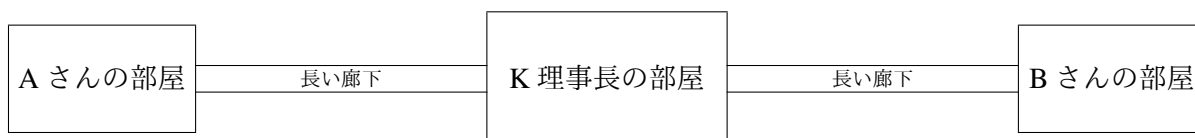




## 一緒に足し算 (Add together)

国際情報オリンピックの日本代表に選ばれた A さんと B さんは、情報処理技術を高めるため、情報オリンピック日本委員会の K 理事長に以下のような課題を提示された。

A さんと B さんは、図のように部屋に隔離されている。A さん・B さん・K 理事長の各部屋には内線電話が設置されているが、電話を発信することが可能なのは K 理事長のみである。その他の連絡手段は断たれており、A さんと B さんは指示がない限り部屋から出ることはいできない。



課題の目的は、K 理事長が A さんと B さんにいくつかの整数を伝えていったとき、A さんと B さんが直接の連絡をとらずに、「A さんに伝えられた偶数すべての和」と「B さんに伝えられた奇数すべての和」の合計を B さんが正しく求めることである。

K 理事長の部屋には赤と青の 2 枚の電光掲示板がある。これらは 0 以上 999 999 999 以下の整数を 1 個表示できる。課題の開始時、両方の電光掲示板は 0 を表示している。

これから、K 理事長が A さんまたは B さんを電話で部屋に呼ぶ、ということが行われていく。A さんまたは B さんは、K 理事長の部屋に来るたびに、整数を 1 個伝えられる。その後、A さんならば赤の電光掲示板、B さんならば青の電光掲示板に表示する整数を 1 個指定する。K 理事長は A さんまたは B さんの指定通りに電光掲示板の表示する整数を変更する。

最後に、K 理事長は B さんを電話で部屋に呼ぶ。このとき B さんは、「A さんに伝えられた偶数すべての和」と「B さんに伝えられた奇数すべての和」の合計を答える。正しく答えられれば正解であり、そうでなければ不正解である。

K 理事長が A さんまたは B さんを部屋に呼ぶタイミングは不規則であり、A さんと B さんは K 理事長がどういう順序で 2 人を部屋に呼んでいるかはわからない。

## 課題

A さんと B さんの戦略を実装し、上で説明された課題で正解できるプログラムを作成せよ。

## 実装の詳細

あなたは同じプログラミング言語で 2 つのファイルを提出しなければならない。

1 つ目のファイルは `playerA.c` または `playerA.cpp` という名前である。このファイルは A さんの戦略を実装したファイルであり、以下のルーチンを実装していなければならない。



- `void InitA(int T)`

このルーチンは、最初に 1 回だけ呼び出される。引数  $T$  は、小課題の番号  $T$  である。

- `int GameA(int Q, int X)`

このルーチンは、A さんが K 理事長の部屋に来ることに対応して呼び出される。引数  $Q$  は、A さんが K 理事長の部屋に来たときに青の電光掲示板が表示している整数  $Q$  である。  $0 \leq Q \leq 999\,999\,999$  を満たす。引数  $X$  は、A さんが K 理事長に伝えられる整数  $X$  である。  $1 \leq X \leq 100$  を満たす。

このルーチンは  $0 \leq P \leq 999\,999\,999$  を満たす整数  $P$  を返さなければならない。これは、A さんが赤い電光掲示板に表示する整数として  $P$  を指定することを表す。範囲外の値を返した場合は、不正解 [1] と判定され、プログラムの実行は終了される。

2 つ目のファイルは `playerB.c` または `playerB.cpp` という名前である。このファイルは B さんの戦略を実装したファイルであり、以下のルーチンを実装していなければならない。

- `void InitB(int T)`

このルーチンは、最初に 1 回だけ呼び出される。引数  $T$  は小課題の番号  $T$  である。

- `int GameB(int P, int X)`

このルーチンは、B さんが K 理事長の部屋に来ることに対応して呼び出される。引数  $P$  は、B さんが K 理事長の部屋に来たときに赤の電光掲示板が表示している整数  $P$  である。  $0 \leq P \leq 999\,999\,999$  を満たす。引数  $X$  は、B さんが K 理事長に伝えられる整数  $X$  である。  $1 \leq X \leq 100$  を満たす。

このルーチンは  $0 \leq Q \leq 999\,999\,999$  を満たす整数  $Q$  を返さなければならない。これは、B さんが青い電光掲示板に表示する整数として  $Q$  を指定することを表す。範囲外の値を返した場合は、不正解 [2] と判定され、プログラムの実行は終了される。

- `int AnswerB(int P)`

このルーチンは、課題の最後に B さんが K 理事長の部屋に来ることに対応して呼び出される。引数  $P$  は、B さんが K 理事長の部屋に来たときに赤の電光掲示板が表示している整数  $P$  である。  $0 \leq P \leq 999\,999\,999$  を満たす。

このルーチンは整数  $Y$  を返さなければならない。これは、B さんが「A さんに伝えられた偶数すべての和」と「B さんに伝えられた奇数すべての和」の合計として  $Y$  を答えることを表す。このとき、 $Y$  が正しい答えであれば正解、そうでなければ不正解 [3] と判定され、プログラムの実行は終了される。

ルーチン `GameA` および `GameB` は合計で高々 100 回呼び出される。

内部での使用のために他のルーチンを実装したり、グローバル変数を宣言するのは自由である。ただし、提出された 2 つのプログラムは、採点プログラムとまとめてリンクされて 1 つの実行ファイルになるので、各ファイル内のすべてのグローバル変数と内部ルーチンを `static` で宣言して、他のファイルとの干渉を避ける必要がある。採点時には、このプログラムは A さん側、B さん側として 2 個のプロセスとして実行されるので、A さん側と B さん側でプログラム中のグローバル変数を共有することはできない。

あなたの提出は標準入力・標準出力、あるいは他のファイルといかなる方法でもやりとりしてはならない。



## コンパイル・実行の方法

作成したプログラムをテストするための、採点プログラムのサンプルが、コンテストサイトからダウンロードできるアーカイブの中に含まれている。このアーカイブには、提出しなければならないファイルのサンプルも含まれている。

採点プログラムのサンプルは1つのファイルからなる。そのファイルは `grader.c` または `grader.cpp` である。作成したプログラムをテストするには、次のようにコマンドを実行する。

- C の場合

```
gcc -std=c11 -O2 -o grader grader.c playerA.c playerB.c -lm
```

- C++ の場合

```
g++ -std=c++11 -O2 -o grader grader.cpp playerA.cpp playerB.cpp
```

コンパイルが成功すれば、`grader` という実行ファイルが生成される。

実際の採点プログラムは、採点プログラムのサンプルとは異なることに注意すること。採点プログラムのサンプルは単一のプロセスとして起動する。このプログラムは、標準入力から入力を読み込み、標準出力に結果を出力する。

## 入力

採点プログラムのサンプルは標準入力から以下の入力を読み込む。

- 1行目には、整数  $T, N$  が空白を区切りとして書かれており、小課題の番号が  $T$ 、K 理事長が A さんまたは B さんに伝える整数の個数が  $N$  であることを表す。
- 続く  $N$  行のうちの  $i$  行目 ( $1 \leq i \leq N$ ) には、A または B である文字  $C_i$  と整数  $X_i$  が空白を区切りとして書かれており、 $C_i$  が A または B である場合、ルーチン `InitA` と `InitB` が呼び出された後に  $i$  番目に呼び出されるルーチンがそれぞれ `GameA` または `GameB` であること、そのとき引数  $X$  の値として  $X_i$  が与えられることを表す。

## 出力

プログラムの実行が正常に終了した場合、採点プログラムのサンプルは標準出力へ以下の情報を1行で出力する。

- 正解の場合、“Accepted” と出力される (引用符は実際には出力されない。以下同様である)。
- 不正解の場合、不正解の種類が「実装の詳細」の節に書かれた番号によって “Wrong Answer [1]” のように出力される。さらに、不正解 [3] については、ルーチン `AnswerB` が返した整数  $Y$  の値が “Wrong Answer [3] :  $Y = 4$ ” のように出力される。



## 制限

すべての入力データは以下の条件を満たす。

- $1 \leq N \leq 100$ .
- $1 \leq X_i \leq 100$  ( $1 \leq i \leq N$ ).

