

技術情報 (Technical Info Sheet)

この資料では、C++ のストリーム (cin/cout) を使用した際に入出力が遅くなるのを避ける方法、64-bit データ型 (変数) を使用する方法、応答型課題において出力を採点システムと適切にやりとりする方法に関する情報を提供する。また、コンパイラオプションに関する情報とスタックの制限に関する情報も提供する。

C++ ストリームを使用した際の遅い入出力

大量の入出力データを扱う課題を解く際に、入出力の処理に cin, cout ストリームを使用した C++ プログラムは scanf, printf 関数を使用した同等のプログラムに比べてとても遅くなることに注意してほしい。よって、cin/cout ストリームを使用しているのであれば、代わりに scanf/printf を使用することを強く薦める。しかし、それでも cin/cout を使いたい場合は、プログラムの冒頭に次の行

```
ios::sync_with_stdio(false);
```

を加え、その上で endl を決して使用せずに代わりに “\n” を使用することを薦める。

ただし、ios::sync_with_stdio(false); を含むと cin/cout と scanf/printf を同時に使えなくなることに注意せよ。つまり、上の行を使用する場合は、cin と scanf を混在させたり、cout と printf を混在させたりしてはならない。

64-bit データ型

課題によっては、32 bit には収まらない大きな数を扱う必要がある。この場合、C/C++ では long long, Pascal では int64 といった 64 bit の整数データ型を用いなければならない。以下は、これらのデータ型の使い方を説明するためのサンプルコードである。

C/C++

```
int main(void) {  
    long long varname;  
  
    scanf("%lld", &varname);  
    // Do something with the varname variable  
    printf("%lld\n", varname);  
    return 0;  
}
```



Pascal

```
var
  varname: Int64;
begin
  read(varname);
  { Do something with the varname variable }
  writeln(varname);
end.
```

応答型課題における採点プログラムとの応答

応答型課題を解答する際、出力に新しい行を書き出すごとに出力のバッファを必ずフラッシュする必要がある。以下は、C/C++ と Pascal でどのようにフラッシュするか説明するためのコードである。

scanf/printf を用いる C/C++

```
fflush(stdout);
```

さらに、scanf を用いる場合は、「標準入力で空白を待つ間プログラムの実行を妨げる」ような入力の読み込みは避けること。このような実行の妨げは、scanf の最初の引数を空白や改行で終わると生じることがある。特に、scanf の引数として"%d" 使うべきで、"%d " (空白を最後につける) や "%d\n" (改行を最後につける) を使ってはならない。

cin/cout を用いる C++

```
cout << flush;
```

Pascal

```
flush(output);
```

コンパイルオプション

次のコマンドが、バッチ型と応答型の課題に対する解答をコンパイルするのに用いられる (課題名を abc としている):

```
C
gcc -o abc abc.c -std=gnu99 -O2 -s -static -lm -x c
```



C++

```
g++ -o abc abc.cpp -O2 -s -static -lm -x c++
```

Pascal

```
fpc -O2 -XS -Sg abc.pas
```

スタック制限

あなたのプログラムが競技システムで実行される際はいつでも、スタックサイズは課題ごとのメモリ制限で制限されるだけである。