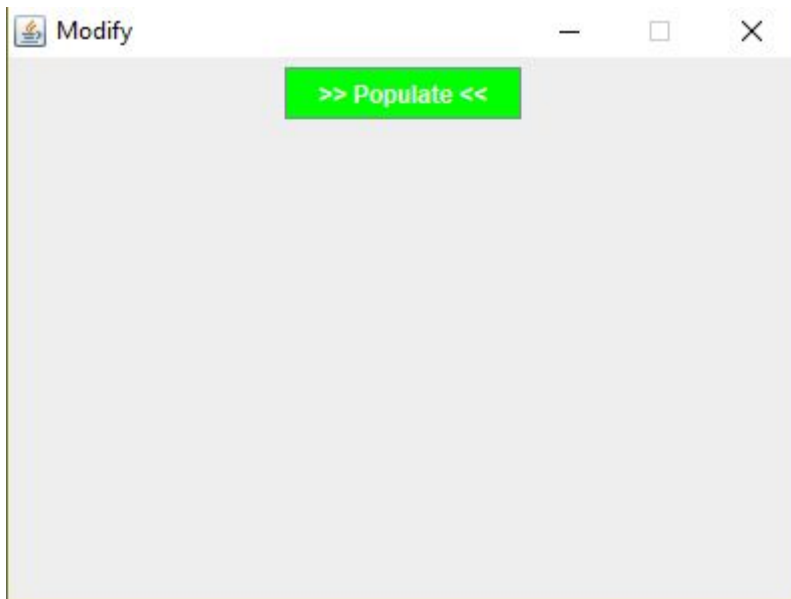
SELECT


DELETE

ADD



' Esempio 1 '										Empty Apartments: 100/100
										Occupied apartments: 0/100
										Number of tenants: 0
1	2	3	4	5	6	7	8	9	10	
11	12	13	14	15	16	17	18	19	20	
21	22	23	24	25	26	27	28	29	30	
31	32	33	34	35	36	37	38	39	40	
41	42	43	44	45	46	47	48	49	50	
51	52	53	54	55	56	57	58	59	60	
61	62	63	64	65	66	67	68	69	70	
71	72	73	74	75	76	77	78	79	80	
81	82	83	84	85	86	87	88	89	90	
91	92	93	94	95	96	97	98	99	100	



 Modify

Owner's name:

Tenants:

Floor: 0 SM: 1.0

Cancel

Save

EVICT

'Esempio 1'

Empty Apartments: 100/100

Occupied apartments: 0/100

Number of tenants: 0

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

## Schermate Effettive



A screenshot of a software window titled "Create new Condominium". The window has a dark blue background and contains the following form elements:

- A text input field with the placeholder text "<Enter Condominium name here>".
- A label "How many floors?" followed by a numeric input field containing the value "1".
- A label "How many apartments does each floor have?" followed by a numeric input field containing the value "1".
- A label "Enter the surface (square meters) of your apartments on a typical floor in your condominium:" followed by a list box.
- The list box contains one item labeled "Ap.1" with a numeric input field next to it containing the value "1".
- A red button at the bottom labeled "Save and Create".

The window's title bar includes standard minimize, maximize, and close buttons.

Create new Condominium

Invalid Name

How many floors?

6

How many apartments does each floor have?

12

Enter the surface (square meters) of your apartments on a typical floor in your condominium:

Ap.1	<div>1</div>	Ap.2	<div>1</div>	Ap.3	<div>1</div>	Ap.4	<div>1</div>
Ap.5	<div>1</div>	Ap.6	<div>1</div>	Ap.7	<div>1</div>	Ap.8	<div>1</div>
Ap.9	<div>1</div>	Ap.10	<div>1</div>	Ap.11	<div>1</div>	Ap.12	<div>1</div>

Save and Create

Create new Condominium

Esempio

How many floors?

