# ლიფტი

დროის ლიმიტი: 10 წამი
მეხსიერების ლიმიტი: 64 MiB (პროცესზე)

*ეს არის კომუნიკაციის ამოცანა.*

მოცემულია შენობა სართულებით, რომლებიც გადანომრილია 0-დან $N$-მდე. სართული $0$ არის ქვედა სართული, ხოლო $N$ არის ბოლო სართული, რომლის ზევითაც კიდევ არის სახურავი. ამრიგად, სახურავის ჩათვლით, შენობას აქვს $N+2$ დონე.

თითოეულ სართულზე $f$ ($0 \le f \le N$) დასახლებული არის ზუსტად ერთი მობინადრე და თითოეულ მათგანს აქვს საიდუმლო მთელი რიცხვი $v_f$ ($0 \le v_f \le 3$), რომელიც მხოლოდ მისთვის არის ცნობილი. ასევე არის კარლსონი, რომელიც ცხოვრობს ამ შენობის სახურავზე. კარლსონის მიზანი არის, რომ განსაზღვროს რაც შეიძლება მეტი მნიშვნელობა $v_0, v_1, \ldots, v_N$-დან, ხოლო თითოეული მობინადრე **თანამშრომლობს** კარლსონის მიერ აღდგენილი საიდუმლო მნიშვნელობების რაოდენობის მაქსიმალურად გაზრდის მიზნით.

კომუნიკაციისთვის მობინადრეები იყენებენ შენობის ლიფტს შემდეგი წესით.

ლიფტი იწყებს $0$ სართულიდან და მოძრაობს მხოლოდ ზემოთ. ლიფტში არის ღილაკები, რომლებზე აღნიშნულია სართულები 1-დან $N$-მდე. თავდაპირველად არცერთი ღილაკი არ არის დაჭერილი. ღილაკზე დაჭერის შემთხვევაში ის რჩება დაჭერილი სამუდამოდ. ლიფტი გაჩერდება და გაიღება $f$ სართულზე, თუ ან $f = 0$, ან $f$ სართულის შესაბამის ღილაკს უფრო ქვედა გაჩერებაზე დააჭირეს ხელი. იმ ყველაზე მაღალი სართულის მონახულების შემდეგ, რომლის ღილაკიც დაჭერილია (ან $0$ სართულის, თუ არცერთი ღილაკი არასდროს არ იქნა დაჭერილი), ლიფტი პირდაპირ სახურავზე აგრძელებს გზას დარჩენილ არცერთ სართულზე გაჩერების გარეშე.

როცა ლიფტი ჩერდება სართულ $f$-ზე, ხდება შემდეგი:

1. მაცხოვრებელი ხედავს ღილაკების სრულ სიმრავლეს, რომლებიც ამჟამად დაჭერილია.
2. მხოლოდ ამ ინფორმაციისა და $v_f$ მნიშვნელობის საფუძველზე მას შეუძლია აირჩიოს ამჟამად დაუჭერელი ღილაკების ნებისმიერი ქვესიმრავლე, რომლებიც შეესაბამება საკუთარ სართულზე **მკაცრად მაღლა** მდებარე სართულებს და დააჭიროს მათ.
3. ლიფტი ადის მომდევნო უდაბლეს სართულზე, რომლის შესაბამისი ღილაკიც დაჭერილი იყო (ან სახურავზე, თუ ყველა სართული, რომლებიც შეესაბამება დაჭერილ ღილაკებს, უკვე მონახულებულია).

თუ ლიფტი არ ჩერდება რომელიმე სართულზე, შესაბამის მობინადრეს არ შეუძლია რომელიმე ღილაკის დაჭერა.

როდესაც ლიფტი აღწევს სახურავს, კარლსონი ხედავს მხოლოდ დაჭერილი ღილაკების საბოლოო სიმრავლეს. ამ ინფორმაციის გამოყენებით ის ცდილობს აღადგინოს $v_0, v_1, \ldots, v_N$ მნიშვნელობებიდან რაც შეიძლება მეტი.

$v$-ის მნიშვნელობები ფიქსირდება თქვენი პროგრამის დაწყებამდე და არ იცვლება პროცესის მიმდინარეობისას.

## იმპლემენტაციის დეტალები

იქნება $T$ სატესტო შემთხვევა.

ატვირთეთ ერთი ფაილი, რომელიც იმპლემენტაციას უკეთებს ამ ორ ფუნქციას.

