

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：

```
1 WriteLn;
```

1.2. 采用多行注释

delphi中的3种注释方法

```
1 // 单行注释
```

```
1 {  
2     这是多行注释  
3     包含多行内容  
4     不会被编译器处理  
5 }
```

```
1 (* 这也是多行注释  
2  可以包含多行内容  
3  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：

```
1 function IsPrime(n : integer) : boolean  
2  
3 //主程序  
4 var  
5  
6 begin
```

2.2. 关键字后要有空格

修改结果：

```
1 | 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：

```
1 | 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

```
1 | program Project2;
2 |
3 | {$APPTYPE CONSOLE}
4 |
5 | {$R *.res}
6 |
7 | uses
8 |     SysUtils,
9 |     Math;
10 |
11 | //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
12 | function CheckPrime(const n:integer):boolean;
13 | var
14 |     i : Integer;
15 | begin
16 |     if (n<=1) then
17 |     begin
18 |         Result := false;
19 |         exit;
```

```

20     end;
21
22     for i := 2 to trunc(sqrt(n)) do
23     begin
24         if(n mod i = 0) then
25         begin
26             Result := false; //非质数
27             Exit;
28         end
29     end;
30
31     //运行到这里，说明这个数是质数
32     Result := true;
33 end;
34
35 //主程序
36 var
37     InputStr : string; //输入的数
38     x, currentEven, first, second : integer; //currentEven表示当前正在验证的偶数
39     x用来验证输入是否是数字
40     found : boolean; //是否找到质数对
41 begin
42     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）');
43     Readln(InputStr);
44
45     //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
46     模块)
47     if not TryStrToInt(InputStr, x) then
48     begin
49         writeln('输入无效，请输入一个数字');
50         Readln;
51         Exit;
52     end;
53
54     //检验是否大于2
55     if (x <= 2) then
56     begin
57         writeln('输入无效，请输入一个大于2的整数!');
58         Readln;
59         Exit;
60     end;
61
62     //遍历3到x之间的所有偶数
63     for currentEven := 3 to x do
64     begin
65         if (currentEven mod 2 = 0) then //currentEven是偶数且大于2
66         begin
67             found := false;
68             first := 2;
69             while (first <= currentEven div 2) and (not found) do //找到一组后立即退
70             出
71             begin
72                 if CheckPrime(first) then //如果第一个数是质数
73                 begin
74                     second := currentEven - first;
75                     if CheckPrime(second) then //检查第二个数是否为质数

```

```

73         begin
74             writeln(currentEven, ' = ', first, ' + ', second);
75             found := true; //标记已找到
76         end;
77     end;
78     //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
79     if first = 2 then
80         first := 3
81     else
82         first := first + 2;
83     end;
84
85     if not found then
86         writeln(currentEven, ' 未找到符合条件的质数对! '); //理论上不会执行
87     end;
88 end;
89
90 readln;
91 end.

```

3.3. bpl封装

```

1  unit Goldbach;
2
3  interface
4
5  uses
6      sysutils;
7
8  // 检查一个数是否为质数
9  function IsPrime(const n: Integer): Boolean;
10
11 // 验证哥德巴赫猜想
12 procedure VerifyGoldbachConjecture(UpToNumber: Integer);
13
14 implementation
15
16 function IsPrime(const n: Integer): Boolean;
17 var
18     i: Integer;
19 begin
20     if n <= 1 then
21     begin
22         Result := False;
23         Exit;
24     end;
25
26     for i := 2 to Trunc(Sqrt(n)) do
27     begin
28         if n mod i = 0 then
29             begin

```

```

30     Result := False;
31     Exit;
32 end;
33 end;
34 Result := True;
35 end;
36
37 procedure VerifyGoldbachConjecture(UpToNumber: Integer);
38 var
39     currentEven, first, second: Integer;
40     found: Boolean;
41 begin
42     if UpToNumber <= 2 then
43         raise Exception.Create('输入无效, 请输入一个大于2的整数! ');
44
45     for currentEven := 4 to UpToNumber do
46     begin
47         if currentEven mod 2 = 0 then
48         begin
49             found := False;
50             first := 2;
51             while (first <= currentEven div 2) and (not found) do
52             begin
53                 if IsPrime(first) then
54                 begin
55                     second := currentEven - first;
56                     if IsPrime(second) then
57                     begin
58                         // 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
59                         writeln(currentEven, ' = ', first, ' + ', second);
60                         found := True;
61                     end;
62                 end;
63                 if first = 2 then
64                     first := 3
65                 else
66                     first := first + 2;
67             end;
68
69             if not found then
70                 writeln(currentEven, ' 未找到符合条件的质数对! ');
71         end;
72     end;
73 end;
74
75 exports
76     IsPrime,
77     verifyGoldbachConjecture;
78
79 end.

```

