

## Project 9 – Poker machine

In this project we will use the *if* selection to build a simple poker machine simulation. We will also review the software development cycle we discussed in Unit 2.

### Definition

We will produce a simple poker machine simulation that displays three random numbers when the user pulls a handle. If any of the three numbers is a 7 the user “wins”.

### Specification

Input will be via a track bar that the user “pulls”. This will initiate the generation of three random numbers which will be displayed on three labels. At the bottom of the pull the track bar will reset to the top. If any of the three numbers is a 7 then a picture of money will be displayed to show the win. The program will be prepared as a Delphi application with on-screen cues to the user.

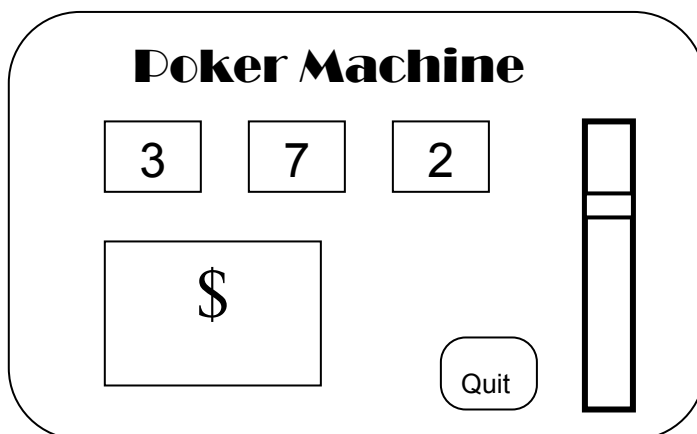
### Design

The algorithm for this process in pseudocode is:


```
if TRACKPOSITION = 100 then
    TRACKPOSITION = 0
    ONE = random 10
    TWO = random 10
    THREE = random 10
    write ONE
    write TWO
    write THREE
    if ONE = 7 or TWO = 7 or THREE = 7 then
        write MONEYIMAGE
```

The first *if* checks to see if the track bar has been pulled all the way to the bottom. If it has it resets the track bar to the top. Three random numbers (*one*, *two* and *three*) are generated and displayed. The second *if* checks to see if any of the values is 7; if it is then the picture of the money is displayed. (This is an example of a *nested if*, one *if* statement is inside another. We will investigate this shortly.)

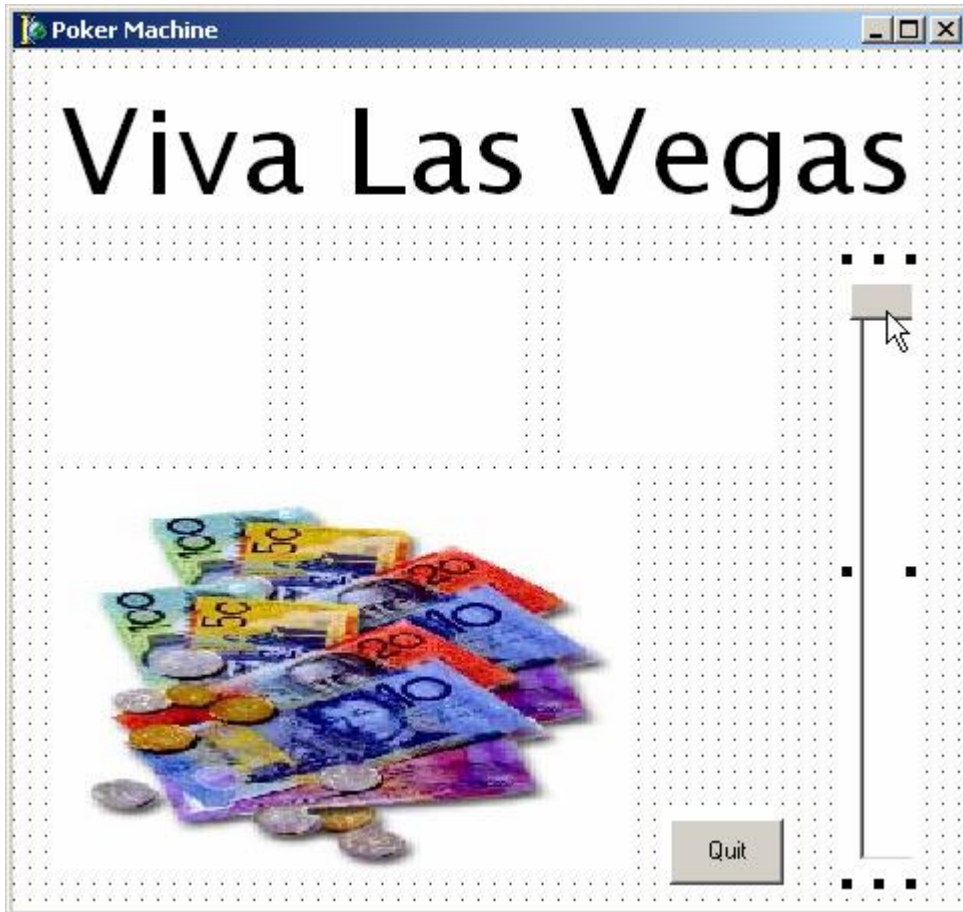
Our screen will need a title, three labels for the number display, a vertical track bar for the pull handle, an image to display when money is won and an exit button. Sketching it might look like this:



