Usuwanie zaznaczonego elementu z jList po naciśnięciu przycisku:

```
private void jBRemoveTaskActionPerformed(java.awt.event.ActionEvent evt) {
    System.out.println("Removed task:");
    int index = jLTask.getSelectedIndex();
    String title = dlm.get(index).toString();
    System.out.println("Index: "+index+"\ttitle: "+title);
    dlm.remove(index);
}
```

dlm – jest to: **DefaultListModel**, który przechowuje elementy do wyświetlania w jList. W powyższym przypadku jest to obiekt **String**.

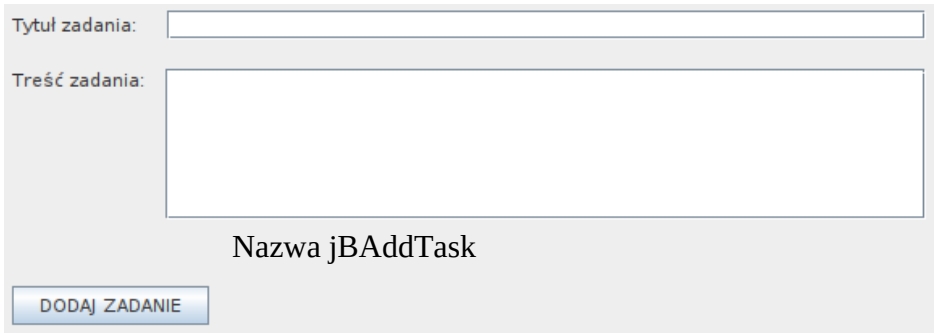
Usunięcie ostatniego elementu (ostatnio dodanego):

```
dlm.removeElementAt(dlm.size()-1);
```

Różnica między: **remove** a **removeElementAt** – pierwszy zwraca usunięty obiekt (Object), a więc można przypisać do zmiennej np.:

```
String temp = (String) dlm.remove(index);
```

Dodawanie elementu do jList:



Nazwa jTFTaskTitle

Nazwa jTATaksDescription

Nazwa jBAddTask

Po naciśnięciu dodaj:

```
private void jBAddTaskActionPerformed(java.awt.event.ActionEvent evt) {
    String title = jTFTaskTitle.getText();
    dlm.addElement(title);
}
```

Jak zapewne zauważyłeś nie użyliśmy widoku: jTATaksDescription.
Pojawi się ono później, jak do stworzymy obiekt w którym będziemy przechowywali odpowiednie informacje – zadania.

Przerobimy teraz nasze zadania tak aby można było zapisać dane o naszym zadaniu takie jak: tytuł i opis.

Patrz następna strona!

```
/**
 * Object to store title and description of task
 * @author adams
 */
public class Task {
    private String title;
    private String description;

    public Task(String title, String description) {
        this.title = title;
        this.description = description;
    }

    public String getTitle() {
        return title;
    }

    public String getDescription() {
        return description;
    }
}
```

Teraz aby dodać nowy **Task** do **DefaultListModel** **nie wolno** zrobić jak poniżej. Sprawdź co się stanie jak wywołamy poniższy kod programu:

```
private void jBAddTaskActionPerformed(java.awt.event.ActionEvent evt) {
    String title = jTFTaskTitle.getText();
    String description = jTATaksDescription.getText();
    dlm.addElement(new Task(title,description));
}
```

Do **DefaultListModel** dodajemy tylko String z tytułem naszego zadania, ale zadanie należy też gdzieś przechować. Do obiektu **Task** należy dodać metodę **toString()**,

```
@Override
public String toString() {
```

która zwróci tylko **title** z naszego obiektu! Wtedy powyższy kod z dodawaniem **Task** do **DefaultListModel** zadziała!!!

Teraz lekko zmodyfikuj usuwanie elementów z **DefaultListModel** tak aby informacje wyświetlały się wszystkie z naszego obiektu jak poniżej:

```
Removed task:
Index: 0        title: Zad 1
Bardzo ciężkie zadania
```

Aby powyższe zadziałało musimy zmodyfikować sposób przechowywania danych, już nie w **DefaultListModel** bo tam są tylko dane typu String. Użyjemy poniższej listy:

```
ArrayList<Task> listOfTasks;
```

Elementy w tej liście odpowiadają elementom w **DefaultListModel** więc jak usuwamy elementy z JList to należy usunąć je z listOfTask i **DefaultListModel**.

Tak samo jak dodajemy elementy do JList należy dodać cały obiekt Task do listOfTask oraz do **DefaultListModel**

Tak naprawdę wystarczy dodać do **DefaultListModel** task.getTitle() czyli użyć:

```
Task t = new Task("Temat","Opis zadania");
dlm.addElement(t.getTitle());
```

Wyjaśnienie:
Tworzymy obiekt Task **t**, następnie dodajemy do **DefaultListModel (dlm)** tylko tytuł zadania!