



Kiểm tra tính đúng đắn và hiệu năng của chương trình bằng bộ test





Bộ test là gì? **01**

Tính đúng đắn
và hiệu năng **02**

Test chương trình
bằng trình chấm **03**

Nội dung

04 Ví dụ

05 Quiz

06 Nhận xét của mọi
người



01

Bộ test là gì? Tại sao cần bộ test?



Bộ test

Tại sao cần bộ test



02

Tính đúng đắn và hiệu năng



Tính đúng đắn

Chương trình chạy ra output cần tìm => Thuật toán đúng

```
Test 1
0.00 s and 0 KiB
ACCEPT
Test 2
0.00 s and 0 KiB
ACCEPT
Test 3
0.00 s and 0 KiB
ACCEPT
Test 4
0.00 s and 0 KiB
```

```
Test 1
0.00 s and 0 KiB
ACCEPT
Test 2
0.00 s and 0 KiB
WRONG
Test 3
0.00 s and 0 KiB
WRONG
Test 4
0.00 s and 0 KiB
```



Hiệu năng

Mức độ tiêu tốn thời gian và không gian dữ liệu của thuật toán

Time Limit:	1 secs
Source Limit:	50000 Bytes

Ví dụ:

- Giả sử dùng thuật toán QuickSort để sắp xếp 1000 phần tử, tốn ít nhất 200ms.

Quick Sort: $O(n \log n)$

- Thuật toán Interchange Sort sẽ mất 66,7s để hoàn thành.

=> Thuật toán Quick Sort trong trường hợp này có hiệu năng cao hơn thuật toán Interchange Sort

Câu hỏi:

Giả sử có một trường hợp cả hai thuật toán Quick Sort và Merge Sort đều có thời gian thực thi như nhau. Vậy thuật toán nào có hiệu năng cao hơn?