## 小課題

### 小課題 1 [30 点]

- $T = 1$  を満たす。
- $N = 2$  であり、 $C_1$  は文字 B,  $C_2$  は文字 A である。すなわち、ルーチン `InitA` および `InitB` が呼び出された後に、`GameB`, `GameA`, `AnswerB` という順でルーチンが呼び出される。

### 小課題 2 [70 点]

- $T = 2$  を満たす。

## やりとりの例

採点プログラムのサンプルが読み込む入力の例と、それに対応するルーチンの呼び出しの例を以下に示す。

入力
2 6
A 6
B 5
B 4
A 3
A 2
B 1



A さん側		B さん側	
呼び出し	戻り値	呼び出し	戻り値
InitA(2)	なし		
		InitB(2)	なし
GameA(0, 6)	6		
		GameB(6, 5)	5
		GameB(6, 4)	60
GameA(60, 3)	53		
GameA(60, 2)	530		
		GameB(530, 1)	601
		AnswerB(530)	14

この例では、「A さんに伝えられた偶数すべての和」は  $6 + 2 = 8$  であり、「B さんに伝えられた奇数すべての和」は  $5 + 1 = 6$  であるから、 $8 + 6 = 14$  が正解となる。



## ESP 数あて (Extrasensory Guessing)

JOI 君は、Just Odd Inventions 社から販売されている超能力開発キットを使い、超能力の修行に励んでいる。

このキットの一つに、「ESP 数あて」がある。これは、少ない数の質問と超能力を用いて、秘密の整数を当てるゲームである。

JOI 君に代わって、秘密の整数を当てるプログラムを作成せよ。

### 課題

少ない数の質問と超能力を用いて、秘密の整数を当てるプログラムを作成せよ。

### 実装の詳細

あなたは、以下のルーチンを実装した、1つのファイルを提出しなければならない。

- `void Game(int N, int Q)`

このルーチンは、最初に 1 回だけ呼び出される。

- 引数  $N$  は、秘密の整数が 1 以上  $N$  以下であることを表す整数である。
- 引数  $Q$  は、質問してよい回数である。

`Game` の中では、以下のルーチン呼び出さなければならない。

- `int Guess(int X)`

- 引数  $X$  は、秘密の整数の推定値である。

ただし、`Guess` の呼び出しは以下を満たすこと。

- $X$  は 1 以上  $N$  以下の整数でなければならない。これを満たさない場合、不正解 [1] となる。
- `Guess` は高々  $Q$  回しか呼び出してはならない。これを満たさない場合、不正解 [2] となる。

`Guess` の呼び出しが上記の条件を満たさなかった場合、その時点でプログラムの実行は終了される。

`Guess` の呼び出しが上記の条件を満たした場合、 $X$  と秘密の整数の大小関係に応じて、`Guess` は以下のような値を返す。

- $X$  が秘密の整数と等しいとき、正解となり、プログラムは終了する。`Guess` の関数呼び出しから戻ることはない。
- $X$  が秘密の整数より小さいとき、`Guess` は  $-1$  を返す。



- $X$  が秘密の整数より大きいとき, `Guess` は 1 を返す.

秘密の整数を引数にして `Guess` を呼び出すことなく `Game` が終了した場合, 不正解 [3] となる.

内部での使用のために他のルーチンを実装したり, グローバル変数を宣言するのは自由である. ただし, あなたの提出は標準入力・標準出力, あるいは他のファイルといかなる方法でもやりとりしてはならない.

## コンパイル・実行の方法

作成したプログラムをテストするための, 採点プログラムのサンプルが, コンテストサイトからダウンロードできるアーカイブの中に含まれている. このアーカイブには, 提出しなければならないファイルのサンプルも含まれている.

