

读了《Go To Statement Considered Harmful Edsger W. Dijkstra》，其“静态文本与动态程序”对立的形容似乎想表达 goto **实在于难以从静态文本翻译成动态程序**，我不能再认同。因为在今天，AI agent 碾过所有程序员的时代，AI 却仍然难以处理文学任务——解码太难了，真的。

某种程度上，如果你压缩了什么东西，就必然要面对解压过程所消耗的能量。（或许要解压很多次）

goto 并没有造成某些难以接受的抽象泄漏——它没有为程序带来更多，只是让某些程序变得（非常）难以阅读。

但我确实喜欢这样写：

```
for (int x = 0; x < xLen; x++) {
  for (int y = 0; y < yLen; y++) {
    for (int z = 0; z < zLen; z++) {
      if (matrix[x][y][z] == 0) {
        printf("found");
        goto done;
      }
    }
  }
}
done:
return;
```

所以，或许 Cyly 中会提供类似的语法糖，但显然我不想让它如历史上的 goto 那么狂野。

实现 scanner 时，我被编码问题困扰了一小段时间。但显然 just UTF-8 是一个极端轻松的选择。用户在何种 OS 上用哪种代码编辑器呢？这是我人生第一次思考这个问题。

Cyly 的编译器需要在 16K memory 的机器上运行吗？但这显然是个小众问题，主流关注的更多的也只是标准库的可裁剪程度。

这似乎是个方向性问题，坐大望小还是坐小望大？其实本质上我们还没做好面对核冬天与“反有机方程”的心理准备。

Cyly 将是个坐小望大的语言，它从一开始就不是完全为人类设计的。

或者，我希望在未来，一门新的编程语言仍有其意义。