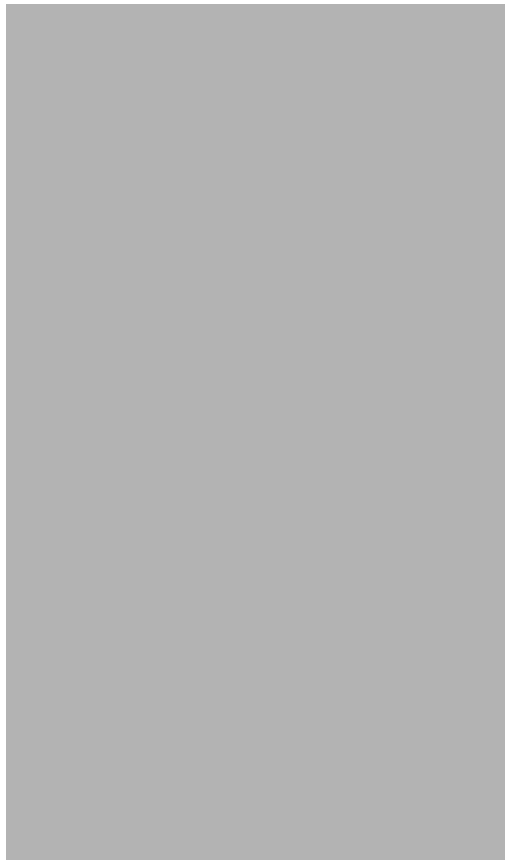

EXERCISE 10

Miracle Thermal Analysis



Objectives:

- Add Thermal Analysis for Miracle.



Problem Description:

In this Exercise we edit and rename the PCL function *exercise_10.pcl* to add the analysis code **Miracle** which has both thermal and structural analysis capabilities. We then execute the PCL function `load_miracle()` to create the **Miracle** preference.

Suggested Exercise Steps:

- Copy and edit *exercise_10.pcl* to create analysis code **Miracle**.
- Compile *exercise_10.pcl* into **miracle.plb**
- Open **miracle_template.db**.
- Display the **Analysis form**.

Exercise Procedure:

1. Either use vi or jot as the text editing tool. Open the file named *exercise_10.template* and add a Function called “thermal” to the `miracle_load_aom_data` class.

Copy the contents of the *structural* function word for word to the **thermal** function.

2. Compile *exercise_10.pcl* into **miracle.plb**.

Type **p3** at the prompt and **<return>**.

After the main menu and command window appear, type:

```
!!compile exercise_10.pcl into miracle.plb
```

The above command must be typed before any database is opened. Resolve any compile errors by editing **exercise_10.pcl** and re-executing the compile command. As mentioned before, it may be wise to rename the *exercise_10.pcl* function `miracle_load_aom_data.pcl` to be consistent with P3's conventions.

3. Open the **miracle_template.db** database.

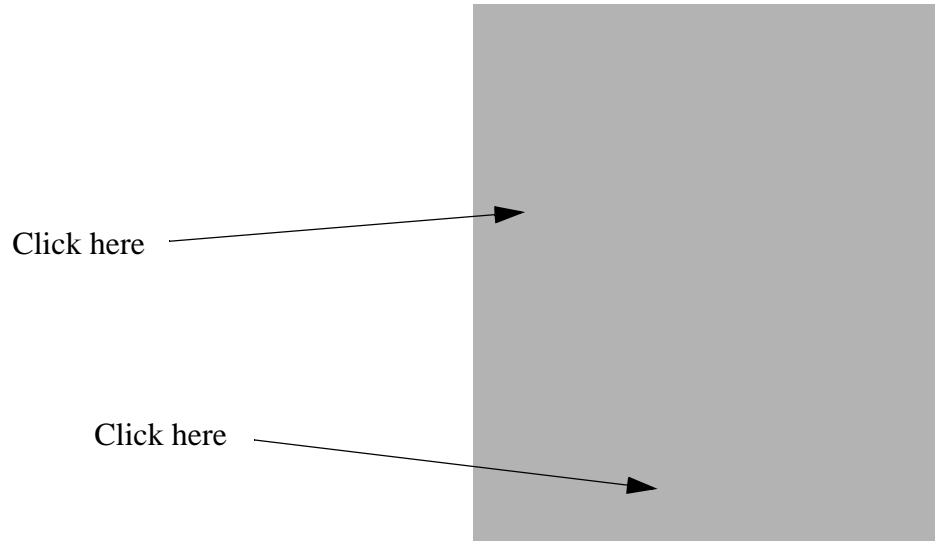
Select **Open Database...** from the *file* pull down menu.

Select **miracle_template.db** from the database list.
Click **OK**.

4. Test the function.

Verify that *miracle preference* has been added.

