Трик со карти

Двајца играчи ќе демонстрираат трик со карти со стандарден шпил со 52 карти. За олеснување, вредностите на картите ќе бидат меѓусебно различни цели броеви од 0 до 51.
Картите првично се стават на маса во еден ред свртени нагоре (со видливи вредности) по некој редослед кој е непознат за играчите.
Првиот играч оди на масата, ги гледа картите и прави замени (swaps), вкупно најмногу S пати. Секоја замена се изведува со избор на две карти на позициите i и j (i и j може да бидат еднакви) и преместување на картата од позиција i во позиција j и обратно.
После тоа, првиот играч заминува без да комуницира со вториот играч и сите карти се превртуваат (нивните вредности веќе не се видливи) без да се промени нивниот редослед. Вториот играч е поканет на масата и е замолен да погоди каде е картата со некоја **целна (target)** вредност и му е дозволено да преврти најмногу T карти една по една. Ако која било од откриените карти е **целната** карта **(target)**, тогаш играчите победуваат. Ако ги истрошат сите T шанси за погодување, тие губат.
Вашата цел е да напишете две програми што ќе ги симулираат постапките на играчите и ќе победат во играта.

Имплементациски детали

Ќе ви бидат дадени две програми - FirstPlayer и SecondPlayer заедно со оценувач.

Во FirstPlayer треба да ја имплементирате следната функција:

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

- Функцијата се повикува точно еднаш од оценувачот
- cards: низа која ги содржи почетните вредности на картите од лево на десно, со точно 52 елементи со индекси од 0 до 51
- S: бројот на дозволени замени
- T: бројот на дозволени погодувања

`swapCards` може да ја повикува функцијата:

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

- i: индекс на првата карта која се заменува, $0 \leq i < 52$
- j: индекс на втората карта која се заменува, $0 \leq j < 52$
- `doSwap` може да се повика најмногу S пати

Во SecondPlayer треба да ја имплементирате следната функција::

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

- S: бројот на дозволени замени
- T: бројот на дозволени погодувања
- target: вредноста на картата која треба да се најде (отвори)

`guessCard` може да ја повикува функцијата:

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

- idx: погодениот индекс, $0 \leq idx < 52$
- Ја враќа вредноста на idx-тата карта
- `guess` може да се повика најмногу T пати.
- Во моментот кога ќе се случи точното погодување евалуацијата завршува со успех

Пример интеракција

Подолу е пример за влез за прикачениот оценувач (grader).
Првиот ред ги содржи: S и T.
Вториот ред има 52 броја. i-тиот ја претставува вредноста на i-тата карта.
Во третиот ред е **target** вредноста.

Sample Input to grader	Sample Calls		
	Calls	sub-calls	Returns
1 51 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 1	swapCards([0,1,…], 1, 51)		
		doSwap(0, 1)	
			swaps cards with indexes 0 and 1
	swapCards finishes		.
	guessCard(1, 51, 1)		
		guess(5)	
			guess returns 5
		guess(1)	
			guess returns 0
		guess(0)	
			Correct!

Ограничувања

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

Subtasks

1. (16 поени): $\boldsymbol{S} = 52, \boldsymbol{T} = 1$
2. (20 поени): $\boldsymbol{S} + \boldsymbol{T} = 52$
3. (22 поени): $\boldsymbol{S} = 13, \boldsymbol{T} = 27$
4. (18 поени): $\boldsymbol{S} = 1, \boldsymbol{T} = 26$
5. (24 поени): Постои победничка стратегија за дадените $\boldsymbol{S}$ и $\boldsymbol{T}$