# Card Trick

Δύο παίκτες πρόκειται να επιδείξουν ένα κόλπο με μια τυπική τράπουλα 52 φύλλων. Για ευκολία, οι τιμές των κάρτων θα είναι διαφορετικοί ακέραιοι αριθμοί από 0 έως 51.
Οι κάρτες τοποθετούνται αρχικά σε ένα τραπέζι με μία σειρά προς τα πάνω (με ορατές τιμές) με κάποια σειρά άγνωστη στους παίκτες.
Ο πρώτος παίκτης πηγαίνει στο τραπέζι, κοιτάζει τα φύλλα και κάνει ανταλλαγές, το πολύ $S$ φορές συνολικά. Κάθε ανταλλαγή πραγματοποιείται επιλέγοντας δύο φύλλα στις θέσεις i και j (τα i και j μπορούν να είναι ίδια) και μετακινώντας την κάρτα από τη θέση i στη θέση j και το αντίστροφο.
Μετά από αυτό, ο πρώτος παίκτης φεύγει χωρίς να επικοινωνήσει με τον δεύτερο παίκτη και όλα τα φύλλα έχουν γυριστεί ανάποδα (οι τιμές τους δεν είναι πλέον ορατές) χωρίς να έχει αλλάξει η σειρά τους. Ο δεύτερος παίκτης προσκαλείται στο τραπέζι και του ζητείται να μαντέψει πού είναι η κάρτα με την τιμή-στόχο (**target**) και επιτρέπεται να αναποδογυρίσει το πολύ T κάρτες μια προς μια. Εάν κάποια από τις κάρτες που έχουν αποκαλυφθεί είναι ο στόχος/**target**, τότε οι παίκτες κερδίζουν. Εάν τελειώσουν οι μαντεψιές/ guesses, χάνουν.
Ο στόχος σας είναι να γράψετε δύο προγράμματα που θα προσομοιώσουν τις ενέργειες των παικτών και θα κερδίσουν το παιχνίδι.

## Λεπτομέρειες υλοποίησης

Θα σας δοθούν δύο προγράμματα - firstPlayer και secondPlayer μαζί με ένα δείγμα βαθμολογητή.

Στη firstPlayer πρέπει να υλοποιήσετε την ακόλουθη συνάρτηση:

```
void swapCards(int cards[], int S, int T)
```

- Αυτή η συνάρτηση καλείται ακριβώς μία φορά από τον βαθμολογητή
- cards: ένας πίνακας που περιέχει αρχικές τιμές κάρτας από αριστερά προς τα δεξιά, με ακριβώς 52 στοιχεία με κατάταξη/δείκτες από 0 έως 51
- S: ο αριθμός των επιτρεπόμενων ανταλλαγών/swaps
- T: ο αριθμός των επιτρεπόμενων μαντεψιών/guesses

Η `swapCards` μπορεί να κάνει κλήσεις στην ακόλουθη συνάρτηση:

```
void doSwap(int i, int j)
```

- i: ο δείκτης της πρώτης κάρτας για ανταλλαγή, $0 \leq i < 52$
- j: ο δείκτης της δεύτερης κάρτας για ανταλλαγή, $0 \leq j < 52$

- Η `doSwap` μπορεί να κληθεί το πολύ S φορές

Στη SecondPlayer πρέπει να υλοποιήσετε την ακόλουθη συνάρτηση:

```
void guessCard(int S, int T, int target)
```

- S: ο αριθμός των επιτρεπόμενων ανταλλαγών/swaps
- T: ο αριθμός των επιτρεπόμενων μαντεψιών/guesses
- target: η αξία της κάρτας που πρέπει να αποκαλυφθεί

Η `guessCard` μπορεί να κάνει κλήσεις στην ακόλουθη συνάρτηση:

```
int guess(int idx)
```

- idx: guessed index/δείκτης μαντεψιών, $0 \leq idx < 52$
- Επιστρέφει την τιμή του idx-th κάρτας
- `guess` μπορεί να κληθεί το πολύ T φορές.
- Όταν μαντέψετε σωστά η αξιολόγηση τερματίζεται με επιτυχία

## Παράδειγμα αλληλεπίδρασης

Ακολουθεί ένα παράδειγμα εισόδου για τον βαθμολογητή.
Η πρώτη γραμμή πρέπει να περιέχει δύο ακέραιους αριθμούς: S and T.
Η δεύτερη γραμμή πρέπει να περιέχει 52 αριθμούς. i-th ένα που δηλώνει την τιμή της i-th κάρτας.
Η τρίτη γραμμή περιέχει έναν ακέραιο **target**.

| Παράδειγμα εισόδου στον βαθμολογητή | Δείγμα κλήσεων | | |
|---|---|---|---|
| | Κλήσεις | Υπο-κλήσεις | Επιστρέφει |
| 1 51<br>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51<br>1 | swapCards([0,1,…], 1, 51) | | |
| | | doSwap(0, 1) | |
| | | | swaps cards με δείκτες 0 και 1 |
| | swapCards finishes | | . |
| | guessCard(1, 51, 1) | | |
| | | guess(5) | |
| | | | Η guess επιστρέφει 5 |
| | | guess(1) | |
| | | | Η guess επιστρέφει 0 |
| | | guess(0) | |
| | | | Σωστό αποτέλεσμα! |

## Περιορισμοί

- $1 \leq S \leq 52$
- $1 \leq T \leq 51$
- $0 \leq target < 52$

## Υποπροβλήματα

1. (16 βαθμοί): $\boldsymbol{S} = 52, \boldsymbol{T} = 1$
2. (20 βαθμοί): $\boldsymbol{S} + \boldsymbol{T} = 52$
3. (22 βαθμοί): $\boldsymbol{S} = 13, \boldsymbol{T} = 27$
4. (18 βαθμοί): $\boldsymbol{S} = 1, \boldsymbol{T} = 26$
5. (24 βαθμοί): Η στρατηγική νίκης υπάρχει για τα δεδομένα $\boldsymbol{S}$ και $\boldsymbol{T}$