Elevator

Ժամանակի սահմանափակում: 10 վայրկյան
Հիշողության սահմանափակում: 64 MiB (յուրաքանչյուր գործընթացի համար)

Սա հաղորդակցման խնդիր է

Կա մի շենք՝ 0-ից N համարակալված հարկերով: 0-րդ հարկը ամենացածր հարկն է, իսկ N-րդ հարկից վերև գտնվում է տանիքը: Այսինքն՝ տանիքը ներառյալ, շենքն ունի $N+2$ մակարդակ:

Յուրաքանչյուր f-րդ հարկում ($0 \le f \le N$) ապրում է ճիշտ մեկ բնակիչ, և այդ հարկին վերագրված է գաղտնի v_f ամբողջ թիվ ($0 \le v_f \le 3$), որը հայտնի է միայն տվյալ հարկի բնակչին: Կառլսոնն ապրում է շենքի տանիքում: Նրա նպատակն է վերականգնել $v_0, v_1, \ldots, v_N$ արժեքներից հնարավորինս շատերը: Յուրաքանչյուր բնակիչ **փորձում է օգնել** Կառլսոնին, որպեսզի նա կարողանա վերականգնել հնարավորինս շատ արժեքներ:

Հաղորդակցվելու համար բնակիչները կարող են օգտվել վերելակից հետևյալ կերպ:

Սկզբում վերելակը գտնվում է 0-րդ հարկում և կարող է շարժվել միայն դեպի վեր: Վերելակի ներսում կան 1-ից N համարակալված կոճակներ: Ի սկզբանե ոչ մի կոճակ սեղմված չէ: Երբ որևէ կոճակ սեղմվում է, այն ընդմիշտ մնում է սեղմված: Վերելակը կանգ է առնում, և դռները բացվում են f-րդ հարկում, եթե $f = 0$, կամ եթե f-ին համապատասխան կոճակը մինչ այդ սեղմված է եղել: Ամենաբարձր սեղմված կոճակին համապատասխան հարկն այցելելուց հետո (կամ 0-րդ հարկից հետո, եթե որևէ կոճակ սեղմված չի եղել) վերելակը, առանց այլ հարկերում կանգ առնելու, հասնում է տանիք:

Երբ վերելակը կանգ է առնում f-րդ հարկում, տեղի է ունենում հետևյալը.

1. Բնակիչը տեսնում է մինչ այդ սեղմված բոլոր կոճակները:
2. Հիմնվելով միայն սեղմված կոճակների և սեփական v_f արժեքի վրա՝ նա կարող է ընտրել դեռևս չսեղմված կոճակների որևէ բազմություն, որոնց համապատասխան հարկերը **խիստ բարձր** են իր հարկից: Բնակիչը սեղմում է այդ կոճակները:
3. Վերելակը բարձրանում է դեպի դեռևս չայցելված՝ սեղմված կոճակին համապատասխանող ամենացածր հարկը (կամ դեպի տանիք, եթե սեղմված կոճակներին համապատասխանող բոլոր հարկերն արդեն այցելվել են):

Եթե վերելակը կանգ չի առնում որևէ հարկում, այդ հարկի բնակիչը չի կարող ոչ մի կոճակ սեղմել։

Երբ վերելակը հասնում է տանիք, Կառլսոնը տեսնում է միայն վերջնական սեղմված կոճակների բազմությունը։ Օգտագործելով այդ տեղեկությունը՝ նա փորձում է վերականգնել $v_0, v_1, \ldots, v_N$ արժեքներից հնարավորինս շատերը։

v-ի արժեքները ֆիքսվում են նախքան ծրագրի աշխատանքի սկսվելը և աշխատանքի ընթացքում չեն փոփոխվում։

Իրականացման մանրամասներ

Ծրագիրը կաշխատի T թեստերի վրա։

Ներկայացրեք մեկ ֆայլ, որը կիրականացնի հետևյալ ֆունկցիաները։

Առաջին ֆունկցիան, որը պետք է իրականացնեք, հետևյալն է.

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

- subtask: Այն ենթախնդրի համարը, որի շրջանակում կանչվում է ֆունկցիան, որտեղ $0 \leq \text{subtask} \leq 4$։
- N: Վերջին հարկի համարը։
- f: Ընթացիկ հարկը, որտեղ $0 \leq f \leq N$։
- v: f-րդ հարկում գրված v_f արժեքը։
- p: Այդ պահին սեղմված կոճակներին համապատասխանող հարկերը՝ աճման կարգով։

Այս ֆունկցիան կանչվում է, երբ վերելակի դռները բացվում են f-րդ հարկում, այսինքն՝ երբ $f = 0$, կամ երբ f-րդ հարկին համապատասխան կոճակը սեղմվել է նախորդ կանգառներից մեկում։ Ֆունկցիան չի կանչվի այն հարկերի համար, որոնց համապատասխանող կոճակները սեղմված չեն։

Վերադարձվող արժեքը նոր սեղմվելիք կոճակների բազմությունն է։ Վերադարձվող ցուցակում արժեքների հերթականությունը նշանակություն չունի։

Յուրաքանչյուր նոր սեղմվող x կոճակ պետք է բավարարի հետևյալ պայմաններին.

- $f < x \leq N$։
- x-ը չկա p-ում, և վերադարձվող ցուցակում x-ը հանդիպում է ճիշտ մեկ անգամ։

Երկրորդ ֆունկցիան, որը պետք է իրականացնեք, հետևյալն է.