პირველი ფუნქცია, რომლის იმპლემენტაციაც უნდა მოახდინოთ არის შემდეგი:

```
std::vector<int> press_buttons(int subtask, int N,
                               int f, int v, std::vector<int> p)
```

- $\text{subtask}$: ქვეამოცანა, რომლისთვისაც ეს ფუნქცია გამოიძახება, სადაც $0 \leq \text{subtask} \leq 4$;
- $N$: ბოლო სართულის ნომერი;
- $f$: მიმდინარე სართული, სადაც $0 \leq f \leq N$;
- $v$: მნიშვნელობა $v_f$, რომელიც დაწერილია $f$ სართულზე;
- $p$: ამჟამად დაჭერილი ღილაკები ზრდადობის მიხედვით.

ეს ფუნქცია გამოიძახება, როცა ლიფტი იღება $f$ სართულზე (თუ ან $f = 0$, ან $f$ სართულის შესაბამისი ღილაკი დაჭერილი იყო უფრო ადრეულ გაჩერებაზე). ეს ფუნქცია არ გამოიძახება კონკრეტული სართულისთვის, თუ შესაბამისი ღილაკი მანამდე არ იყო დაჭერილი.

დასაბრუნებელი მნიშვნელობა არის ახალი დასაჭერი ღილაკების სია. დაბრუნებულ მასივში მნიშვნელობების თანმიმდევრობა არ არის მნიშვნელოვანი. ყოველი დაჭერილი $x$ ღილაკი უნდა აკმაყოფილებდეს შემდეგს:

- $f < x \leq N$;
- $x$ უკვე არ არის $p$-ში და $x$ ჩნდება ზუსტად ერთხელ დაბრუნებულ მასივში.

მეორე ფუნქცია, რომლის იმპლემენტაციაც უნდა მოახდინოთ არის შემდეგი:

```
std::vector answer(int subtask, int N, std::vector p)
```

- $\text{subtask}$: ქვეამოცანა, რომლისთვისაც ეს ფუნქცია გამოიძახება, სადაც $0 \leq \text{subtask} \leq 4$;
- $N$: ბოლო სართულის ნომერი;
- $p$: დაჭერილი ღილაკების საბოლოო სიმრავლე დალაგებული ზრდადობის მიხედვით.

ეს ფუნქცია გამოიძახება ზუსტად ერთხელ თითო ტესტისთვის, როცა ლიფტი აღწევს სახურავამდე.

დაბრუნებული მასივის სიგრძე უნდა იყოს $N+1$, მისი $i$-ური ელემენტი უნდა უდრიდეს $v_i$-ს, ხოლო მნიშვნელობები, რომელთა **აღდგენაც კარლსონს არ შეუძლია**, უნდა ჩანაცვლდეს $-1$-ით.

**მნიშვნელოვანი:** შენობაში მყოფ ადამიანებს არ შეუძლიათ ერთმანეთთან ურთიერთობა სხვა გზით, გარდა დაჭერილი ღილაკებისა. ამის უზრუნველსაყოფად, შენი პროგრამა გაეშვება როგორც $N+2$ ცალკეული პროცესი, თითო თითოეული სართულისთვის და ერთი — სახურავისთვის. სართულების თითოეულ პროცესში იქნება მაქსიმუმ $T$ გამოძახება `press_buttons` ფუნქციისა, ხოლო სახურავის პროცესში — ზუსტად $T$ გამოძახება `answer` ფუნქციისა. ამიტომ შენი პროგრამა არ უნდა ეყრდნობოდეს რაიმე გაზიარებულ მდგომარეობას (მაგალითად, გლობალურ ცვლადებს). თითოეულ პროცესს ექნება 64 MiB მეხსიერება, რომელიც განცალკევებულია სხვა პროცესების მეხსიერებისგან და ყველა ეს გამოძახება (მაქსიმუმ $(N+1) \times T$ ცალი `press_buttons` და ზუსტად $T$ ცალი `answer`) უნდა დასრულდეს 10 წამში.

## შეზღუდვები

- $N = 60$
- $T \leq 10\,000$
- $0 \leq v_i \leq 3$ ყოველი $0 \leq i \leq N$-სათვის

## მაგალითი

მაგალითის ტესტს აქვს 1 სატესტო შემთხვევა შემდეგი საიდუმლო მნიშვნელობებით:

```
1 1 0 1 1 1 1 1 1 0 1 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 1
1 0 1 1 0 1 1 1 0 1 0 1 0 1 1 0 1 0 1 1 1 1 0 1 1 1 1 0 1 1
```

ურთიერთქმედების მაგალითი:

| მონაწილის პროგრამა | ჟიურის პროგრამა |
|---|---|
| | `press_buttons(0, 60, 0, 1, {})` |
| `return {2}` | |
| | `press_buttons(0, 60, 2, 0, {2})` |
| `return {13, 42}` | |
| | `press_buttons(0, 60, 13, 0, {2, 13, 42})` |
| `return {}` | |
| | `press_buttons(0, 60, 42, 1, {2, 13, 42})` |
| `return {}` | |
| | `answer(0, 60, {2, 13, 42})` |
| `return {1, -1, 0, -1, -1, ..., -1}` | |

ეს ურთიერთქმედება სწორად აღადგენს 0-ვან და მე-2 მნიშვნელობებს. ყველა სხვა დაბრუნებული მნიშვნელობა უდრის $-1$-ს, რადგან კარლსონმა ისინი არ აღადგინა.