## Implementation

1. Start Delphi and create a new application.
2.
  - a. Place three labels to display the numbers, and a label for the title on the form. (For once we will not rename these labels because *Label1*, *Label2* and *Label3* will be appropriate names for the three numbers to be displayed.)
  - b. Set suitable fonts and properties for the labels but delete their current captions. Place a suitable title in the top label.
  - c. From the *Win32* tab of the component palette add a track bar  with a vertical orientation (*trVertical*). Call it *HandleTrack*, set its *Min* property to 0 and *Max* to 100. Make *TickMarks* *tmBoth* and *TickStyle* *tsNone*. Set *ThumbLength* to 30 to make the handle wider.
  - d. Place an image control on the form. Link it to a picture of money and set its *Visible* property to *False*. (It will only appear when there is a win.)
  - e. Finally add a button to exit or quit and code it.
  - f. Save your work to the Project 9 folder.

Your screen should now look something like this:



3. Double click on the track bar and add the following code:

```
poker.pas
poker

procedure TPokerForm.HandleTrackChange(Sender: TObject);
//Use trackbar to simulate poker machine
var one, two, three : integer;
begin
  randomize;
  if HandleTrack.Position = 100 then //if end of pull
  begin
    HandleTrack.Position := 0; //reset handle to top
    one := random(10) + 1; //select values
    two := random(10) + 1;
    three := random(10) + 1;
    Label1.Caption := IntToStr(one); //display values
    Label2.Caption := IntToStr(two);
    Label3.Caption := IntToStr(three);
    if (one = 7) or (two = 7) or (three = 7) then //check for win
    begin
      MoneyImage.Visible := true; //win
      beep;
    end
    else
      MoneyImage.Visible := false; //no win
    end;
  end;
end;
```

4. a Add tool tips to the track bar and quit button.  
b Change the *cursor* property of the track bar to *crHandPoint*.
5. Re-save the project and run it by “pulling” the handle of the track bar down.



Besides the use of the nested *if* the main point of interest in this program is the use of a random number generator. For this we need two functions:

```
randomize;
and
one := random(10) + 1;
two := random(10) + 1;
three := random(10) + 1;
```

The *randomize* reads a digit from your computer’s system clock to get a seed number to start with. The *random* function then works from this seed and generates a value between 0 and 1. This value is multiplied by the value in brackets and converted to an integer. The +1 ensures that the number stored in the variable is from 1 to 10, and not between 0 and 9.

To understand the need for the +1 we will look at two examples. Say the *random* function generates a very low value such as 0.012. This value is multiplied 10 (0.12) and rounded down to 0. A high random value of say 0.909 multiplied by 10 (9.09) and rounded down is 9. Without the +1 random values will be only between 0 and 9.

Note also the use of *beep* to generate a sound when a 7 appears.

Now to complete the software development cycle:

Test