03

Cách test chương trình bằng trình chấm

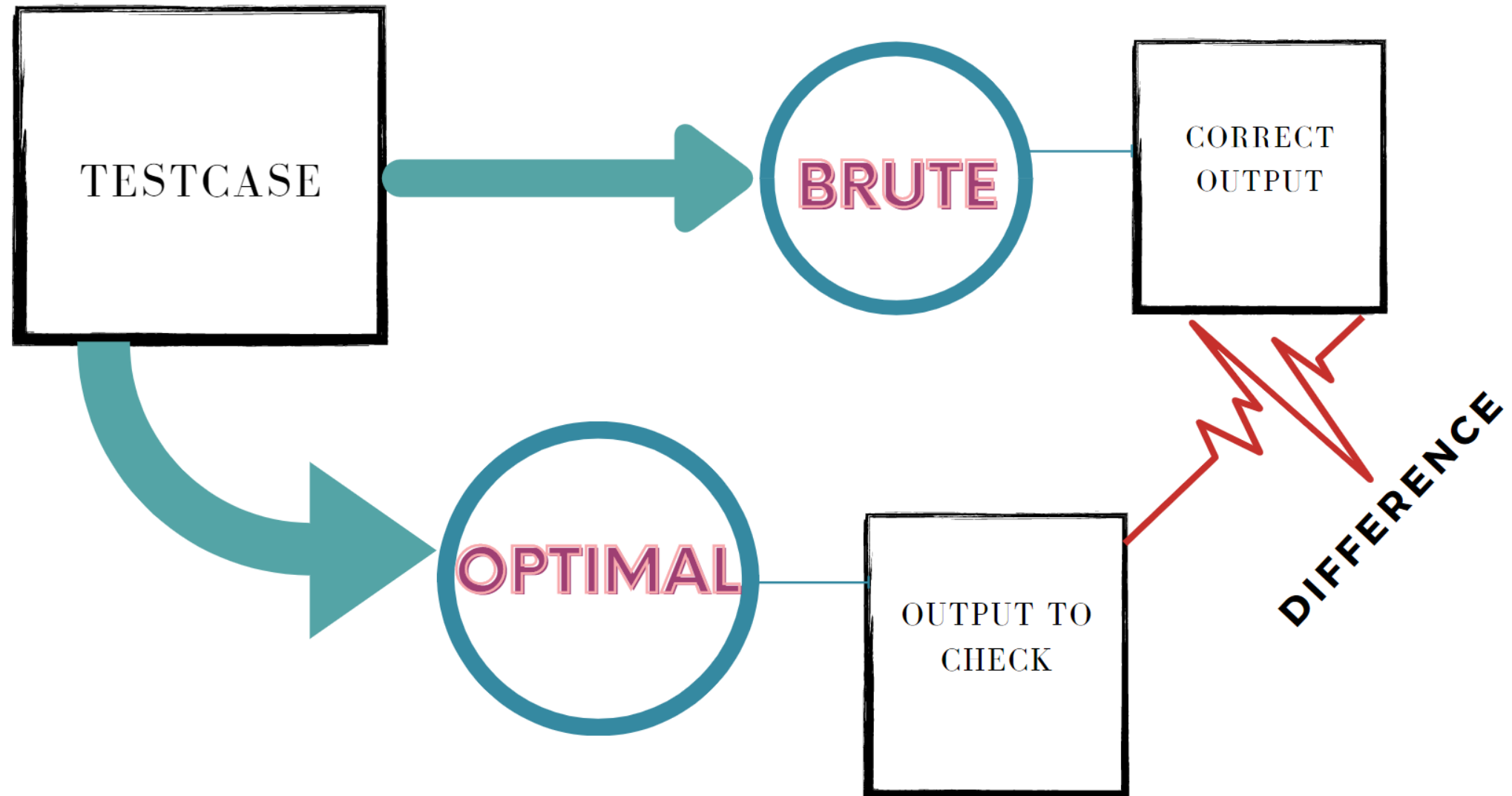


Giới thiệu về trình chấm

Trình chấm là một bộ chương trình giúp bạn tự động sinh ra các test input ngẫu nhiên và tự động chạy hai chương trình lời giải khác nhau với các input đó để so sánh output.

Trình chấm bao gồm 4 chương trình

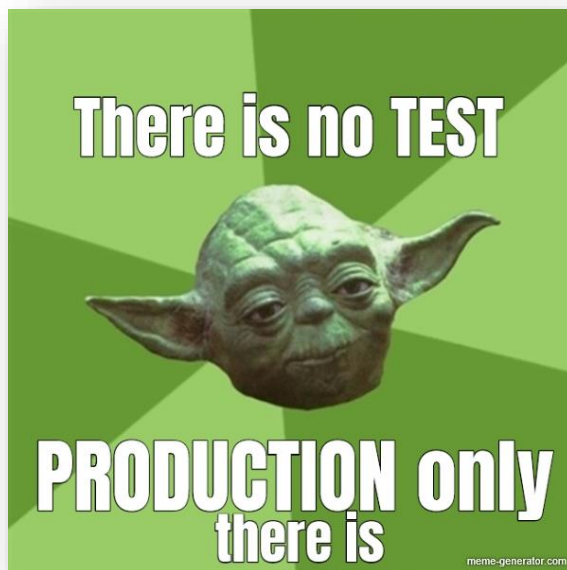
- Trình sinh test
- Lời giải 1
- Lời giải 2
- Trình so test





Trình sinh test

Trình sinh test là 1 chương trình sinh test ngẫu nhiên theo yêu cầu và giới hạn của đề





Một số hàm hữu ích để viết trình sinh test

Hàm sinh số ngẫu nhiên

```
randomNumber.cpp > ...
1  #include<iostream>
2  #include <cstdlib>
3  #include <ctime>
4
5  using namespace std;
6
7  int randomNumber(int a, int b){
8      return rand() % (b - a + 1) + a;
9  }
10
11 int main() {
12     srand(time(NULL));
13     cout << randomNumber(1, 100);
14     return 0;
15 }
```

Hàm sinh mảng ngẫu nhiên

```

6
7  int randomNumber(int a, int b){
8      return rand() % (b - a + 1) + a;
9  }
10
11 void randomArray(int n, int *arr, int a, int b){
12     for (int i = 0; i < n; i++){
13         arr[i] = randomNumber(a,b);
14     }
15 }
16
17 int main() {
18     srand(time(NULL));
19     cout << randomNumber(1, 100);
20     return 0;
21 }
```



Một số hàm hữu ích để viết trình sinh test

Hàm sinh số nguyên Long Long ngẫu nhiên

```
109  mt19937 rd(chrono::steady_clock::now().time_since_epoch().count());
110  #define rand rd
111
112  long long randomLLNumber(long long l, long long h) {
113      assert(l <= h);
114      return l + rd() * 1LL * rd() % (h - l + 1);
115  }
```



Một số hàm hữu ích để viết trình sinh test

Vậy làm sao để sinh ra 1 mảng đã xấp sếp ngẫu nhiên ?





Một số hàm hữu ích để viết trình sinh test

Hàm sinh chuỗi ngẫu nhiên

```
randomString.cpp > ...
1  #include<iostream>
2  #include <cstdlib>
3  #include <ctime>
4  #include <string>
5
6  using namespace std;
7
8  char randomCharacter(){
9      return 'a' + rand() % 25;
10 }
11
12 string randomString(int len){
13     string res = "";
14     for (int i = 0; i < len; i++){
15         res.push_back(randomCharacter());
16     }
17     return res;
18 }
19
```

```
20  int main() {
21      srand(time(NULL));
22      cout << randomString(10);
23      return 0;
24 }
```

```
PS D:\CODE STUFF\C++\CS112> ./randomString
waeddwkkyu
PS D:\CODE STUFF\C++\CS112> ./randomString
kivevbtnxv
PS D:\CODE STUFF\C++\CS112> ./randomString
kkkuwukkk
```



Một số hàm hữu ích để viết trình sinh test

Vậy làm sao để sinh ra 1 chuỗi palindrome?





Một số hàm hữu ích để viết trình sinh test

Hàm sinh cây không trọng số

```
42  set< pair<int,int> > unweighted_tree_generation(int n){
43      vector< vector<int> > adj(n);
44      set< pair<int,int> > container;
45      for (int i = 0; i < n-1; i++){
46          int u = rand() % n;
47          int v = rand() % n;
48          pair<int,int> p = {u,v};
49          adj[u].push_back(v);
50
51          while (container.find(p) != container.end()
52                 || isCyclic(n, adj) == true)
53              {
54                  adj[u].pop_back();
55                  u = rand() % n;
56                  v = rand() % n;
57                  p = {u,v};
58                  adj[u].push_back(v);
59              }
60          container.insert(p);
61      }
62      return container;
63  }
```

```
65  int main() {
66      srand(time(NULL));
67      int n = 6;
68      set<pair<int,int>> container = unweighted_tree_generation(n);
69
70      cout<< n << endl;
71      set<pair<int, int>>::iterator it;
72      for (it=container.begin(); it!=container.end(); ++it)
73          cout<< it->first << " " << it->second << endl;
74      return 0;
75  }
```

```
PS D:\CODE STUFF\C++\CS112> ./randomTree
6
0 3
1 5
2 3
3 4
4 5
```



Một số hàm hữu ích để viết trình sinh test

Vậy làm sao để sinh ra 1 cây có trọng số?





Một số hàm hữu ích để viết trình sinh test

Hàm sinh đồ thị có hướng không trọng số

```
4  set< pair<int,int> >
5  directed_unweighted_graph_generation(int n, int maxEdge){
6  set< pair<int,int> > container;
7  for (int j=1; j<=maxEdge; j++)
8  {
9      int a = 1 + rand() % n;
10     int b = 1 + rand() % n;
11     pair<int, int> p = make_pair(a, b);
12
13     while (container.find(p) != container.end())
14     {
15         a = 1 + rand() % n;
16         b = 1 + rand() % n;
17         p = make_pair(a, b);
18     }
19     container.insert(p);
20 }
21 return container;
22 }
```

```
24 int main() {
25     srand(time(NULL));
26     int n = 6;
27     int m = 10;
28     set<pair<int,int>> container
29     = directed_unweighted_graph_generation(n, m);
30
31     cout<< n << " " << m << endl;
32     set<pair<int, int>>::iterator it;
33     for (it=container.begin(); it!=container.end(); ++it)
34         cout<< it->first << " " << it->second << endl;
35     return 0;
36 }
```



Một số hàm hữu ích để viết trình sinh test

Vậy làm sao để sinh ra 1 đồ thị vô hướng ?





Lời giải 1 và lời giải 2

Lời giải 1 là lời giải tối ưu mà chúng ta cần kiểm tra về tính đúng đắn và hiệu suất

Lời giải 2 là lời giải đúng dễ dàng cài đặt không nhất thiết phải nhanh, thường sẽ cài bằng các thuật toán trau





Tìm lời giải trâu cho bài toán sau

ĐƯỜNG BẬC THANG

Các bậc thang của đường dẫn lên đỉnh núi được đánh số từ 1 đến n , 1 là bậc thang đầu tiên, n – bậc cuối cùng trên đỉnh núi. Ở các bậc có số chẵn 10 người ta ghi số của bậc trên bậc đó. Ngoài ra bậc đầu tiên và bậc cuối cùng cũng cũng được ghi số. Chi phí để sơn mỗi chữ số là như nhau và được coi là bằng 1.

Hãy xác định tổng chi phí của việc đánh dấu trên toàn bộ đường đi.

Dữ liệu: Vào từ thiết bị nhập chuẩn gồm một dòng chứa số nguyên n ($1 \leq n \leq 10^{12}$).

Kết quả: Đưa ra thiết bị xuất chuẩn một số nguyên – tổng chi phí tìm được.

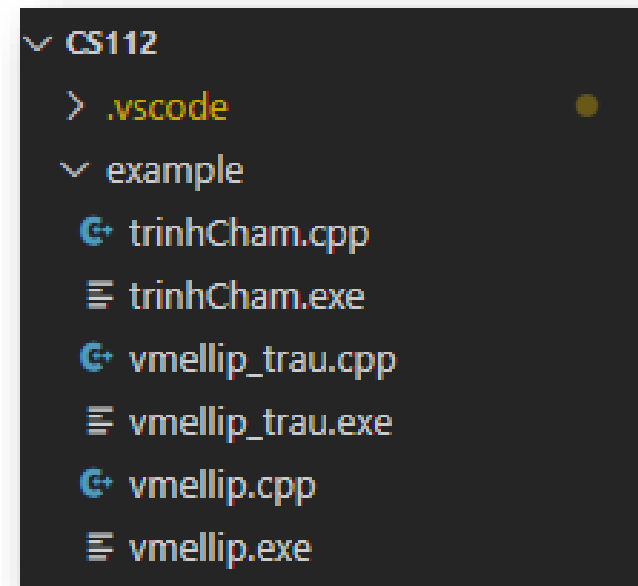
Ví dụ:

INPUT	OUTPUT
23	7



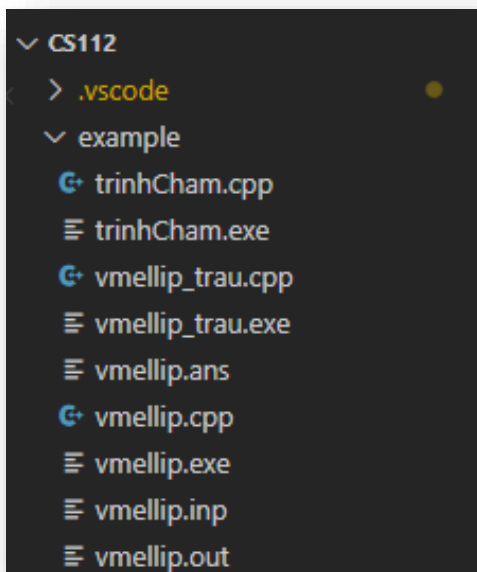
Trình so test

Chương trình này có nhiệm vụ duyệt N lần, mỗi lần duyệt thì chạy trình sinh test rồi chạy 2 chương trình lời giải 1 và lời giải 2 so sánh output mà đưa ra kết quả





Template cho trình chấm C++



```
19  int main()
20  {
21      string NAME = "baitap";
22      int NTEST = 100;
23      srand(time(NULL));
24      for (int iTest = 1; iTest <= NTEST; iTest++)
25      {
26          ofstream inp((NAME + ".inp").c_str());
27          /**
28           |   Code phần sinh test ở đây
29          ***/
30          inp.close();
31
32          auto start = high_resolution_clock::now();
33          system((NAME + ".exe").c_str());
34          auto stop = high_resolution_clock::now();
35          auto duration1 = duration_cast<microseconds>(stop - start);
36
37          start = high_resolution_clock::now();
38          system((NAME + "_trau.exe").c_str());
39          stop = high_resolution_clock::now();
40          auto duration2 = duration_cast<microseconds>(stop - start);
41
42          if (system(("fc " + NAME + ".out " + NAME + ".ans").c_str()) != 0)
43          {
44              cout << "Test " << iTest << ": WRONG!\n";
45              return 0;
46          }
47          cout << "Test " << iTest << ": CORRECT!\n";
48          cout << NAME + ".exe" + "excution time: " << duration1.count() << " microseconds" << endl;
49          cout << NAME + "_trau.exe" + "excution time: " << duration2.count() << " microseconds" << endl;
50      }
51      return 0;
52  }
```



Chương trình sẽ tự động sinh test và kiểm tra

```
PS D:\CODE STUFF\C++\CS112\example> ./trìnhCham
Comparing files vmellip.out and VMELLIP.ANS
FC: no differences encountered

Test 1: CORRECT!
vmellip.exe excution time: 76797 microseconds
vmellip_trau.exe excution time: 56845 microseconds
Comparing files vmellip.out and VMELLIP.ANS
FC: no differences encountered

Test 2: CORRECT!
vmellip.exe excution time: 54852 microseconds
vmellip_trau.exe excution time: 39893 microseconds
Comparing files vmellip.out and VMELLIP.ANS
FC: no differences encountered

Test 3: CORRECT!
vmellip.exe excution time: 53855 microseconds
vmellip_trau.exe excution time: 35904 microseconds
Comparing files vmellip.out and VMELLIP.ANS
FC: no differences encountered
```

