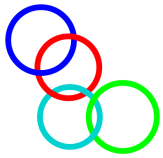
Պարաշյուտի օղակներ

Ներկայիս պարաշյուտի նախատիպի մի բարդ տարբերակ նկարագրված է Լեոնարդոի *Codex Atlanticus* (ca. 1485) աշխատանքում: Լեոնարդոի պարաշյուտը իրենից ներկայացնում էր մի բրգածև կոնստրուկցիա, որի վրա ամուր ձգված էր կտոր:

Կապակցող օղակներ

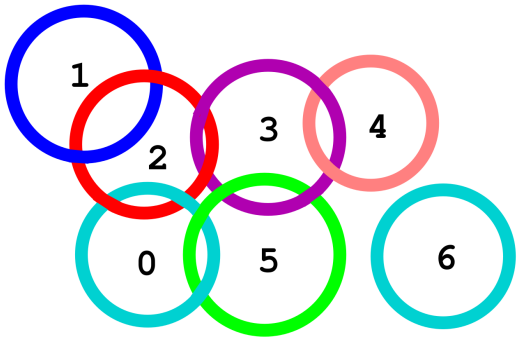
Ավելի քան 500 տարի անց, պարաշյուտիստ Ադրիան Նիկոլասը փորձարկեց Լեոնարդոյի նախագիծը: Դրա համար նա ժամանակակից թեթևացված կոնցտրուկցիան կապեց մարդու մարմնին :Մեմ ցանկանում եմ օգտագործել կապող օղակները, ապահովության համար կիրառելով նաև օղակները հագուստին ամրացնող կեռիկներ: Օղակը պատրաստված է ճկուն և ամուր նյութից: Ամեն օղակ կարող է հեշտորեն փակվել և հետո նորից բացվել: "Շղթան" կապող օղակների համակ կառուցվածք է: "Շղթան" օղակների հաջորդականություն է, որտեղ ամեն օղակ կապված է միայն իր երկուսից ոչ ավելի հարևանների հետ, ինչպես դա ներկայացված է ներքևում: Այս հաջորդականությունը պետք է ունենա սկիզբ և վերջ (օղակներ,որոնցից յուրաքանչյուրը կապված է առավելագույնը մեկ օղակի հետ): Մասնավորապես, մեկ օղակը նույնպես շղթա է:



Այլ կառուցվածքներ նույնպես հնարավոր են, քանի որ օղակը կարող է կապվել երեք և ավելի այլ օղակների հետ: Կասենք, որ օղակը "կրիտիկական" է, եթե այն բացելուց և հեռացնելուց հետո, մնացած օղակները կազմում են շղթաների բազմություն, կամ այլևս օղակներ չեն մնացել: Այլ խոսքերով, կարող է ոչինչ չմնա, բայց լինի շղթա:

Օրինակ

Դիտարկենք հետևյալ պատկերի 7 օղակները, համարակալված 0-ից 6-ով: Կան երկու կրիտիկական օղակներ: Մի կրիտիկականը 2-ն է. այն հեռացնելուց հետո մնացած օղակները շղթաներ են կազմում [1], [0, 5, 3, 4] և [6]: Մեկ այլ կրիտիկական օղակ էլ 3-ն է. այն հեռացնելուց հետո մնացած օղակները շղթաներ են կազմում [1, 2, 0, 5], [4] և [6]-ը: Եթե մենք հեռացնենք այլ օղակ, չենք ստանա հնարավոր շղթաների բազմություն: Օրինակ, 5 օղակը հեռացնելուց հետո [6]-ը շղթա կլինի, սակայն իրան հետ կապված 0, 1, 2, 3 և 4 օղակները շղթա չեն կազմի:



Խնդիր

Ձեր խնդիրն է ձեր ծրագրի հետ փոխգործողության արդյունքում ստացված կառուցվածքում հաշվել կրիտիկական օղակների քանակը:

Սկզբում կան ինչ-որ քանակությամբ զույգ առ զույգ չհատվող օղակներ: Ապա այդ օղակները միացվում են իրար: Ցանկացած պահին ձեզ կարող են հարցնել, թե ընթացիկ կոնֆիգուրացիայում քանի կրիտիկական օղակ կա: Դուք պետք է իրականացնեք երեք ենթածրագիր:

- `Init`(N) — կանչվում է ճիշտ մեկ անգամ սկզբում, հաղորդելու համար, որ սկզբնական կոնֆիգուրացիայում կան N զույգ առ զույգ չկապված օղակներ՝ համարակալված 0-ից N - 1 (ներառյալ) թվերով:
- `Link`(A, B) — A և B համարներով օղակները կապած են իրար հետ: Երաշխավորվում է, որ A-ն և B-ն տարբեր են և մինչ այդ անմիջականորեն չեն կապվել: Ուրիշ լրացուցիչ պայմաններ A-ի և B-ի համար չկա, մասնավորապես, ֆիզիկական սահմանափակումներից բխող ոչ մի պայման չկա: Պարզ է, որ `Link`(A, B)-ը և `Link`(B, A)-ը համարժեք են:
- `CountCritical`() — կապված օղակների ընթացիկ կոնֆիգուրացիայի համար վերադարձնում է կրիտիկական օղակների քանակը:

Օրինակ

Դիտարկենք մեր նկարը, որտեղ կա N = 7 օղակ և ենթադրենք, որ նրանք սկզբում կապված չեն: Ներկայացնենք ֆայլերի հաջորդականություն, որոնց կատարման արդյունքում կստացվի նկարում պատկերված իրավիճակը:

Կանչ	Վերադարձնում է
`Init`(7)	
`CountCritical`()	7
`Link`(1, 2)	
`CountCritical`()	7
`Link`(0, 5)	
`CountCritical`()	7
`Link`(2, 0)	
`CountCritical`()	7
`Link`(3, 2)	
`CountCritical`()	4
`Link`(3, 5)	
`CountCritical`()	3
`Link`(4, 3)	
`CountCritical`()	2

Ենթախնդիր 1 [20 միավոր]

- $N \leq 5\,000$.
- `CountCritical` ֆունկցիան կանչվում է միայն մեկ անգամ; the function `Link` ֆունկցիան կանչվում է առավելագույնը 5 000 անգամ:

Ենթախնդիր 2 [17 միավոր]

- $N \leq 1\,000\,000$.
- `CountCritical` ֆունկցիան կանչվում է միայն մեկ անգամ մյուս բոլոր կանչերից հետո: `Link` ֆունկցիան կանչվում է առավելագույնը 1 000 000 անգամ:

Ենթախնդիր 3 [18 միավոր]

- $N \leq 20\,000$.
- `CountCritical` ֆունկցիան կանչվում է առավելագույնը 100 անգամ: `Link` ֆունկցիան կանչվում է առավելագույնը 10 000 անգամ:

Ենթախնդիր 4 [14 միավոր]

- $N \leq 100\,000$.
- `CountCritical` և `Link` կանչվում են գումարային առավելագույնը 100 000 անգամ:

Ենթախնդիր 5 [31 միավոր]

- $N \leq 1\,000\,000$.
- `CountCritical` և `Link` կանչվում են գումարային առավելագույնը 1 000 000 անգամ:

Իրականացման մանրամասներ

Դուք պետք է submit անեք ձիշտ մեկ ֆայլ, `rings.c`, `rings.cpp` կամ `rings.pas` անունով: Այդ ֆայլը պետք է իրականացնի վերը նկարագրված ենթածրագրերը օգտագործելով հետևյալ սիգնատուրները:

C/C++ ծրագրեր

```
void Init(int N);
void Link(int A, int B);
int CountCritical();
```

Pascal ծրագրեր

```
procedure Init(N : LongInt);
procedure Link(A, B : LongInt);
function CountCritical() : LongInt;
```

Այս ֆունկցիաները պետք է աշխատեն այնպես, ինչպես նկարագրված է վերևում: Իհարկե դուք կարող եք այլ ֆունկցիաներ ևս իրականացնել ձեր նպատակների համար: Ձեր submit-ները ոչ մի կերպ չպետք է առընչվեն ստանդարտ մուտքի/ելքի և որևէ այլ ֆայլի հետ:

grader-ի օրինակ

grader-ի օրինակը կարդում է մուտքային տվյալները հետևյալ ֆորմատով.

- Տող 1: N, L;
- Տողեր 2, …, L + 1:
  - -1 `CountCritical` ֆունկցիան կանչելու համար;
  - A, B `Link`-ի պարամետրերը:

grader-ի օրինակը տպում է `CountCritical`-ի բոլոր արդյունքները: