PLANETA OPIC (140 bodů)

Šimpanzi se rozhodli, že si vytvoří svůj programovací jazyk. Slovní zásoba šimpanze je ale omezená, průměrný šimpanz totiž umí vydat pouze tři zvuky: `Oo!` `Oo?` `Oo.` Tyto tři zvuky ovšem šimpanzům stačily k tomu, aby vytvořili kompletní programovací jazyk.

Jak jazyk funguje?

Jazyk obsahuje celkem 8 příkazů. Každý příkaz je kombinace 2 zvuků (1 příkaz = 2 zvuky). Celý program (posloupnost příkazů) je uložen v souboru. Cesta k tomuto souboru je zadaná jako první parametr příkazové řádky.
Dále program potřebuje nějakou pracovní paměť - pracuje tedy nad polem buněk. Počet buněk v poli je zadán jako druhý parametr příkazové řádky. Pokud parametr chybí, implicitní velikost je 30 000 buněk. Každá buňka má velikost Char (2 byty). Na začátku jsou všechny buňky prázdné (tzn. hodnota v každé buňce je nula).

Příkazy

`Oo. Oo?`	posun datového ukazatele o jednu buňku doprava
`Oo? Oo.`	posun datového ukazatele o jednu buňku doleva
`Oo. Oo.`	zvýšení hodnoty aktivní buňky o 1 (buňky, nad kterou je ukazatel), operace *tiše* přetéká (tzn. Pokud v buňce max hodnota a my k ní přičteme 1, tak se hodnota v buňce změní na min. Nedojde k vypsání žádné hlášky).
`Oo! Oo!`	snížení hodnoty aktivní buňky o 1, operace *tiše* podtéká (tzn. Pokud v buňce min hodnota a my od ní odečteme 1, tak se hodnota v buňce změní na max. Nedojde k vypsání žádné hlášky)
`Oo! Oo.`	výpis hodnoty aktivní buňky na standardní výstup Pro výpis se používá hodnota aktivní buňky převedena dle kódování Unicode na znak.
`Oo. Oo!`	uložení hodnoty ze vstupu do aktivní buňky (původní hodnota v buňce je přepsána)
`Oo! Oo?`	pokud je hodnota aktivní buňky rovna nule, provede přesun instrukčního ukazatele doprava **za** odpovídající `Oo? Oo!` . Jedná se tedy o ekvivalent konce cyklu

`Oo? Oo!`	pokud je hodnota aktivní buňky různá od nuly, provede přesun instrukčního ukazatele doleva **před** odpovídající `Oo! Oo?`. Jedná se tedy o ekvivalent začátku cyklu
`Oo? Oo?`	Dejte šimpanzovi za odměnu banán (tzn. tento příkaz nedělá nic :D)

Vašim úkol je vytvořit interpreter pro programovací jazyk šimpanzů (tzn. vytvořte program, který vykoná příkazy podle šimpanzího jazyka).

Tento program vypíše ***Hello world!***:

```
Oo. Oo? Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo! Oo? Oo? Oo. Oo. Oo.
Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo? Oo! Oo! Oo? Oo! Oo? Oo. Oo! Oo. Oo. Oo?
Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo! Oo? Oo? Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo.
Oo. Oo? Oo! Oo! Oo? Oo! Oo? Oo. Oo. Oo. Oo! Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo.
Oo! Oo. Oo! Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo! Oo. Oo. Oo? Oo. Oo? Oo. Oo? Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo.
Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo! Oo? Oo? Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo? Oo! Oo! Oo? Oo!
Oo? Oo. Oo! Oo. Oo. Oo? Oo. Oo? Oo. Oo? Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo.
Oo. Oo. Oo. Oo. Oo! Oo? Oo? Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo.
Oo. Oo? Oo! Oo! Oo? Oo! Oo? Oo. Oo! Oo! Oo! Oo! Oo! Oo! Oo. Oo? Oo. Oo? Oo. Oo? Oo. Oo? Oo. Oo!
Oo. Oo. Oo. Oo. Oo. Oo. Oo. Oo! Oo. Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo. Oo! Oo! Oo!
Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo! Oo. Oo. Oo? Oo. Oo? Oo. Oo. Oo! Oo.
```

Tento program sečte dvě čísla na vstupu a vypíše výsledek:

```
Oo. Oo! Oo. Oo! Oo. Oo! Oo. Oo! Oo. Oo! Oo. Oo! Oo? Oo! Oo? Oo. Oo! Oo.Oo? Oo! Oo. Oo. Oo! Oo! Oo. Oo!
Oo? Oo! Oo! Oo!
```

Úkoly k vyřešení jsou následující:

1. Váš program správně interpretuje prvních 6 příkazů
2. Cokoli, co není příkaz v jazyce šimpanzů, bude váš program ignorovat
3. Při posunu datového ukazatele mimo pracovní paměť se vypíše chyba ***Memory overflow*** nebo ***Memory underflow***.
4. Váš program správně interpretuje příkazy `Oo! Oo?` a `Oo? Oo!`
5. V případě chyb (neuzavřené/neotevřené cykly, tj. nepárové příkazy `Oo! Oo?` a `Oo! Oo?`) se program ukončí hláškou ***Unclosed cycle*** popř. ***Unopened cycle***

GENETICKÉ MUTACE (55 bodů)

