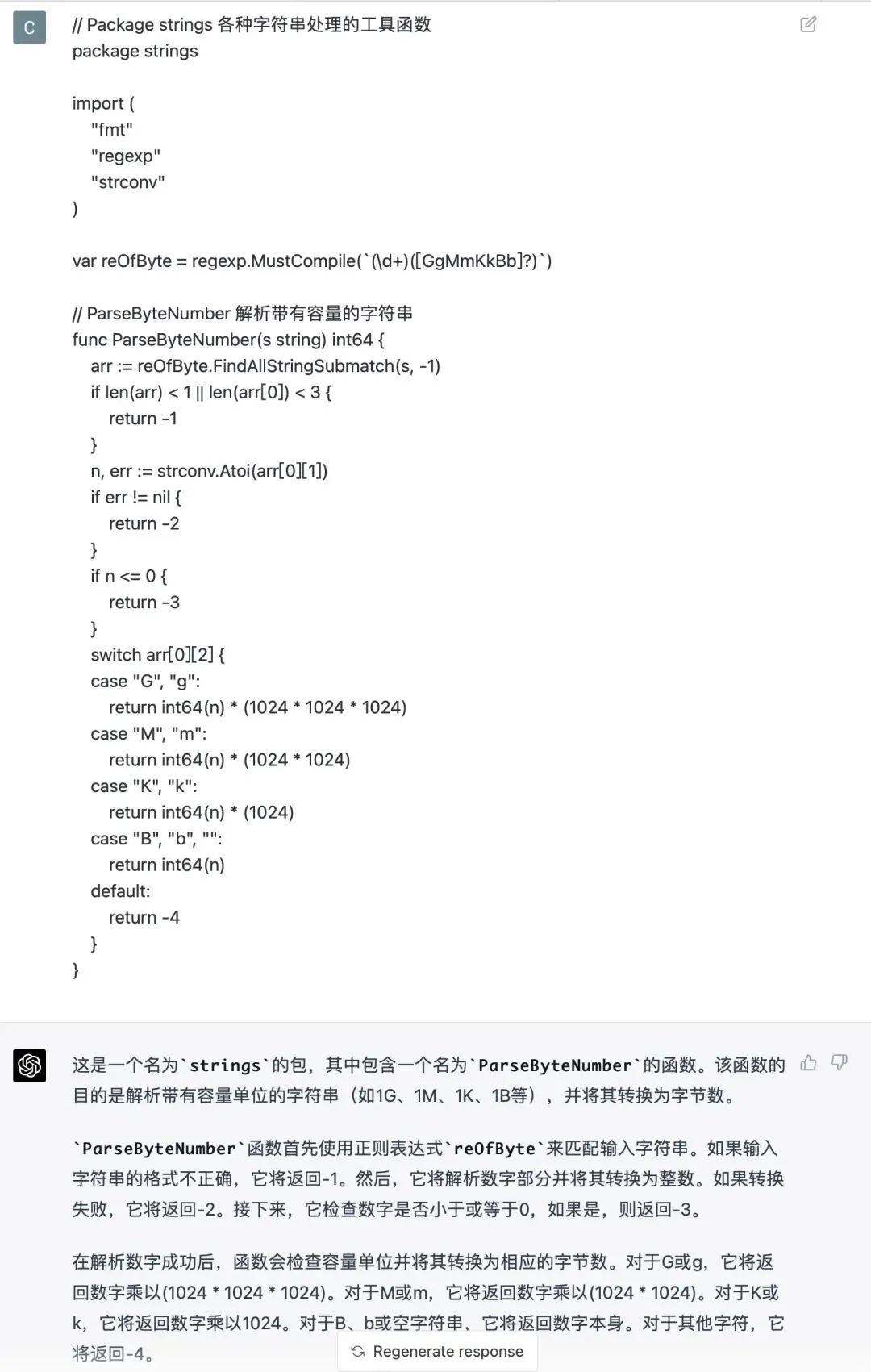
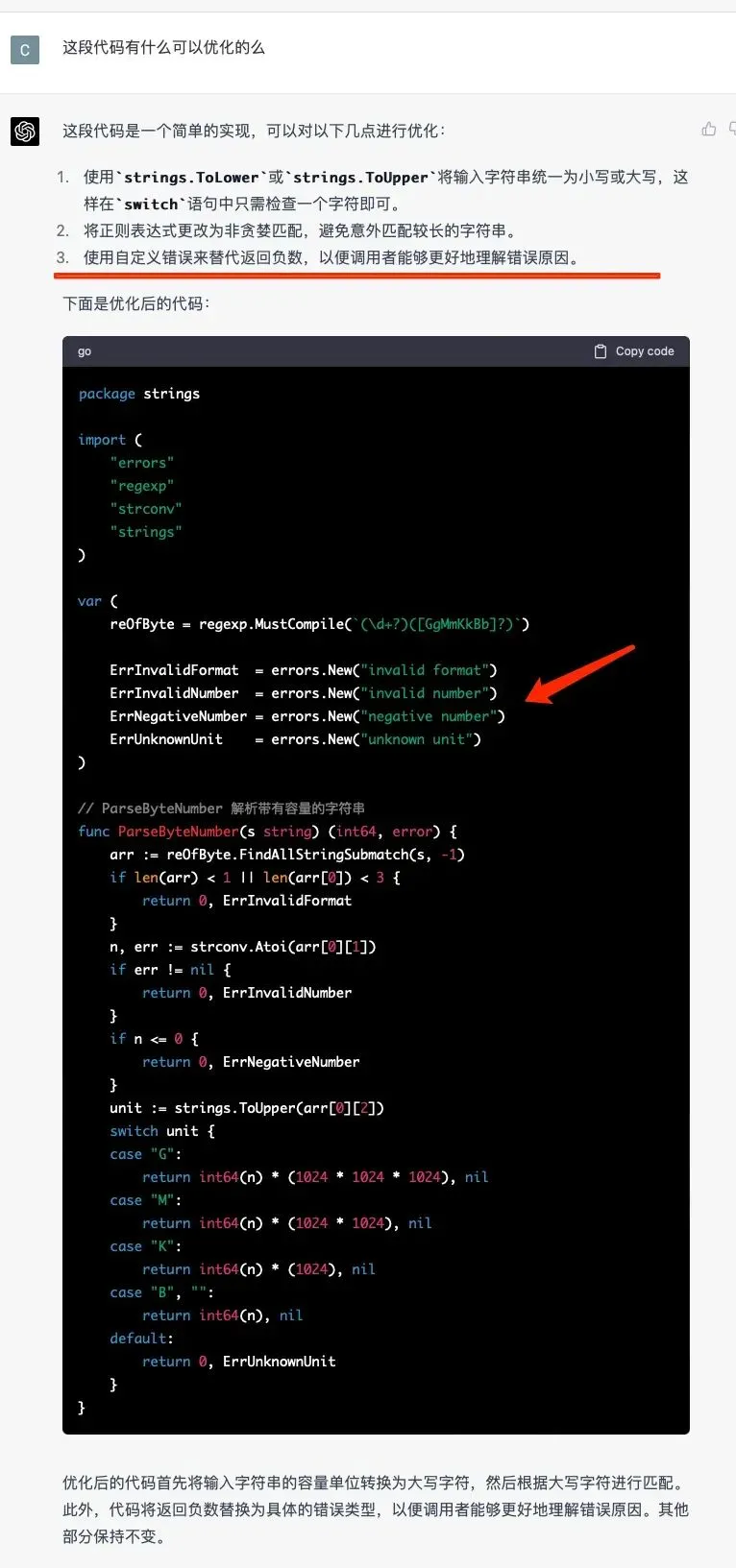
**ChatGPT结对编程实战**