採点プログラムのサンプルは 1 つのファイルからなる. そのファイルは `grader.c` または `grader.cpp` である. 作成したプログラムをテストするには, 次のようにコマンドを実行する.

- C の場合

```
gcc -std=c11 -O2 -o grader grader.c extrasensory.c -lm
```

- C++ の場合

```
g++ -std=c++11 -O2 -o grader grader.cpp extrasensory.cpp
```

コンパイルが成功すれば, `grader` という実行ファイルが生成される.

実際の採点プログラムは, 採点プログラムのサンプルとは異なることに注意すること. 採点プログラムのサンプルは単一のプロセスとして起動する. このプログラムは, 標準入力から入力を読み込み, 標準出力に結果を出力する.

## 採点プログラムのサンプルの入力

採点プログラムのサンプルは標準入力から以下の入力を読み込む.

- 1 行目には, 整数  $N, Q, S$  が空白を区切りとして書かれており, 秘密の整数の最大値として JOI 君に伝えられる数が  $N$ , JOI 君が質問してよい回数が  $Q$ , 実際の秘密の整数の値が  $S$  であることを表す.

## 採点プログラムのサンプルの出力

プログラムの実行が正常に終了した場合, 採点プログラムのサンプルは標準出力へ以下の情報を 1 行で出力する (引用符は実際には出力されない).

- 正解の場合, `Guess` の呼び出された回数が “Accepted : G = 1” のように出力される.
- 不正解の場合, 不正解の種類が “Wrong Answer [1]” のように出力される.



## 制限

すべての入力データは以下の条件を満たす。

- $N = 100$ .
- $Q = 1$ .

## 小課題

### 小課題 1 [100 点]

追加の制約はない。

## やりとりの例

採点プログラムのサンプルが読み込む入力の例と、それに対応するルーチンの呼び出しの例を以下に示す。

入力例	ルーチンの呼び出しの例	
100 1 99	呼び出し	戻り値
	Guess(50)	-1

この例では、秘密の整数を引数にして **Guess** を呼び出すことなく **Game** が終了したため、不正解 [3] となる。

入力例	ルーチンの呼び出しの例	
100 1 99	呼び出し	戻り値
	Guess(99)	プログラムは終了する

この例では、秘密の整数を引数にして **Guess** を呼び出したため、**Guess** の関数呼び出しから戻ることなく正解となる。





## ヒット (Hit)

### 出力のみの課題 (Output Only Task)

情報オリンピック日本委員会の倉庫には、秘密文書が保管された金庫が眠っている。金庫の鍵は 10 個の秘密の整数  $N_1, N_2, \dots, N_{10}$  であり、K 理事長だけが知っている。金庫を開けるには、これらの整数を揃える必要がある。  $N_1, N_2, \dots, N_{10}$  はそれぞれ 0 以上  $10^{10}$  未満の整数である。

IOI 2018 の日本開催に向けて情報オリンピック日本委員会の歴史を調査しているあなたは、K 理事長に次のような質問を何回か行うことで金庫の鍵である秘密の整数  $N_i$  ( $1 \leq i \leq 10$ ) を特定することにした。

- $M$  を 0 以上  $10^{10}$  未満の整数とし、K 理事長に「 $N_i$  と  $M$  を 10 桁の十進法で書いたとき、値が一致している桁の個数はいくつですか」と質問する。

質問に対して K 理事長は正しく答える。K 理事長に質問を行うことにより、秘密の整数  $N_1, N_2, \dots, N_{10}$  を決定せよ。

ただし、 $10^9$  より小さい整数を 10 桁の十進法で書くときは、先頭にいくつかの '0' を補うものとする。例えば、 $N_i = 123405678$  のときに  $M = 444888$  として K 理事長に質問した場合は、 $N_i$  を 10 桁の十進法で書くとき文字列 `0123405678` となり、 $M$  を 10 桁の十進法で書くとき文字列 `0000444888` となる。1 文字目の '0'、5 文字目の '4'、10 文字目の '8' の 3 箇所が一致するから、この場合の K 理事長の答えは 3 である。