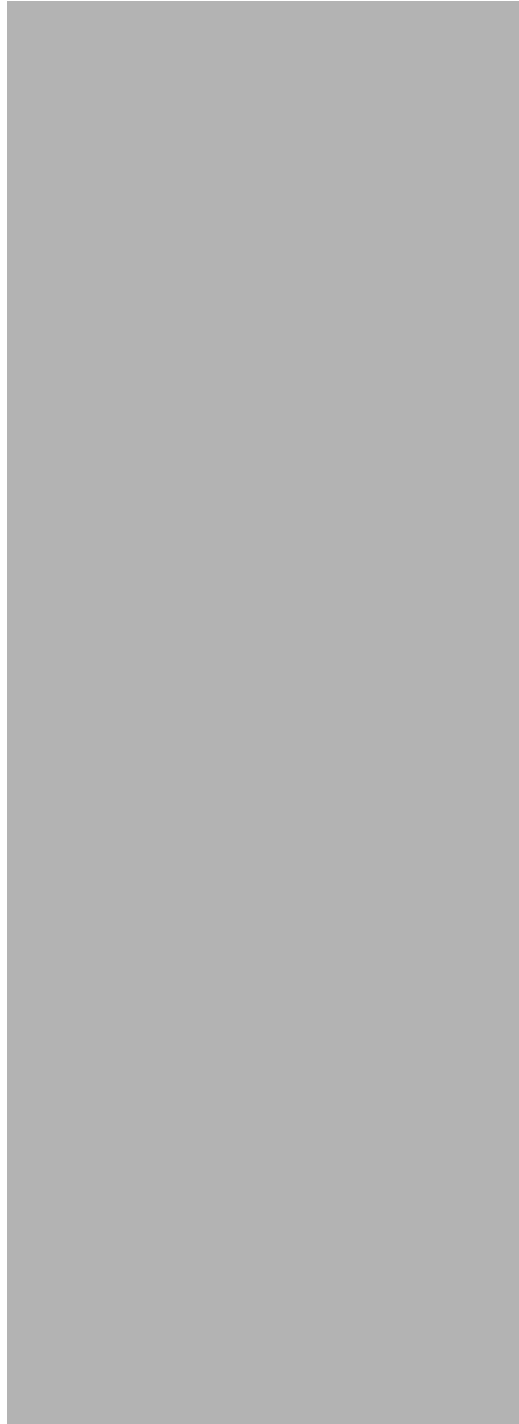
Select **Analysis...** from the *Preference Menu*. Click on the *Analysis Type* Option menu and select **thermal**.



5. Verify that the analysis form is updated for the Miracle preference

Exercise 10

Select the **Analysis Application** radio button on the *Control Panel*. The *Analysis* form should appear as shown below:



Sample Solution

```

/* $Header: /madrid/users9/pflib/pcl/custom/RCS/
miracle_load_aom_data.pcl,v 1.2 93/02/21 13
:45:28 sprack Exp $ */

/**$h */
/*
 * Purpose:
 * Define the option menu selections, button labels, and
 * button pcl classes for Miracle.
 */

CLASS miracle_load_aom_data

/*-----
*$ FUNCTION structural
*
 * Purpose:
 * Load the option menu data for "Miracle-Structural".
 */

FUNCTION structural( num_actions, action_items, @
num_objects, object_items, @
num_methods, method_items, @
num_buttons, button_labels, @
button_callbacks, preference_class, @
callback_diagnostics )
/*
 * Local declarations:
 */
INTEGER num_actions,i
STRING action_items[]()
INTEGER num_objects()
STRING object_items[]()
INTEGER num_methods()
STRING method_items[]()
INTEGER num_buttons()
STRING button_labels[]()