# 重构代码

我们写代码的过程中，往往会因为疏忽，而产出各种bug 和坏味道」。我们来试试 ChatGPT4 能帮我们做什么。下面随机找了一段我们代码仓库里面的不涉密基础代码，发给ChatGPT4。先让ChatGPT看一眼代码：



ChatGPT4 表示它看懂了。接下来我们给 ChatGPT4 提一下重构的需求，看看ChatGPT4 的表现：



ChatGPT4 这些优化，使得代码 readability 了很多。特别是错误码返回的模块，原来的代码真是天坑。但同时我们也发现这个函数实现是不太符合需求的——它只匹配了 substring。例如 XXXX100KBXXX 这类参数也会被错误匹配。我们把这些情况告诉 GPT4，看看它的表现：



这里我们看到：GPT4 不仅仅完成了需求，它还做到了兼容浮点数输入，使得返回的精度更高了。例如 1.5MB 实际是 1536B ，按我们最初的实现确实会丢失精度，变成1024B。此外，还帮我们发现了个 BUG。

照例，让它补充一下单元测试代码：



通过15分钟的简单交流，我和 ChatGPT 一起完成了这次代码重构。

# 实现业务逻辑

虽然要求 chatGPT4 一次性给我们交付整个完整需求有点过分，但我们依然可以把需求拆分成小的逻辑单元给chatGPT实现，并要求它编写单元测试代码。

这次，我们找了我们项目里面最新的需求来做个实验，让 ChatGPT 帮我们完成需求——做一个事件的聚合能力。我们的可观测平台（伽利略）会收集各个平台的事件数据，聚合之后以更加可视化的方式给用户展示：









在我不断的追加需求细节之后，chatGPT4交付了一个还算可以的成果。但是这里我们发现这段代码有个BUG。当然，我是不会自己动手修复的，让GPT4自己来吧：

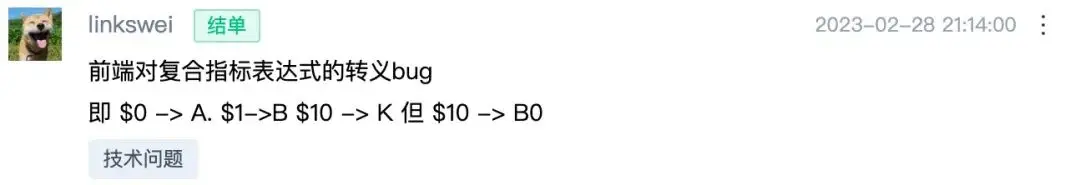


其交付的整体代码虽然不算特别清晰，但做一些修改还是可以用的。当然我觉得这跟我个人没把需求描述的太清楚也有关系。

# 改BUG

我们的可观测平台（伽利略）支持自定义指标，底层识别的变量类型是$1,$2,$3这种形式，UI为了用户可读性，是表现为A,B,C。最近出现了一个BUG，A+B+C+D+E+F+G+H+I+J+K配置下去再加载上来之后编程A+B+C+D+E+F+G+H+I+J+B0。

原因K是$11，但是前端转码的实现，是 for 循环遍历处理的，所有处理两位数字有 BUG，转回来识别成了$1和0：



我们把这个问题给ChatGPT，看它怎么解决：



此外，各位开发者通过上面的例子应该能感受到 GPT4 写单元测试代码的能力了。它不管是表驱动、测试用例的构造能力、代码的 readability 能力都非常强！此处我们不进行展开赘述了。

# 取变量名和函数名

我们知道取变量名和函数名是工程师们最头疼的问题。

这个 ChatGPT4 非常擅长，毕竟这是它的母语呀。例如上面的解析存储容量的函数，它给的建议确实比原名好太多了。



摘自https://zhuanlan.zhihu.com/p/629409280