



実装上の注意 (Implementation Notice)

競技システムについて

- 課題の趣旨に合致するプログラムを作成し、解答提出用 Web インタフェースからソースを提出すること。解答プログラムは、入力を標準入力から読み込み、出力を標準出力に出力し、正常終了すること (0 を返すこと)。各課題ごとに、ソースを提出することができる回数は 50 回までである。
- ソースを複数回提出した場合は、提出された全てのソースが採点される。各小課題に対し、提出された全てのソースにおけるその小課題の得点の最大値が、その小課題の最終的な得点となる。各課題の得点は、その課題に含まれる小課題の最終的な得点の合計である。ただし、競技サーバに予期せぬ障害が発生した場合は、最後に提出されたソースのみを採点の対象とする場合がある。提出したソースは、各自の競技用 PC に保存しておくことを勧める。
- 採点システムで実行される際のスタックサイズに制限はない。使用されるスタックは、そのプログラムが使用するメモリの一部として扱われるので、メモリ制限を超えることはできない。再帰処理を行うときはスタックオーバーフローに注意すること。競技用 PC 上で実行する際にスタックサイズの制限を解除するには、シェル上で `ulimit -s unlimited` を実行すること。このコマンドを実行することで、そのシェルを閉じるまでは、そのシェル上で実行されるプロセスに対してスタック制限が解除される。

競技課題について

- 競技課題は難易度順に並べられている。ただし、この並びは科学委員会の主観的なものであることに注意せよ。あなたが感じる難易度は科学委員会によるものとは異なる可能性があるため、はじめに複数の課題に目を通すことを推奨する。
- 難しい課題であっても簡単な小課題があるかもしれないので、課題全体が解けると思わない場合でも、その課題について考えること。
- 競技課題では、高速な入出力が有用になることがあるかもしれない。

○ 例えば C++ では、

```
std::cin.tie(nullptr);
```

```
std::ios_base::sync_with_stdio(false);
```

と書くことで、それ以降 `std::cin` を使うときに処理が高速になる。ただし、`scanf` や `printf` などと併用すると正常に動作しないことがあるので注意すること。また、改行をするときに `std::cout << std::endl;` と書く代わりに `std::cout << '\n';` と書くと、`flush` がされな



いため処理が高速になる.

- 例えば Python では, `input()` と書く代わりに `sys` モジュールを `import` した上で `sys.stdin.readline()` と書くと, `flush` がされないため処理が高速になる. ただし, `sys.stdin.readline()` は末尾に改行文字が含まれることに注意すること.

コンパイラオプション

C++20 (g++)	<code>g++ -DEVAL -std=gnu++20 -march=native -O2 -pipe -static -s -o {file} {file}.cpp</code>
Python3 (PyPy)	<code>pypy3 {file}.py</code>

制限

問題コード	問題名	実行時間制限	メモリ制限
joig_poker	ポーカー (Poker)	2.0 sec	1024 MB
joig_snowballs	雪玉 2 (Snowballs 2)	1.0 sec	1024 MB
joig_ambulance	救急車 (Ambulance)	5.0 sec	1024 MB
joig_thief	宇宙怪盗 (Space Thief)	4.0 sec	1024 MB

ソースの連続提出制限について

各課題ごとに, 最後の提出から 1 分間は解答を提出することができない.

ただし, 競技時間の最後の 15 分間はこの制限は適用されない.