Introduction to the Turing GUI Library - Part #1

The program below is an example of a program which uses the Turing Graphical User Interface (GUI) Library. The GUI Library contains a number of predefined procedures and functions which are used to create objects or "widgets" on the screen. These objects are used to allow the user to easily interact with the computer.

When creating a program using the GUI library, keep the following points in mind:

1) The GUI Library is not included in the standard code and therefore every program using any of the routines must begin with the following line of code typed exactly as shown: **import GUI in "%oot/lib/GUI"**
2) Most GUI widgets have a Create predefined function which allows the programmer to assign parameter values to specify the size, location, the action procedure (ie. the procedure to be called when the widget is manipulated) and/or other things.
3) Each GUI widget is identified by a widgetID variable. This variable is usually an integer and is assigned a value when the widget is created using the Create predefined function described in (2).
4) Programs which use these objects constantly look for an appropriate user action and act upon these user actions as required. This search is done in the following code which must be at the end of every program using the GUI library:

```
loop
   exit when GUI.ProcessEvent
end loop
```

Required:

1. Copy the following program and execute it on your computer. Save it as **Slider Example**

```
import GUI in "%oot/lib/GUI"   % Loads the Graphical User Interface (GUI) Library

% Global Variables Section
var horSlider, verSlider : int   % Variables for the slider widgets

% Action Procedures Section
proc ShowHorSliderValue (value : int)
   % Action procedure for the horizontal slider
   % This procedure responds to the user action of "sliding" the horizontal slider
   % value - the value of the slider that is passed to the action procedure

   locate (1, 1)
   put "The value of the horizontal slider ", value, " "
end ShowHorSliderValue

proc ShowVerSliderValue (value : int)
   % Action procedure for the vertical slider
   % This procedure responds to the user action of "sliding" the vertical slider
   % value - the value of the slider that is passed to the action procedure
   locate (2, 1)
   put "The value of the vertical slider ", value, " "
end ShowVerSliderValue
```

```
% GUI Object Creation Section
GUI.SetBackgroundColor (blue)        % Changes the background color of the windows
verSlider := GUI.CreateVerticalSlider (10, 50, 200, 0, 100, 50,  ShowVerSliderValue)
horSlider := GUI.CreateHorizontalSlider (50, 50, 200, 0, 100, 50,  ShowHorSliderValue)

/* This loop checks for an event (ie. mouse click or key being pressed) and then performs an
appropriate action and usually calls the required event procedure.*/
loop
   exit when GUI.ProcessEvent
end loop
```

2. Add the following code to your **Slider Example** program. Execute it and insure that it works

Add to the Global Variables Section

```
var buttonOne, buttonTwo : int % Variables for button objects
```

Add to the Action Procedures Section

```
proc AddSliderValue
% Action procedure for the Add button
 % Responds to the user action of clicking the Add button by outputting the sum of the two
 % slider values.
   var sum : int     % sum of the two slider values
   sum := GUI.GetSliderValue (verSlider) +  GUI.GetSliderValue (horSlider)
      locate (3,1)
      put "":50
      locate (3,1)
   put "The sum of the two slider values is ", sum
end AddSliderValue

proc MultiplySliderValue
/* Action procedure for the Multiply button
   Responds to the user action of clicking the Multiply button by outputting the product of % the two
   slider values. */
    var product : int       % product of the two slider values
    product := GUI.GetSliderValue (verSlider) *   GUI.GetSliderValue (horSlider)
      locate (3,1)
      put "":50
      locate (3,1)
   put "The product of the two slider values is ", product
end MultiplySliderValue
```

Add to the GUI Object Creation Section

```
buttonOne := GUI.CreateButton (200, 200, 0, "Multiply", MultiplySliderValue)
buttonTwo:= GUI.CreateButton (200, 150, 0, "Add", AddSliderValue)
```

3. Improve your **Slider Example** program by:
   - Changing the background colour of the text to blue and the text colour to white.
   - Adding buttons and corresponding procedures to output the difference and the quotient of slider values.
   - Adding a section that will allow the user to input the maximum value of the sliders.

- Adding a third slider (horizontal or vertical) and updating your program to show the results of the three slider values.
- Labeling the sliders with an appropriate label (hint: use the CreateLabel of CreateLabelFull predefined procedures.)