```
unit P90_Likelihood;

interface

uses SysUtils, LCRGlobals, VCL.FlexCel.Core, FlexCel.XlsAdapter;

type
  TP90_Likelihood = class
  public
    pBinLSLs1, pBinNoLSLs1: double;
    pBinLSLs2, pBinNoLSLs2: double;
    pBinLSLs3, pBinNoLSLs3: double;
    pBinLSLs4, pBinNoLSLs4: double;
    pBinLSLs5, pBinNoLSLs5: double;

    constructor Create;
    destructor Destroy; override;

    procedure GetpBinValues(aOption, aEstimate: string);
  private
    Xls: TExcelFile;
    function GetP90Value(aOption, aEstimate, aHasLSLs: string; aID: integer):
double;
  end;

implementation

{ TP90_Likelihood }

constructor TP90_Likelihood.Create;
var
  filename: string;
begin
  filename := DataPath + 'P90_Likelihood.xlsx';

  Xls := TXlsFile.Create(Filename, False);
  Xls.ActiveSheetByName := 'Sheet1';
  {
    P90_Likelihood.xlsx
    1 Option    string (4)  LCR, LCRR, LCRI
    2   Estimate  string (4)  Low, High
    3   ID        integer     1 - 5
    4   Category  string (25)
    5   No_LSLs   double
    6   Has_LSLs  double

    column labels in first row
    ID  Category
    5   P90 ? 5 µg/L
    4   5 µg/L < P90 ? 10 µg/L
```

```
    3   10 µg/L < P90 ? 12 µg/L
    2   2 µg/L < P90 ? 15 µg/L
    1   P90 >15 ?g/L
  }
end;

destructor TP90_Likelihood.Destroy;
begin
  FreeAndNil(Xls);

  inherited;
end;

function TP90_Likelihood.GetP90Value(aOption, aEstimate, aHasLSLs: string; aID:
integer): double;
var
  r: integer;
  bFound: boolean;
begin
  bFound := false;

  GetP90Value := -1;

  for r := 2 to Xls.RowCount do
  begin
    if (Xls.GetStringFromCell(r,1) = aOption) and
       (Xls.GetStringFromCell(r,2) = aEstimate) and
       (Xls.GetCellValue(r,3).AsVariant = aID) then
    begin
      if aHasLSLs = 'No' then
        GetP90Value := Xls.GetCellValue(r,5).AsVariant
      else
        GetP90Value := Xls.GetCellValue(r,6).AsVariant;
      bFound := true;
      break;
    end;
  end;

  if not bFound then
    raise Exception.Create('Row not found in P90_Likelihood.xlsx');
end;

procedure TP90_Likelihood.GetpBinValues(aOption, aEstimate: string);
begin
  pBinLSLs1 := GetP90Value(aOption, aEstimate, 'Yes', 1);
  pBinLSLs2 := GetP90Value(aOption, aEstimate, 'Yes', 2);
  pBinLSLs3 := GetP90Value(aOption, aEstimate, 'Yes', 3);
  pBinLSLs4 := GetP90Value(aOption, aEstimate, 'Yes', 4);
  pBinLSLs5 := GetP90Value(aOption, aEstimate, 'Yes', 5);
  pBinNoLSLs1 := GetP90Value(aOption, aEstimate, 'No', 1);
```

```
  pBinNoLSLs2 := GetP90Value(aOption, aEstimate, 'No', 2);
  pBinNoLSLs3 := GetP90Value(aOption, aEstimate, 'No', 3);
  pBinNoLSLs4 := GetP90Value(aOption, aEstimate, 'No', 4);
  pBinNoLSLs5 := GetP90Value(aOption, aEstimate, 'No', 5);
end;

end.
```