## ქვეამოცანები

| ქვეამოცანა | ქულები | დამატებითი შეზღუდვები | სართულების რაოდენობა, რომელიც უნდა აღდგეს სრული ქულების მისაღებად |
|---|---|---|---|
| $0$ | $0$ | მაგალითი. | – |
| $1$ | $15$ | $0 \leq v_i \leq 1$<br>$v_0 = v_N = 1$<br>თუ $v_i = 0$, მაშინ $v_{i+1} = 1$ ყოველი $0 \leq i < N$-ისათვის | $60$ |
| $2$ | $35$ | $0 \leq v_i \leq 1$ | $40$ |
| $3$ | $15$ | $0 \leq v_i \leq 2$ | $30$ |
| $4$ | $35$ | $0 \leq v_i \leq 3$ | $25$ |

# შეფასება

თუ თქვენი პროგრამა დააბრუნებს არასწორად აღდგენილ მნიშვნელობას ან შეასრულებს არავალიდურ ოპერაციას (მაგალითად, დააბრუნებს არასწორი სიგრძის `vector`-ს) ქვეამოცანის რომელიმე ტესტში, თქვენი ამოხსნა ამ ქვეამოცანისთვის მიიღებს 0 ქულას.

ტესტისთვის თქვენი ამოხსნის *აღდგენილი რაოდენობა* არის იმ პოზიციების რიცხვი, რომელთა დაბრუნებული მნიშვნელობა არ არის $-1$. ავღნიშნოთ $K$-თი ქვეამოცანის ყველა ტესტს შორის მინიმალური აღდგენილი რაოდენობა (თითოეულ ქვეამოცანას აქვს მხოლოდ $1$ ტესტი, რომელიც შეიცავს $10\ 000$-მდე შემთხვევას). შემდეგ, თითოეული ქვეამოცანის ქულები გამოითვლება ქვემოთ მოცემული ცხრილის მიხედვით.

| ქვეამოცანა | $K$-ს დიაპაზონი | ქულები |
|---|---|---|
| $0$ | – | $0$ |
| $1$ | $K < 60$ | $0.25K$ |
| | $K \geq 60$ | $15$ |
| $2$ | $K < 30$ | $0.5K$ |
| | $30 \leq K < 40$ | $2K - 45$ |
| | $K \geq 40$ | $35$ |
| $3$ | $K < 30$ | $0.5K$ |
| | $K \geq 30$ | $15$ |
| $4$ | $K < 20$ | $0.7K$ |
| | $20 \leq K < 25$ | $4K - 66$ |
| | $K \geq 25$ | $35$ |

# სატესტო გრადერი

სატესტო გრადერი ფუნქციის ყველა გამოძახებას ყველა სატესტო შემთხვევისთვის ერთ გაშვებაში ასრულებს.

შესატანი მონაცემების ფორმატი შემდეგია:

- ხაზი $1$: სამი მთელი რიცხვი – სატესტო შემთხვევების რაოდენობა $T$, ბოლო სართულის ნომერი $N$ და ქვეამოცანის ნომერი $S$;
- ხაზი $1+i$: $N+1$ ცალი მთელი რიცხვი $v_0, v_1, \ldots, v_N$, სადაც $v_f$ არის $f$ სართულზე ჩაწერილი საიდუმლო მნიშვნელობა ამ სატესტო შემთხვევისთვის.

გამოსატანი მონაცემების ფორმატი შემდეგია:

- ხაზი $i$: ერთი მთელი რიცხვი – $v$-ის სწორად აღდგენილი მნიშვნელობების რაოდენობა $i$ სატესტო შემთხვევაში;
- ხაზი $T+1$: საბოლოო ქულები.

უფრო დეტალური ფიდბექისთვის, შეცვალეთ მაკრო `DETAILED` `false`-დან `true`-ზე შემფასებლის პირველ ხაზში.

შეგიძლიათ დაუშვათ, რომ გრეიდერმა თავად შექმნას სატესტო შემთხვევები ($v_f$-ის მნიშვნელობები) თქვენთვის, მაკრო `AUTO_GENERATE`-ს `false`-დან `true`-ზე შეცვლით გრეიდერის მეორე ხაზში.