## 課題

秘密の整数  $N_1, N_2, \dots, N_{10}$  に対するインターフェースが C/C++ の関数ライブラリの形で提供される。また、テストのために自分で秘密の整数を定義する方法も提供される。与えられたインターフェースを用いて秘密の整数を可能な限り決定し、各々を記述したファイルを提出せよ。

## 実装の詳細

秘密の整数に関する質問をするためのライブラリが、コンテストサイトからダウンロードできるアーカイブの中に含まれている。このライブラリをリンクすることで、以下の関数を呼び出すことができる。

- `void Initialize(int T);`

ライブラリを初期化する。この関数は、プログラム開始時にちょうど 1 回呼び出なければならない。引数  $T$  は、 $0 \leq T \leq 10$  をみたす整数  $T$  である。  $0 < T$  の場合、プログラムにおいて秘密の整数  $N_T$  に関する質問を行うことを意味する。  $T = 0$  の場合、カレントディレクトリのファイル `hit.in` から整数を読み込み、それを  $N_0$  とおき、これに関する質問を行うことを意味する。

- `int Hit(long long M);`

秘密の整数  $N_T$  に関する質問をする。戻り値は、 $N_T$  と  $M$  を 10 桁の十進法で書いたときに値が一致している桁の個数である。



- `void Finalize();`

やりとりを安全に終了する。プログラムの終了時に、この関数を呼ぶべきである。

ライブラリに `main` 関数は含まれていない。あなたのプログラムは、通常通り `main` 関数を実装する必要がある。

## コンパイル・実行の方法

ライブラリを使ったプログラムをコンパイルするには、`hitlib.h` と `hitlib.o` が、作成したプログラムと同じディレクトリにある必要がある。

例えば、作成したプログラムを `Hit.c` または `Hit.cpp` とするとき、作成したプログラムをテストするには、次のようにコマンドを実行する。

- C の場合

```
gcc -std=c11 -O2 -o hit hitlib.o Hit.c -lm
```

- C++ の場合

```
g++ -std=c++11 -O2 -o hit hitlib.o Hit.cpp
```

## 実験

関数 `Initialize` に整数 `0` を渡すと、ライブラリは秘密の整数の情報をファイル `hit.in` から読み込む。これによりライブラリを用いて実験をすることができる。ファイル `hit.in` の形式を次に示す。

<code>hit.in</code>	説明
<code>0123456789</code>	1 行目には秘密の整数 $N$ を書く。

## エラーメッセージ

例外的な場合には、ライブラリは標準エラー出力にエラーメッセージを出力する。起きうるエラーメッセージとその意味は次の通りである。

エラーメッセージ	意味
<code>ERR 1 Invalid T.</code>	$T$ の値が正しくない
<code>ERR 2 Invalid M.</code>	$M$ の値が正しくない
<code>ERR 3 Initialization called twice.</code>	関数 <code>Initialize</code> が 2 回呼ばれた
<code>ERR 4 Hit called before Initialization.</code>	関数 <code>Hit</code> の前に関数 <code>Initialize</code> が呼ばれた
<code>ERR 5 File error.</code>	ファイルエラー
<code>ERR 6 Network error.</code>	ネットワークエラー



## 制限

秘密の整数  $N_1, N_2, \dots, N_{10}$  は 0 以上  $10^{10}$  未満である。

## 出力

特定した秘密の整数をあらわす長さ 10 の文字列を 1 行で出力せよ。文字列は文字 ‘0’, ..., ‘9’ または ‘?’ からなる。‘?’ は、その桁を特定できなかったことを表す。秘密の整数  $N_1, N_2, \dots, N_{10}$  に関する出力を、それぞれファイル `output_01.txt`, `output_02.txt`, ..., `output_10.txt` として出力せよ。

## 採点について

秘密の整数  $N_i$  ( $1 \leq i \leq 10$ ) に対し、次のように採点を行う。

- あなたの出力が課題の要請を満たしていない場合、その出力に対するあなたの得点は 0 点となる。
- あなたの出力において ‘0’, ..., ‘9’ が間違った場所に書かれている場合、その出力に対するあなたの得点は 0 点となる。
- そうでない場合、あなたの出力において ‘0’, ..., ‘9’ が書かれている個数を、その出力に対するあなたの得点とする。