6

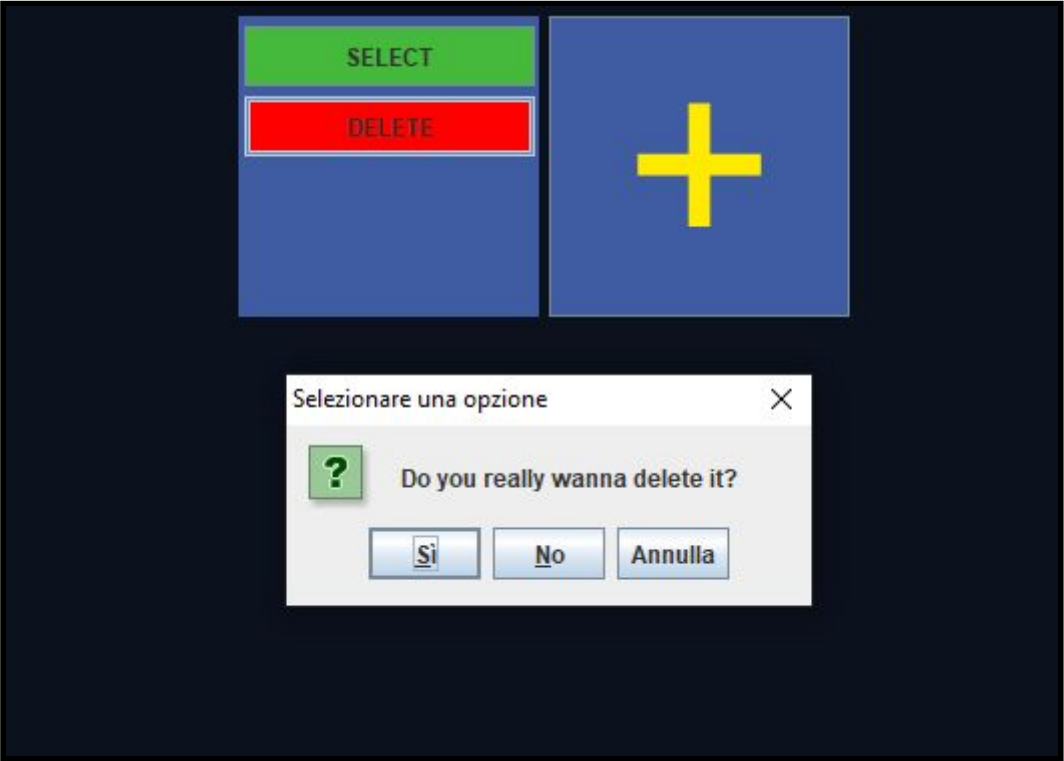
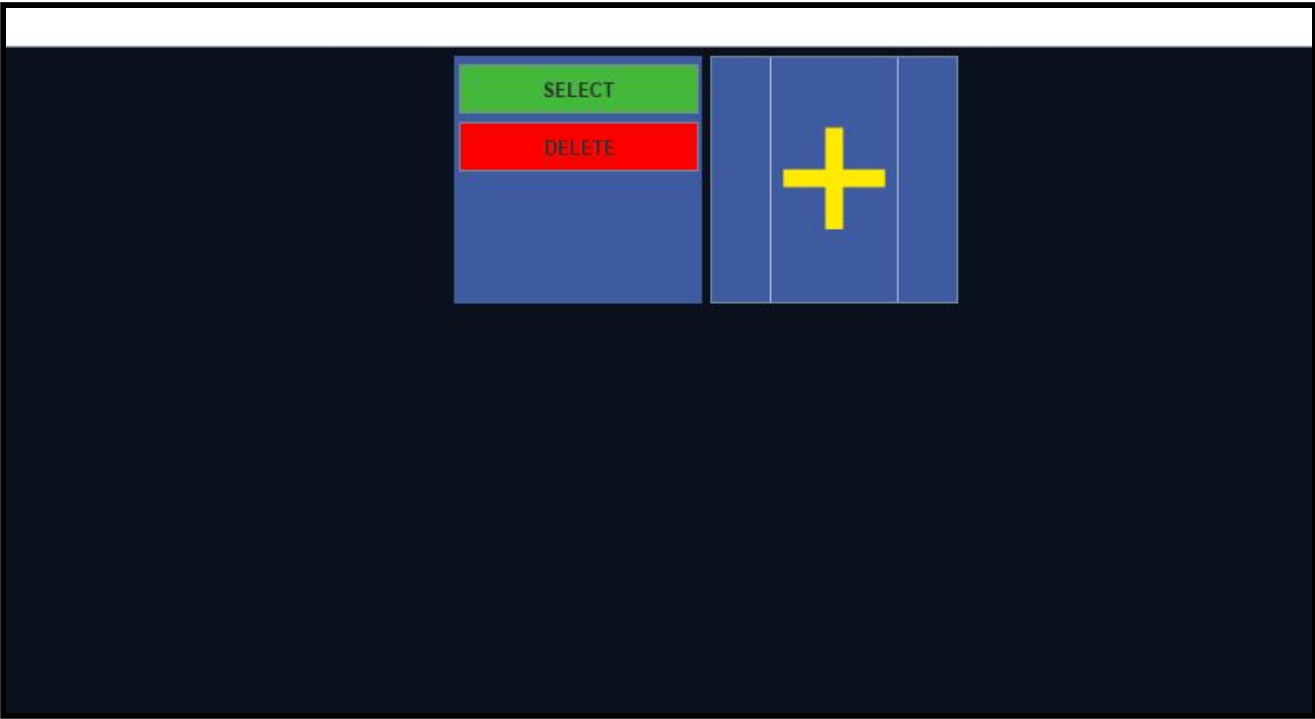
How many apartments does each floor have?

