前置警告，C党请离开此博文

得意之作

众所周知洛谷管理员 ltt 最近把优化禁了，导致我们这群不会卡常的小蒟蒻，就特别痛苦。

link

可以看到，ltt 在编译器层面，把优化给禁了，所以说，在代码上动手脚是不可能的啦，我们要弄，也要在编译器层面上弄。

可是编译器在洛谷上，我们又不可能通过物理黑客的手段闯入洛谷总部把编译器改了，怎么办呢？

来，仔细想想，洛谷测评机上总共有几个编译器？

GCC、FPC……

好多对吧，GCC 用不了了，FPC 还是可以用的

通过翻查资料可以得知，在代码最开头加入这三行代码可以有效加快速度。

源文档里提到 fpc 是支持 O4 的，但是不知道为什么实际上用不了。

```
{$OPTIMIZATION ON}     //解放FPC的力量！（）
{$OPTIMIZATION LEVEL3}//开启 O3 优化
{$I-}                  //开启快读
```

这样就可以愉快的绕开 ltt 的限制，继续使用 O3 啦/cy

附上资料原文：

```
1.2.58 $OPTIMIZATION : Enable Optimizations
This switch enables optimization. It can have the following possible values:
ON Switches on optimizations, corresponding to level 2 optimizations.
OFF Switches of all kinds of optimizations.
DEFAULT Returns to default (i.e. command-line or config file) specified optimizations.
XYZ Parses the string and switches on the optimizations found in the string as if they were passed
using the -Oo command line option. The optimizations must be separated by commas.
The following strings are supported:
REGVAR Use register variables.
UNCERTAIN Use uncertain optimizations.
SIZE Optimize for size.
STACKFRAME Skip stackframes.
PEEPHOLE Peephole optimizations.
ASMCSE Use common subexpression elimination at the assembler level.
LOOPUNROLL Unroll loops
TAILREC change tail recursion to regular while
ORDERFIELDS Reorder fields if this results in better alignment.
FASTMATH Fast math operations
REMOVEEMPTYPROCS remove calls to empty procedures.
CSE Use common subexpression elimination
DFA Use DFA (Data Flow Analysis).
You can disable a certain optimization by prefixing the value with NO.
Example:
{$OPTIMIZATION ON}
is equivalent to
{$OPTIMIZATION LEVEL2}
34
CHAPTER 1. COMPILER DIRECTIVES
Multiple optimizations can be specified with:
{$OPTIMIZATION REGVAR,SIZE,LEVEL2}
Dusable loop unrolling with:
{$OPTIMIZATION NOLOOPUNROLL}
This switch is also activated by the -Ooxxx command line switch. Note the small 'o': it is -Oo
```

followed by the switch name.

Idea：223588，std：279700

测试效果（为了体现效果特意把代码写的特别丑orz）：

- 无优化 2.41s
- O2 1.7s
- O3 1.62s