Introduction to the Turing GUI Library - Part #2

In this assignment, you will learn about the GUI check box and radio button widgets and the Font predefined unit.

Type the following program which will create a series of check boxes and radio buttons to allow the user to choose whether or not to display a series of messages using a variety of backgrounds.

```
import GUI in "%oot/lib/GUI"

% Global Variable Section
var cb1, cb2 : int          % The check box objects
var rb1, rb2, rb3 : int     % The radio button objects
var clr := grey             % Colour of the text outputted on the screen

% Action Procedure Section
proc BackgroundChoice
   % Action procedure for the three radio buttons
   % The background colour is changed according to the users choice

   if rb1 = GUI.GetEventWidgetID then
      GUI.SetBackgroundColour (grey)
      clr := grey
   elsif rb2 = GUI.GetEventWidgetID then
      GUI.SetBackgroundColour (green)
      clr := green
   elsif rb3 = GUI.GetEventWidgetID then
      GUI.SetBackgroundColour (yellow)
      clr := yellow
   end if
end BackgroundChoice

function FIND_HEIGHT (font : int) : int
    /* The height in pixels of the font used to output the message is returned using the Font.Size
    predefined procedure */

   var height : int                % Stores the height of the font
   var junk1, junk2, junk3 : int   % Store values returned by Font.Sizes that are not required

   Font.Sizes (font, height, junk1, junk2, junk3)
   result height
end FIND_HEIGHT

proc DrawFont (font : int, cb : int, yPos : int)
   /* Called by the  DrawFontBig and DrawFontMedium procedures
   The message is outputted on the monitor in the correct position if check box is checked and erases the
   text if the check box is unchecked.
   font - the font identification variable for the chosen font
   cb - the checkbox chosen
   yPos - the bottom left position of the message being outputted */

   if GUI.GetCheckBox (cb) then
      Font.Draw ("A message", 150, yPos, font, red)
   else
      drawfillbox (150, yPos - 10, 150 + Font.Width ("A message", font),
         yPos + FIND_HEIGHT (font), clr)
   end if
end DrawFont
```

```
proc DrawFontBig (filled : boolean)
   % Action procedure for the first checkbox (cb1)
    % The DrawFont procedure is called to draw the message in the appropriate spot

   var font := Font.New ("sans serif:24:bold")   % A 24pt sans serif font is selected
   DrawFont (font, cb1, 200)
end DrawFontBig

proc DrawFontMedium (filled : boolean)
   % Action procedure for the second check box (cb2)
    % The DrawFont procedure is called to draw the message in the appropriate spot

   var font := Font.New ("palatino:15:bold")      % A 15pt palitino font is selected
   DrawFont (font, cb2, 150)
end DrawFontMedium

% GUI Creation Section
cb1 := GUI.CreateCheckBox (100, 100, "Draw Big Font", DrawFontBig)
cb2 := GUI.CreateCheckBox (100, 75, "Draw Big Medium", DrawFontMedium)
rb1 := GUI.CreateRadioButton (250, 100, "Grey Background", 0,  BackgroundChoice)
rb2 := GUI.CreateRadioButton ( - 1, - 1, "Green Background", rb1,  BackgroundChoice)
rb3 := GUI.CreateRadioButton ( - 1, - 1, "Yellow Background", rb2, BackgroundChoice)
GUI.SetBackgroundColour (grey)

loop
   exit when GUI.ProcessEvent
end loop
```

Note
Notice that when the background colour changes, the message disappears. When an any item is drawn on the monitor, changing the background colour will erase it. Using Font.Draw "draws" the text on the screen. Remember that text drawn on the screen is no longer a string!

Improve your program by doing the following:

1) The problem with the program as it is that the check boxes are still checked when the background colour is changed even though the messages do not appear. Have your computer re-set the values of the check boxes when the colour changes. (Hint: Use the GUI.SetCheckBox predefined procedure at the end of the BackgroundChoice procedure)

2) Add two more check boxes to draw two more <u>different</u> fonts. Use the Turing manual and/or the Help Functions (F9 and F10) to learn about the different fonts you can use. Note that different computers may have different font lists!

3) Improve one of the check boxes created in (2) to allow the user to use a shortcut key. (Hint: Use the GUI.CreateCheckBoxFull function.)

4) Add a final check box called "Erase" which will erase all the text outputted on the screen and "uncheck" all the checkboxes.