Chương trình sẽ dừng lại khi có 1 test không khớp

```
Test 16: CORRECT!
vmellip.exe excution time: 54852 microseconds
vmellip_trau.exe excution time: 35905 microseconds
Comparing files vmellip.out and VMELLIP.ANS
FC: no differences encountered

Test 17: CORRECT!
vmellip.exe excution time: 53856 microseconds
vmellip_trau.exe excution time: 35904 microseconds
Comparing files vmellip.out and VMELLIP.ANS
***** vmellip.out
81
24
***** VMELLIP.ANS
81
8
*****

Test 18: WRONG!
PS D:\CODE STUFF\C++\CS112\example> BAKA
```



04

Ví dụ



Ví dụ 1



Problem 7.08

Problem7.08

Nhập ma trận vuông A ($n \times n$). Kiểm tra A có phải là ma trận Frobenius hay không.

Ma trận Frobenius được định nghĩa là ma trận vuông thỏa các tính chất sau:

1. Tất cả các phần tử trên đường chéo chính bằng 1.
2. Có tối đa một cột mà các phần tử dưới đường chéo chính có thể nhận giá trị bất kì.
3. Ngoài ra, tất cả các phần tử khác có giá trị bằng 0.

INPUT

- 2 số nguyên là số dòng, số cột của ma trận A
- Giá trị các phần tử của ma trận A là số thực

OUTPUT

- "Yes", nếu A là ma trận Frobenius
- "No", nếu A không là ma trận Frobenius



Ví dụ 2



Thầy Sơn mới thiết kế một hệ thống N máy tính (các máy tính được đánh số từ 1 đến N) được nối lại thành một mạng bởi M kênh nối, mỗi kênh nối hai máy nào đó và cho phép ta truyền tin từ máy này sang máy kia. Nhưng không may rằng Thầy Sơn đã bị 1 hacker nào đó tấn công vào một máy bất kì trong N máy. Để khắc phục sự cố thì thầy Sơn cần liệt kê ra các máy đã bị tấn công. Mặc dù vậy nhưng mà hệ thống máy tính của thầy rất lớn nên không thể liệt kê bằng cách thông thường được. Em hãy viết chương trình giúp thầy xác định các máy bị tấn công? Biết rằng, nếu một máy bị tấn công thì các máy được kết nối trực tiếp cũng sẽ bị tấn công.



Dữ liệu:

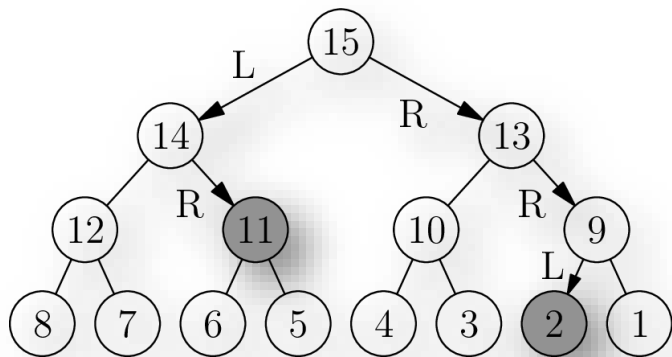
- Dòng thứ nhất gồm số nguyên N, M ($1 \leq N, M \leq 10^6$). N là số máy tính và M là số kênh nối.
- M dòng tiếp theo, mỗi dòng gồm 2 số nguyên u và v ($1 \leq u \neq v \leq N$), cho biết máy u sẽ kết nối trực tiếp với máy v .
- Dòng cuối cùng chứa số nguyên s ($1 \leq s \leq N$), là số hiệu của máy bị tấn công.

Kết quả:

- Dòng đầu tiên in ra số lượng các máy bị tấn công.
- Dòng thứ hai in ra dãy số hiệu các máy tính bị tấn công theo thứ tự tăng dần.



Ví dụ 3



Numbers On a Tree

Task

Lovisa's task is to calculate the label of a node, given the height of the tree H and the description of the path from the root.

Input

The only line of input contains the height of the tree H , $1 \leq H \leq 30$ and a string consisting of the letters 'L' and 'R', denoting a path in the tree starting in the root. The letter 'L' denotes choosing the left child, and the letter 'R' choosing the right child. The description of the path may be empty and is at most H letters.

Output

Output one line containing the label of the node given by the path.

Sample Input 1

3 LR

Sample Output 1

11

Sample Input 2

3 RRL

Sample Output 2

2

Sample Input 3

2

Sample Output 3

7





THANKS FOR LISTENING!