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

- subtask: Այն ենթախնդրի համարը, որի շրջանակում կանչվում է ֆունկցիան, որտեղ $0 \leq \text{subtask} \leq 4$:
- N: Վերջին հարկի համարը:
- p: Վերջնական սեղմված կոճակներին համապատասխանող հարկերը՝ աճման կարգով:

Այս ֆունկցիան յուրաքանչյուր թեստի համար կանչվում է ճիշտ մեկ անգամ՝ երբ վերելակը հասնում է տանիք:

Վերադարձվող զանգվածը պետք է ունենա $N+1$ երկարություն: Դրա i-րդ տարրը պետք է հավասար լինի v_i-ին, իսկ այն արժեքները, որոնք Կառլսոնը **չի կարող վերականգնել**, պետք է փոխարինվեն -1-երով:

Կարևոր: Բնակիչներն ու Կառլսոնը չեն կարող միմյանց հետ հաղորդակցվել որևէ այլ եղանակով՝ բացի սեղմված կոճակներն օգտագործելուց: Դա երաշխավորելու համար ձեր ծրագիրը կաշխատեցվի $N+2$ առանձին գործընթացներում՝ մեկը յուրաքանչյուր հարկի համար և մեկը տանիքի համար: Յուրաքանչյուր հարկի գործընթացում `press_buttons` ֆունկցիան կկանչվի առավելագույնը T անգամ, իսկ տանիքի գործընթացում `answer` ֆունկցիան կկանչվի ճիշտ T անգամ: Հետևաբար ձեր ծրագիրը չպետք է ենթադրի, որ տվյալները որևէ կերպ կարող են փոխանցվել գործընթացների միջև (օրինակ՝ գլոբալ փոփոխականների միջոցով): Յուրաքանչյուր գործընթացին հատկացվում է 64 MiB հիշողություն, որն անկախ է մյուս գործընթացների հիշողությունից, և բոլոր այս կանչերը (մինչև $(N+1) \times T$ հատ `press_buttons` կանչ և ճիշտ T հատ `answer` կանչ) պետք է իրականացվեն 10 վայրկյանի ընթացքում:

Սահմանափակումներ

- $N = 60$
- $T \leq 10\,000$
- $0 \leq v_i \leq 3$ յուրաքանչյուր $0 \leq i \leq N$-ի համար

Օրինակ

Օրինակը բաղկացած է ճիշտ 1 թեստից՝ հետևյալ արժեքներով.

```
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
```

Ստորև ներկայացված է հաղորդակցման օրինակ.

Մասնակցի ծրագիր	Ժյուրիի ծրագիր
	`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}`	

Այս հաղորդակցությամբ ճիշտ վերականգնվում են 0-րդ և 2-րդ հարկերի արժեքները։ Մնացած բոլոր տարրերը -1 պետք է վերադարձնել, քանի որ Կառլսոնը չի կարող վերականգնել դրանք։

Ենթախնդիրներ

Ենթախնդիր	Միավորներ	Լրացուցիչ սահմանափակումներ	Վերականգնված արժեքների քանակը լրիվ միավորի համար
0	0	Օրինակը	-
1	15	$0 \le v_i \le 1$ $v_0 = v_N = 1$ եթե $v_i = 0$, ապա $v_{i+1} = 1$ յուրաքանչյուր $0 \le i < N$-ի համար	60
2	35	$0 \le v_i \le 1$	40
3	15	$0 \le v_i \le 2$	30
4	35	$0 \le v_i \le 3$	25

Գնահատում

Եթե ձեր ծրագիրը որևէ արժեք սխալ է վերականգնում կամ որևէ թեստում կատարում է անվավեր գործողություն (օրինակ՝ վերադարձնում է սխալ երկարությամբ `vector`), ապա այդ ենթախնդրի համար ձեր լուծումը կստանա 0 միավոր։

Թեստի համար *վերականգնվածների քանակ* կանվանենք այն դիրքերի քանակը, որոնցում արժեքը -1 չէ։ Թող K-ն լինի տվյալ ենթախնդրի բոլոր թեստերի համար վերականգնվածների քանակների նվազագույնը (յուրաքանչյուր ենթախնդիր բաղկացած է մինչև $10\,000$ թեստից)։ Այնուհետև յուրաքանչյուր ենթախնդրի միավորները հաշվարկվում են ըստ հետևյալ աղյուսակի։

Ենթախնդիր	K-ի միջակայքեր	Միավորներ
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

Գրեյդերի նմուշ

Գրեյդերի նմուշը բոլոր թեստերի համար բոլոր ֆունկցիաների կանչերը կատարում է մեկ գործարկման ընթացքում։

Մուտքի ձևաչափը հետևյալն է.

- Տող 1: երեք ամբողջ թիվ՝ թեստերի քանակը T, վերջին հարկի համարը N և ենթախնդրի համարը S։
- Տող $1 + i$: $N + 1$ ամբողջ թիվ՝ $v_0, v_1, \ldots, v_N$, որտեղ v_f-ը տվյալ թեստում f-րդ հարկի արժեքն է։

Ելքի ձևաչափը հետևյալն է.

- Տող i: մեկ ամբողջ թիվ՝ i-րդ թեստում v-ի ճիշտ վերականգնված արժեքների քանակը։
- Տող $T+1$: վերջնական միավորը։

Գրեյդերի ավելի մանրամասն արձագանք ստանալու համար գրեյդերի առաջին տողում `DETAILED` մակրոսի արժեքը `false`-ից փոխեք `true`-ի։

Կարող եք նաև գրեյդերին թույլ տալ ինքնաշխատ գեներացնել թեստերը (v_f-ի արժեքները)՝ գրեյդերի երկրորդ տողում `AUTO_GENERATE` մակրոսի արժեքը `false`-ից փոխելով `true`-ի։