Zjistili jste, že programovat už umíte dokonale a už to pro vás není žádná výzva, proto jste se rozhodli, že se naučíte něco nového. Po chvíli hledání jste narazili na obor Bioinformatika (kombinace biologie a informatiky). Začali jste studovat, co taková bioinformatika zahrnuje a narazili jste na následující úlohu. Vaším úkolem je, tuto úlohu vyřešit:

Definujeme gen jako slovo složené z 8 písmen, kde písmena mohou být pouze “A”, ”C”, “G” nebo ”T”. Příklad genu je například sekvence písmen “ACCGATGC”. Máme zadaný počáteční gen a chtěli bychom spočítat a vypsat **nejmenší počet mutací**, abychom dostali výsledný gen. Mutace je změna jednoho písmene v genu (na jedno ze čtyř možných písmen). Ne všechny geny je samozřejmě možné vytvořit. Geny, které lze vytvořit, jsou uloženy v souboru (na každém řádku jeden gen). Cesta k tomuto souboru je zadaná jako první parametr příkazové řádky. Jestliže Cíl není možné vytvořit, program vypíše -1.

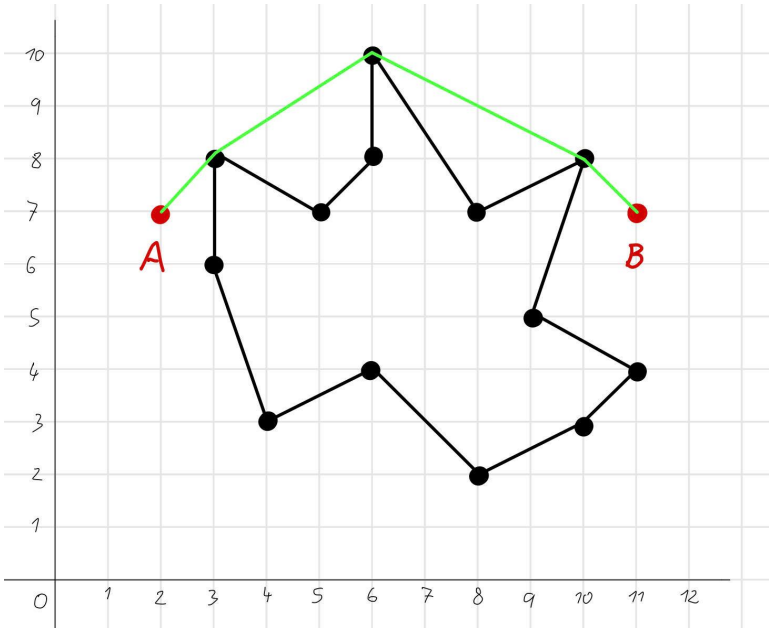
Příklad:

seznam možných genů:	[“AAACGGTT”, “AAACGGTA”, “ATACGGTA”, “AACCGGTT”]
počáteční gen:	“AACCGGTT”
cílový gen:	“AAACGGTA”
řešení:	“AACCGGTT” → “AAACGGTT” → “AAACGGTA”

Program by tedy vypsal tyto tři geny (na každý řádek jeden gen) na příkazovou řádku. Pokud nevíte jak začít, můžete na začátku předpokládat, že lze vytvořit libovolný gen. Za takové řešení dostanete část bodů

VESMÍRNÁ ROBOTIKA (105 bodů)

Na marsu přistálo nové robotické vozítko. Vozítko má zkoumat povrch Marsu. Ukázalo se ale, že zásoby paliva na Marsu jsou dosti omezené, proto je potřeba, aby se robot pohyboval co nejúsporněji.
Váš úkol je proto následující: Robot se potřebuje dostat z místa A do místa B po nejkratší možné trase. Mezi místem A a B je překážka ve tvaru mnohoúhelníku. Níže vidíte konkrétní situaci - černě je znázorněn mnohoúhelník (se zvýrazněnými vrcholy), červeně je znázorněna startovní a cílová pozice robota, zeleně je znázorněna nejkratší trasa okolo překážky.



Vstup programu je na konzoli a má následující formát:

1. na prvním řádku jsou dvě čísla oddělená mezerou, která udávají souřadnice bodu A
2. na druhém řádku jsou dvě čísla oddělená mezerou, která udávají souřadnice bodu B
3. na třetím řádku je jedno číslo X, které udává počet vrcholů mnohoúhelníku
4. na dalších X řádcích jsou souřadnice jednotlivých vrcholů mnohoúhelníku
   Poznámka: všimněte si, že záleží na pořadí, ve kterém jsou body mnohoúhelníku zadané. V případě jiného pořadí bodů by se jednalo o úplně jiný mnohoúhelník

pro situaci na obrázku výše by byl vstup následující:

2 7
11 7
13
4 3
6 4
8 2
10 3
11 4
9 5
10 8
8 7
6 10
6 8
5 7
3 8
3 6

Řešením je vždy nějaká lomená čára, kterou můžeme popsat jako posloupnost bodů. Souřadnice těchto bodů (nejkratší cesty) vypište do konzole. Začátek je v bodě A, konec v bodě B.
Řešení vypsané do konzole pro situaci na obrázku výše by bylo následující:
2 7
3 8
6 10
10 8
11 7

Pokud nevíte jak začít, tak si zadání zjednodušte tak, že výsledek nemusí být ***nejkratší*** cesta (ovšem cesta by stále neměla procházet skrz mnohoúhelník). V případě, že vaše řešení bude “dostatečně krátká” cesta, můžete stále obdržet velké množství bodů z úlohy.