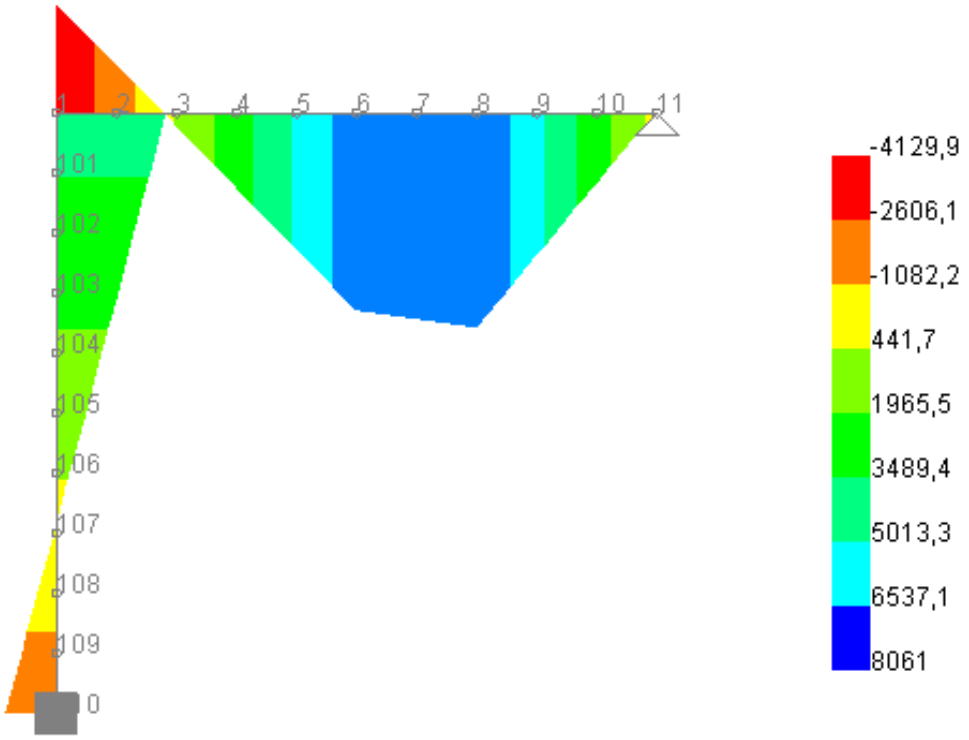
JSTRUCT
User's Guide



Miguel Fernández Ruiz
phD. Civil Engineer

Madrid, 2004

Contents

---

1 Introduction

`JSTRUCT` is a bidimensional program that allows to define, solve and postprocess framed structures. One analysis may contain several input models as well as several results models. They can be seen and reviewed both interactively and via commands through an input file.

This documentation is completed with the one dealing with the classes of the program (written in `HTML` and generated with `javadoc`).

The definition and solution of the models can be performed both interactively and thanks to the use of a file input. The latter is preferable due to its greater control and speed while defining a model, the use of the Graphic User Interface is only advisable when checking and postprocessing results.

2 Starting the program

To start the program, the operative system has to have the JAVA virtual machine installed. A free runtime environment may be directly downloaded from the Sun's webpage `www.java.sun.com`, but other implementations of the virtual machine exist.

To start the program under Linux, a command archive named `jstruct` should be invoked, under Windows, an archive named `jstruct.bat` is given so that it can be also directly started.

3 Graphic User Interface

The graphic interface allows to develop several tasks. In it, there are two option bars:

- Menu Bar. The menu bar has several submenus:

  - `File`. In this menu, it can be selected to start a `new` analysis (cleaning the stored databases), to `open` an existing input file of the program or to `exit` the program
  - `Model`. This menu displays a grid with the existing `general` information, `nodes`, `elements`, `materials`, `boundaries`, `linear spring`, `nodal loads` and `element loads`. These informations can be edited, changed added or removed in the grid.
  - `Calc`. Performs a linear calculation of the model
  - `Draw`. It draws in the main frame of the program some of the options over the model: `mesh`, `numbers` (mesh with numbers). It may also display the following results: `deformed mesh`, `axial force`, `shear force` and `bending moment`. This menu also allows to `save` the current picture into a file, also, two drawing menus can be displayed, concerning `pan-zoom` and a `drawing menu` to select models and draw the desired information
  - `List`. It generates several list files being: a jst `input file` of the current model, and it prints in a friendly way the `models` and `results`