## やりとりの例

採点プログラムのサンプルが読み込む入力の例と、それに対応する関数とルーチンの呼び出しの例を以下に示す。

hit.in の例	呼び出しの例	
	呼び出し	戻り値
0123456789	Initialize(0)	
	Hit(9290741307LL)	0
	Hit(7583348375LL)	1
	Hit(4145466449LL)	4
	Hit(99)	2
	Finalize()	

ただし、9290741307LL は、C/C++ における long long 型の整数 9290741307 を表す。7583348375LL や 4145466449LL も同様である。



## 電飾 (Illumination)

JOI 高校の文化祭では毎年廊下に電飾が飾られる。電飾は  $N$  個の電球で構成されており、電球は廊下の西側から東側に一列に並んでいる。各電球は明かりがついているか、ついていないかのいずれかの状態である。

JOI 高校の倉庫には電球を操作する機械が眠っている。この機械は電飾内で連続した電球を指定すると、指定された電球のうち、明かりがついている電球全てを明かりがついていない状態にし、明かりがついていない電球全てを明かりがついている状態にする。ただし、機械は老朽化のため、1 回しか使用できない。

JOI 高校の生徒達は明かりがついている電球とついていない電球が交互に並んだ列（このような電球の列を交互列と呼ぶ）が好きである。そこで、この機械を必要ならば 1 回だけ使って、できるだけ長い交互列を含む電飾を作ることにした。

### 例

例えば、電飾の配置が西から東に向かって

○ ○ ● ● ○ ● ○ ○ ○ ●

となっていたとする（○は明かりがついている電球を、●は明かりがついていない電球を表す）。このとき、4 番目から 7 番目までの 4 個の電球に対して機械を操作すると、

○ ○ ● ○ ● ○ ● ○ ○ ●

となり、2 番目から 8 番目までの電球が長さ 7 の交互列をなす。

○ ○ ● ○ ● ○ ● ○ ○ ●

また、8 番目の電球のみに対して機械を操作すると、

○ ○ ● ● ○ ● ○ ● ○ ●

となり、4 番目から 10 番目までの電球が長さ 7 の交互列をなす。

○ ○ ● ● ○ ● ○ ● ○ ●

機械を最大 1 回使用することで、長さが 8 以上の交互列を作ることはできない。

### 課題

電飾の情報が与えられたとき、機械を最大 1 回使用することで得られる電球の配列に含まれる交互列の長さとして考えられるものの最大値を求めるプログラムを作成せよ。



## 入力

標準入力から以下の入力を読み込め。

- 1 行目には、整数  $N$  が書かれている。
- 2 行目には、 $N$  個の 0 または 1 が空白を区切りとして書かれている。各整数は機械を操作する前における電球の情報を表している。左から  $i$  番目 ( $1 \leq i \leq N$ ) の整数は西側から  $i$  番目の電球の情報を表す。整数が 1 ならば電球の明かりがついていることを、0 ならば明かりがついていないことを表す。

## 出力

標準出力に、作成可能な電球の列に含まれる交互列の長さの最大値を表す整数を 1 行で出力せよ。

## 制限

すべての入力データは以下の条件を満たす。

- $2 \leq N \leq 100\,000$ 。

## 小課題

### 小課題 1 [10 点]

- $N \leq 50$  を満たす。

### 小課題 2 [10 点]

- $N \leq 500$  を満たす。

### 小課題 3 [20 点]

- $N \leq 2\,000$  を満たす。

### 小課題 4 [60 点]

追加の制限はない。



## 入出力例

入力例 1	出力例 1
10 1 1 0 0 1 0 1 1 1 0	7

これは問題文中で説明された例である。

入力例 2	出力例 2
10 1 0 0 0 0 1 0 1 0 1	8

西側から 4 番目の電球のみを操作すると、最大値 8 を満たす交互列が得られる。

入力例 3	出力例 3
5 1 1 0 1 1	5

西側から数えて 2 番目から 4 番目までの電球を操作すると、全ての電球からなる交互列を作ることができる。

入力例 4	出力例 4
3 0 1 0	3

機械を使用しなくても良い場合があることに注意せよ。