```

Exercise 10

```
STRING button_callbacks[]()
STRING preference_class[]
LOGICAL callback_diagnostics
/*
 * Define the Actions, Objects and Methods. Note that at least
 * one actions must exist and for every action, one object
 * must exist. Methods are optional.
 */
num_actions = 3

action_items(1) = "Analyze"
action_items(2) = "Read Results File"
action_items(3) = "Read Input File"

num_objects(1) = 2
num_objects(2) = 1
num_objects(3) = 1

object_items(1,1) = "Entire Model"
object_items(1,2) = "Current Group"
object_items(2,1) = "Results Entities"
object_items(3,1) = "Model Data"

num_methods(1,1) = 3
num_methods(1,2) = 3
num_methods(2,1) = 1
num_methods(3,1) = 1

method_items(1,1,1) = "Full Run"
method_items(1,1,2) = "Check Run"
method_items(1,1,3) = "Input File Only"
method_items(1,2,1) = "Full Run"
method_items(1,2,2) = "Check Run"
method_items(1,2,3) = "Input File Only"
method_items(2,1,1) = "Translate"
method_items(3,1,1) = "Translate"
/*
 * Define the number of buttons for each action-object
```

```
* combinations. Then define the button labels and
* callbacks.
*/
num_buttons(1,1) = 5
num_buttons(1,2) = 5
num_buttons(2,1) = 2
num_buttons(3,1) = 2

FOR( i = 1 TO 2 )
  button_labels(1,i,1) = "Translation Parameters..."
  button_labels(1,i,2) = "Solution Type..."
  button_labels(1,i,3) = "Solution Parameters..."
  button_labels(1,i,4) = "Select Load Cases..."
  button_labels(1,i,5) = "Output Requests..."
  button_callbacks(1,i,1) = "miracle_analyze_tp"
  button_callbacks(1,i,2) = "miracle_analyze_sol_typ"
  button_callbacks(1,i,3) = "solution_param"
  button_callbacks(1,i,4) = "miracle_loadcases"
  button_callbacks(1,i,5) = "miracle_analyze_out_switch"
END FOR

button_labels(2,1,1) = "Button One"
button_labels(2,1,2) = "Button Two"
button_labels(3,1,1) = "Button One"
button_labels(3,1,2) = "Button Two"

button_callbacks(2,1,1) = " "
button_callbacks(2,1,2) = " "
button_callbacks(3,1,1) = " "
button_callbacks(3,1,2) = " "

/*
* Define the class for general button functions, such as
* the "apply" function.
*/
preference_class = " "

callback_diagnostics = FALSE

END FUNCTION /* structural */
```

Exercise 10

```
/*-----  
*$$ FUNCTION thermal  
*  
* Purpose:  
* Load the option menu data for "Miracle-Thermal".  
*/  
  
FUNCTION thermal( num_actions, action_items, @  
num_objects, object_items, @  
num_methods, method_items, @  
num_buttons, button_labels, @  
button_callbacks, preference_class, @  
callback_diagnostics )  
/*  
* Local declarations:  
*/  
INTEGER num_actions, i  
STRING action_items[ ]()  
INTEGER num_objects()  
STRING object_items[ ]()  
INTEGER num_methods()  
STRING method_items[ ]()  
INTEGER num_buttons()  
STRING button_labels[ ]()  
STRING button_callbacks[ ]()  
STRING preference_class[ ]  
LOGICAL callback_diagnostics  
/*  
* Define the Actions, Objects and Methods. Note that at least  
* one actions must exist and for every action, one object  
* must exist. Methods are optional.  
*/  
num_actions = 3  
  
action_items(1) = "Analyze"  
action_items(2) = "Read Results File"  
action_items(3) = "Read Input File"
```

```
num_objects(1) = 2
num_objects(2) = 1
num_objects(3) = 1

object_items(1,1) = "Entire Model"
object_items(1,2) = "Current Group"
object_items(2,1) = "Results Entities"
object_items(3,1) = "Model Data"

num_methods(1,1) = 3
num_methods(1,2) = 3
num_methods(2,1) = 1
num_methods(3,1) = 1

method_items(1,1,1) = "Full Run"
method_items(1,1,2) = "Check Run"
method_items(1,1,3) = "Input File Only"
method_items(1,2,1) = "Full Run"
method_items(1,2,2) = "Check Run"
method_items(1,2,3) = "Input File Only"
method_items(2,1,1) = "Translate"
method_items(3,1,1) = "Translate"
/*
* Define the number of buttons for each action-object
* combinations. Then define the button labels and
* callbacks.
*/
num_buttons(1,1) = 5
num_buttons(1,2) = 5
num_buttons(2,1) = 2
num_buttons(3,1) = 2

FOR( i = 1 TO 2 )
button_labels(1,i,1) = "Translation Parameters..."
button_labels(1,i,2) = "Solution Type..."
button_labels(1,i,3) = "Solution Parameters..."
button_labels(1,i,4) = "Select Load Cases..."
button_labels(1,i,5) = "Output Requests..."
```

Exercise 10

```
button_callbacks(1,i,1) = "miracle_analyze_tp"
button_callbacks(1,i,2) = "miracle_analyze_sol_typ"
button_callbacks(1,i,3) = "miracle_analyze_sol_switch"
button_callbacks(1,i,4) = "analysis_analyze_loadcases"
button_callbacks(1,i,5) = "miracle_analyze_out_switch"
END FOR

button_labels(2,1,1) = "Button One"
button_labels(2,1,2) = "Button Two"
button_labels(3,1,1) = "Button One"
button_labels(3,1,2) = "Button Two"

button_callbacks(1,2,1) = " "
button_callbacks(1,2,2) = " "
button_callbacks(1,2,3) = " "
button_callbacks(2,1,1) = " "
button_callbacks(2,1,2) = " "
button_callbacks(3,1,1) = " "
button_callbacks(3,1,2) = " "

/*
 * Define the class for general button functions, such as
 * the "apply" function.
 */
preference_class = " "

callback_diagnostics = FALSE

END FUNCTION /* thermal */

END CLASS /* miracle_load_aom_data */
```