- Button Bar. The button bar has several options that can also be accesed via the menu bar but that act as a shortcut of them:
  - `new file`. Same as in the `File` menu
  - `open file`. Same as in the `File` menu
  - `pan-zoom`. Displays the `pan-zoom` and`drawing menu`
  - `save the current picture`. Same as in the `Draw` menu
  - `properties of the model`. Displays the information grid
  - `solve the model`. Same as in the `Calc` menu

4 File Input

The input commands have to be introduced in an input file that is read by the program and whose meaning is determined in the the `ModelLoader.java` class, who performs several tasks. If a precise knowledge of these tasks is desired, then it is advised to deeply review the source code.

Focusing on a typical `JSTRUCT` file, it may contain the following commands:

- `title`
  Specifies the title of the analysis and the force and length units used to define the models. The information should be given in the following way:

```
title,Framed bridge
units,KN,m
```

- `n`
  Specifies that a node is going to be created, it should be followed by the identifier of the node, the x coordinate and the y coordinate:

  ```
  n,1,0.00,0.00
  ```

- `ndele`
  Deletes a node that has been previously introduced (usefull when more than one model is defined), in order to do it, the node identifier has also to be given in the following way:

  ```
  ndele,1
  ```

- `e`
  Specifies that an element is going to be created, it should be followed by the identifier of the element, the material identifier of the element and finally the first and second nodes:

  ```
  e,1,2,101,102
  ```

- `edele`
  Deletes an element that has been previously introduced (usefull when more than one model is defined), in order to do it, the element identifier has also to be given in the following way:

  ```
  edele,1
  ```

- `m`
  Specifies that a material is going to be created, it should be followed by the identifier of the material, the elastic modulus value E_m, the area A and finally the inertia I of the cross section of the beam

  ```
  m,1,1000000,1.6,2.9
  ```

- `mdele`
  Deletes a material that has been previously introduced (usefull when more than one model is defined), in order to do it, the material identifier has also to be given in the following way:

  ```
  mdele,1
  ```

- `b`
  Specifies that a boundary condition is going to be introduced, it should be followed by the identifier of the node where it is going to be applied, then the code of the bounday in the x direction ($1 \leftrightarrow$ Restraint; $0 \leftrightarrow$ Free), the code in the y direction and finally the z axis (rotations) restraint, for instance:

  ```
  b,103,1,0,1
  ```

- `bdele`
  Deletes a boundary condition that has been previously introduced (usefull when more than one model is defined), in order to do it, the node identifier of the boundary has also to be given in the following way:

  ```
  bdele,1
  ```

- `ls`
  Specifies that a linear spring is going to be introduced, it should be followed by the identifier of the node where it is going to be applied, then the value of the spring constant in the x direction, the value in the y direction and finally the value in the z axis (rotations), for instance:

  ```
  ls,105,1000,20000,0
  ```

- `lsdele`
  Deletes a linear spring that has been previously introduced (usefull when more than one model is defined), in order to do it, the node identifier of the linear spring has also to be given in the following way:

  ```
  lsdele,1
  ```

- `hn`
  Specifies that a nodal load is going to be introduced, it should be followed by the identifier of the node where it is going to be applied, then the value of the load in the x direction, the value in the y direction and finally the value in the z axis (bending moment), for instance:

  ```
  hn,15,100,200,200
  ```

- `hndele`
  Deletes a nodal load that has been previously introduced (usefull when more than one model is defined), in order to do it, the node identifier of the nodal load has also to be given in the following way:

```
hndele,1
```

- `he`
  Specifies that an element load is going to be introduced, it should be followed by the identifier of the element where it is going to be applied, then the type of element load (currently only the type 1 is available, meaning an uniform load), then the value of the element load and finally an auxiliary parameter (for loads type 1, it means the angle of the load with the element in degrees) for instance:

  ```
  he,3,1,200,90
  ```

- `hedele`
  Deletes an element load that has been previously introduced (usefull when more than one model is defined), in order to do it, the number of the introduced element load (starting with 1 for the first element load introduced) has to be specified (because different type of loads should be defined over the same element):

  ```
  hndele,1
  ```

