```
unit LCRMicroOut;

interface

uses Classes, Generics.Collections, LCRGlobals, SysUtils;

type
  THoldPWS = TList<TCostGenRec>;

  TLCRMicroOut = class
  private
    fPWS : THoldPWS;
    fVars, fCosts : TObject;
    fACtive : boolean;
    fPath:string;
    procedure SaveFiles;
  public

    constructor create(aActive : boolean; aOutPath : string);
    destructor Destroy; override;

    procedure AddPWS(const a : TCostGenRec);
  end;

implementation

{ TLCRMicroOut }

procedure TLCRMicroOut.AddPWS(const a: TCostGenRec);
var b : TCostGenRec;
begin
  if not fActive then exit;
  b := a;
  FPWS.Add(b);
end;

constructor TLCRMicroOut.create(aActive: boolean; aOutPath: string);
begin
  fActive:=aActive;
  fPath:=aOUtPath;
  fPWS := THoldPWS.Create;

  FCosts:=TObject.Create;
  FVars:=TObject.Create;
end;

destructor TLCRMicroOut.Destroy;
begin
```

```
  SaveFiles();
  fVars.Free;
  fCosts.Free;
  fPWS.Free;
  inherited;
end;

procedure TLCRMicroOut.SaveFiles;
var ofVars, ofCosts, ofPWS : TStreamWriter;
    PWS : TCostGenRec;
begin
  if not fActive then exit;
  ForceDirectories(FPath);
  ofVars := TStreamWriter.Create(FPath+'VARS.csv',false,TEncoding.ASCII,32768);
  ofCosts  := TStreamWriter.Create(FPath+'COSTS.csv',false,TEncoding.ASCII,32768);
  ofPWS  := TStreamWriter.Create(FPath+'PWS.csv',false,TEncoding.ASCII,32768);

  ofPWS.WriteLine(TCostGenRec.HeaderString);
  for PWS in FPWS do begin
     ofPWS.WriteLine(PWS.DataString);
  end;

  ofVars.Flush;
  ofCosts.Flush;
  ofPWS.Flush;
  ofVars.Free;
  ofCosts.Free;
  ofPWS.Free;
end;

end.
```