Delphi作业（一）— Review

（一些说明：缺少图片 以及 关于bpl封装在知识库中）

1. 乘法口诀表

1.1. 变量名首字母应该大写

修改结果：

```
Writeln;
```

1.2. 采用多行注释

delphi中的3种注释方法

```
// 单行注释
```

```
{
    这是多行注释
    包含多行内容
    不会被编译器处理
}
```

```
(* 这也是多行注释
可以包含多行内容
同样不会被编译器处理 *)
```

1.3. 删除无意义空行

2. 找质数

2.1. 属于主程序的var类型变量应该直接定义在 Begin上面

修改结果：

```
function IsPrime(n : integer) : boolean

//主程序
var

begin
```

2.2. 关键字后要有空格

修改结果：

```
if (x <= 1) then
```

3. 哥德巴赫猜想

3.1. 规范

- 使用大驼峰命名法(比如isPrime 改为 IsPrime)
- 变量名建议增加可描述性（比如 x 可以改为 InputNumber）
- 函数命名建议使用动词短语（比如 CheckPrime 更符合规范）
- 形参变量可以使用 const 进行修饰

修改结果：

```
function CheckPrime(const n : integer) : boolean
```

3.2. 对输入是否为偶数的校验：缺少对输入是否为偶数的校验；缺少对非数字输入的处理

```
program Project2;

{$APPTYPE CONSOLE}

{$R *.res}

uses
  SysUtils,
  Math;

//判断质数 -- 如果能整除i 返回false。如果循环结束没有发现能整除i 返回true
function CheckPrime(const n:integer):boolean;
var
  i : Integer;
begin
   if (n<=1) then
   begin
     Result := false;
     exit;
```

```
    end;

    for i := 2 to trunc(sqrt(n)) do
    begin
      if(n mod i = 0) then
      begin
        Result := false;  //非质数
        Exit;
      end
    end;

    //运行到这里，说明这个数是质数
    Result := true;
end;

//主程序
var
  InputStr : string;  //输入的数
  x, currentEven, first, second : integer; //currentEven表示当前正在验证的偶数
x用来验证输入是否是数字
  found : boolean; //是否找到质数对
begin
  Writeln('请输入一个大于2的整数（程序将验证2到该数之间的所有偶数）');
  Readln(InputStr);

  //输入的数是否为数字(如果输入的数不是整数，TryStrToInt返回false,结果为true执行该函数
模块)
  if not TryStrToInt(InputStr, x) then
  begin
      Writeln('输入无效，请输入一个数字');
      Readln;
      Exit;
  end;

  //检验是否大于2
  if (x <= 2) then
  begin
    Writeln('输入无效，请输入一个大于2的整数！');
    Readln;
    Exit;
  end;

  //遍历3到x之间的所有偶数
  for currentEven := 3 to x do
  begin
    if (currentEven mod 2 = 0) then //currentEven是偶数且大于2
    begin
      found := false;
      first := 2;
      while (first <= currentEven div 2) and (not found) do //找到一组后立即退
出
      begin
        if CheckPrime(first) then //如果第一个数是质数
        begin
          second := currentEven - first;
          if CheckPrime(second) then //检查第二个数是否为质数
```

```
          begin
            Writeln(currentEven, ' = ', first, ' + ', second);
            found := true; //标记已找到
          end;
        end;
        //步进：如果first是2，则下一步是3；否则步长为2（跳过偶数）    除了2质数都是奇数
        if first = 2 then
          first := 3
        else
          first := first + 2;
      end;

      if not found then
        Writeln(currentEven, ' 未找到符合条件的质数对! '); //理论上不会执行
    end;
  end;

  readln;
end.
```

3.3. bpl封装

```
unit Goldbach;

interface

uses
  SysUtils;

// 检查一个数是否为质数
function IsPrime(const n: Integer): Boolean;

// 验证哥德巴赫猜想
procedure VerifyGoldbachConjecture(UpToNumber: Integer);

implementation

function IsPrime(const n: Integer): Boolean;
var
  i: Integer;
begin
  if n <= 1 then
  begin
    Result := False;
    Exit;
  end;

  for i := 2 to Trunc(Sqrt(n)) do
  begin
    if n mod i = 0 then
    begin
```

```
      Result := False;
      Exit;
    end;
  end;
  Result := True;
end;

procedure VerifyGoldbachConjecture(UpToNumber: Integer);
var
  currentEven, first, second: Integer;
  found: Boolean;
begin
  if UpToNumber <= 2 then
    raise Exception.Create('输入无效，请输入一个大于2的整数！');

  for currentEven := 4 to UpToNumber do
  begin
    if currentEven mod 2 = 0 then
    begin
      found := False;
      first := 2;
      while (first <= currentEven div 2) and (not found) do
      begin
        if IsPrime(first) then
        begin
          second := currentEven - first;
          if IsPrime(second) then
          begin
            // 这里可以改为回调函数或事件，但为了简单起见，我们直接输出
            Writeln(currentEven, ' = ', first, ' + ', second);
            found := True;
          end;
        end;
        if first = 2 then
          first := 3
        else
          first := first + 2;
      end;

      if not found then
        Writeln(currentEven, ' 未找到符合条件的质数对！');
    end;
  end;
end;

exports
  IsPrime,
  VerifyGoldbachConjecture;

end.
```