- `defModel`
  It creates a new Model from the previous data and adds it to the collection of models. It should be followed by a short name to identify the defined model, for instance:

  ```
  defModel,mod1
  ```

- `solve`
  Solves the currently defined model and stores the resulting information into a results model named as the parameter of this command, for instance

  ```
  solve,resultsMod1
  ```

- `scaleFact`
  Defines a scale factor for the drawing window meaning the ratio of the model length to window dimension. If it is not specified, then a default value of 0.8 is taken. The value is specified as the parameter of the commend:

  ```
  scaleFact,0.7
  ```

- `centerX`
  Defines a point to center in the x axis the draw (refered to the model length to window dimension ratio). If it is not specified, then a default value of 0.0 is taken. The value is specified as the parameter of the commend:

```
centerX,-0.1
```

- `centerY`
  Defines a point to center in the x axis the draw (refered to the model length to window dimension ratio). If it is not specified, then a default value of 0.0 is taken. The value is specified as the parameter of the commend:

  ```
  centerY,0.1
  ```

- `resultsScale`
  Defines the scale to be given to the graphical results (refered to the model length to window dimension ratio). If it is not specified, then a default value of 0.25 is taken. The value is specified as the parameter of the commend:

  ```
  resultsScale,0.30
  ```

- `drawMesh`
  Draws the mesh of the current model.

- `drawNumbers`
  Draws the mesh of the current model with the numbers of nodes, elements, ...

- `drawDefo`
  Draws the deformed mesh of the current results model.

- `drawForc`
  Draws the sectional forces of the current results model. A parameter specifying which results is to be plotted should be given (being the possible labels "Axial", "Shear" and "Moment") and other parameter should be typed concerning the type of desired draw (being the possible labels "Sharpen", "Smoothed" and "Unfilled")), for instance:

  ```
  drawForc,Axial,Sharpen
  ```

- `savePict`
  Saves the current draw into a png file whose name has to be specified:

  ```
  savePict,mesh.png
  ```

- `printModel`
  Prints in a friendly way the information concerning the different defined models (each model is defined into a separate file).

- `printModRes`
  Prints in a friendly way the information concerning the different calculated results models (each results model is defined into a separate file).

5 Example of jst input file

In this section an example of an input file where several models and results models are created is presented. The meaning of the different commands is the one expalined in the previous section, where some options have been commented to be activated if they are wanted to be seen:

```
title,Framed structure
units,KN,m

n,1,0.00,0.00
n,2,1.00,0.00
n,3,2.00,0.00
n,4,3.00,0.00
n,5,4.00,0.00
n,6,5.00,0.00
n,7,6.00,0.00
n,8,7.00,0.00
n,9,8.00,0.00
n,10,9.00,0.00
n,11,10.00,0.00
n,101,0.00,-1.00
n,102,0.00,-2.00
n,103,0.00,-3.00
n,104,0.00,-4.00
n,105,0.00,-5.00
n,106,0.00,-6.00
n,107,0.00,-7.00
n,108,0.00,-8.00
n,109,0.00,-9.00
n,110,0.00,-10.00


e,1,1,1,2
e,2,1,2,3
e,3,1,3,4
e,4,1,4,5
e,5,1,5,6
e,6,1,6,7
e,7,1,7,8
e,8,1,8,9
e,9,1,9,10
e,10,1,10,11
```

```
e,101,2,1,101
e,102,2,101,102
e,103,2,102,103
e,104,2,103,104
e,105,2,104,105
e,106,2,105,106
e,107,2,106,107
e,108,2,107,108
e,109,2,108,109
e,110,2,109,110


b,110,1,1,1
b,11,1,1,0

m,1,1000000,1,1
m,2,1000000,2,1

hn,6,0.00,-2000.00,0.00

ls,7,0.00,1000.00,0.00

defModel,mod1

scaleFact,0.7
centerX,0.0
centerY,0.0
resultsScale,0.25

drawMesh
!savePict,pruebas/mesh_pict.png
!drawNumbers

solve,cargasMod1

hn,8,0.00,-3000.00,0.00
hndele,6
lsdele,7

defModel,mod2
solve,cargasMod2

drawDefo
```

```
drawForc,Moment,Smoothed
!printModel,alls
printModRes,alls
```