12

Enter the surface (square meters) of your apartments on a typical floor in your condominium:

Ap.1	<div>1</div>	Ap.2	<div>1</div>	Ap.3	<div>1</div>	Ap.4	<div>1</div>
Ap.5	<div>1</div>	Ap.6	<div>1</div>	Ap.7	<div>1</div>	Ap.8	<div>1</div>
Ap.9	<div>1</div>	Ap.10	<div>1</div>	Ap.11	<div>1</div>	Ap.12	<div>1</div>

Save and Create





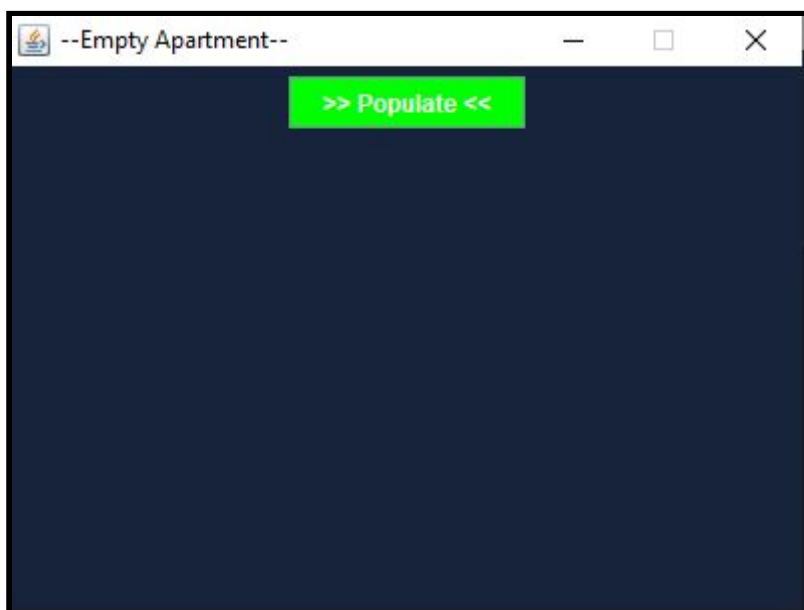
# 'Esempio'

Empty Apartments: 72/72

Occupied apartments: 0/72

Number of tenants: 0

1	2	3	4	5	6	7	8	9	10	11	12
13	14	15	16	17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32	33	34	35	36
37	38	39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70	71	72



--Empty Apartment--

Owner's name: Invalid Name

Tenants:

Floor: 0 SM: 1.0

--Empty Apartment--

Owner's name:

Tenants:

Floor: 0 SM: 1.0

Mario Rossi's Apartment (F:0 SM:1.0)

Owner's name:

Tenants:

Floor: 0 SM: 1.0

# ' Esempio '

Empty Apartments: 65/72

Occupied apartments: 7/72

Number of tenants: 17

1	2	3	4	5	6	7	8	9	10	11	12
13	14	15	16	17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32	33	34	35	36
37	38	39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70	71	72

## Collegamenti utili

→ [Collor.com](https://www.collor.com)

**INDICE**