Wprowadzenie do Algorytmiki. Podstawy.

Artur Laskowski

21 października 2021, Poznań

Eliminacje do zawodów

Leaderboard

All | Friends | Filter by: Select filter | Type username to compare | Compare

Rank	Team	Solved	Time	A	B	C	D	E	F	G	H	I
1	misztsu	6	343		1 (4:47 + 0)	1 (25:18 + 0)	1 (55:01 + 0)	1 (40:06 + 0)	2 (87:13 + 20)	1 (110:27 + 0)		
2	KamilP	6	401		1 (4:49 + 0)	1 (63:53 + 0)	2 (33:50 + 20)	1 (42:34 + 0)	2 (128:10 + 20)	1 (87:43 + 0)	2 (0:00 + 0)	
3	pavveu	4	407		1 (7:35 + 0)	3 (0:00 + 0)	2 (134:42 + 20)	2 (54:22 + 20)		1 (169:36 + 0)		
4	Ztrk1	3	142		1 (12:50 + 0)		3 (0:00 + 0)	1 (31:16 + 0)		1 (97:14 + 0)		
5	sebastian_micho1	3	174		1 (31:55 + 0)	2 (74:38 + 20)	7 (0:00 + 0)	1 (46:32 + 0)				
6	k_jaworski	3	213		1 (20:21 + 0)	2 (114:32 + 20)	1 (0:00 + 0)	2 (37:47 + 20)				
7	maciej_falbogow1	3	275		1 (2:20 + 0)		5 (159:30 + 80)	1 (32:28 + 0)	25 (0:00 + 0)			
8	jan_kiwlenko	3	394		1 (13:17 + 0)	1 (0:00 + 0)	1 (162:20 + 0)	3 (177:38 + 40)	2 (0:00 + 0)	4 (0:00 + 0)		
9	rafa_szubert	2	147		2 (41:10 + 20)	2 (0:00 + 0)		2 (65:41 + 20)		2 (0:00 + 0)		
10	Antollo	2	204		1 (59:41 + 0)			4 (84:00 + 60)				

www.hackerrank.com/eliminacje-2021-pp

C++ Tricks & Hacks

C++ Tricks & Hacks

Kompilacja z użyciem g++ (Unix)

```
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  vim hello.cpp
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  cat hello.cpp
#include <iostream>

using namespace std;

int main() {
        cout << "Hello, World!" << endl;
        return 0;
}
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  g++ hello.cpp -o A.out
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  ./A.out
Hello, World!
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures
```

Testowanie w Bashu

```
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  cat addition.cpp
#include <iostream>

using namespace std;

int main() {
        int a, b;
        cin >> a >> b;
        cout << a + b << endl;
        return 0;
}
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  cat input/test01.txt input/test02.txt
2 3
123 567
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  g++ addition.cpp -o B.out
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  ./B.out < input/test01.txt > output/test01.txt
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  ./B.out < input/test02.txt > output/test02.txt
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  cat output/test01.txt output/test02.txt
5
690
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures
```

Testowanie skryptem

```
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  cat test-all.sh
#!/bin/bash
for i in {1..2}
do
        ./$1 < input/test0${i}.txt > output/test0${i}.txt
done
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  chmod +x test-all.sh
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  ll output
total 0
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  ./test-all.sh B.out
 arturlaskowski@lab-seq-2  ~/Documents/github/WdA-lectures  cat output/test01.txt output/test02.txt
5
690
```

Typy danych

Jakie są istotne typy danych?

bool	1	$\langle True; False \rangle$
short int	2	$\langle -32'768; 32'767 \rangle$
unsigned short int	2	$\langle 0; 65'535 \rangle$
int	4	$\langle -2'147'483'648; 2'147'483'647 \rangle$
unsigned int	4	$\langle 0; 4'294'967'295 \rangle$
long long int	8	$\langle$ -(2^{63}) ; $2^{63} - 1$ $\rangle$
unsigned long long int	8	$\langle 0; 18'446'744'073'709'551'615 \rangle$
char	1	$\langle -128; 127 \rangle$
unsigned char	1	$\langle 0; 255 \rangle$
float	4	–
double	8	–
long double	16	–

Typy danych

Jakie typy danych w następujących sytuacjach?

- Przechowywanie l. c. z $\langle 0; 10'000 \rangle$

Typy danych

Jakie typy danych w następujących sytuacjach?

- Przechowywanie l. c. z $\langle 0; 10'000 \rangle$ short int, unsigned short int

Typy danych

Jakie typy danych w następujących sytuacjach?

- Sumowanie 1'000'000 l. c. z $\langle -10'000; 10'000 \rangle$

Typy danych

Jakie typy danych w następujących sytuacjach?

- Sumowanie 1'000'000 l. c. z $\langle -10'000; 10'000 \rangle$ long long int, long long

Typy danych

Jakie typy danych w następujących sytuacjach?

- Przechowywanie 1'000'000'000 l. c. z $\langle 0; 100 \rangle$

Typy danych

Jakie typy danych w następujących sytuacjach?

- Przechowywanie 1'000'000'000 l. c. z $\langle 0; 100 \rangle$ char, unsigned char

Typy danych

Jakie typy danych w następujących sytuacjach?

- Przechowywa liczb zmiennoprzecinkowych

Typy danych

Jakie typy danych w następujących sytuacjach?

- Przechowywa liczb zmiennoprzecinkowych double

Typy danych

Jakie typy danych w następujących sytuacjach?

- Przechowywanie ciągu znaków

Typy danych

Jakie typy danych w następujących sytuacjach?

- Przechowywanie ciągu znaków string $\#include\langle string\rangle$

Typy danych

Jakie typy danych w następujących sytuacjach?

- Przechowywanie l. c. z $\langle 0; 10'000 \rangle$ short int, unsigned short int
- Sumowanie 1'000'000 l. c. z $\langle -10'000; 10'000 \rangle$ long long int, long long
- Przechowywanie 1'000'000'000 l. c. z $\langle 0; 100 \rangle$ char, unsigned char
- Przechowywa liczb zmiennoprzecinkowych double
- Przechowywanie ciągu znaków string $\#include\langle string \rangle$

Tablice

```
int arr[1000];

int* arr = new int[1000];

#include <vector>

std::vector <int> arr(1000, 0);
```

Wszystkie te sposoby korzystają ze zbliżonej ilości miejsca w pamięci.
Zapis pierwszy jest najkrótszy i najbardziej odporny na literówki.
Zapis pierwszy jest najszybszy w wykonaniu.
Statyczna tablica globalna jest wypełniona 0.

Operator ternarny

```
char c;
if(a > b) {
    c = 'A';
} else {
    c = 'B';
}

char c = a > b ? 'A' : 'B';
```

Binary Search

Wyszukiwanie binarne służy do znajdowania wartości w posortowanym ciągu.

- Wybierz element środkowy tablicy
- Sprawdź, czy środkowy element jest tym szukanym
- Jeśli tak, to koniec poszukiwań
- Jeśli nie, to czy jest on większy od szukanego
- Jeśli tak, to szukamy w lewej połowie
- Jeśli nie, to szukamy w prawej połowie

Zadanie

W tablicy wszystkie elementy są unikalne i posortowane.
Wartości w tablicy są z przedziału 1 do N, gdzie N to rozmiar tablicy + 1.
Brakuje w niej jednego elementu.
Zadaniem jest znalezienie brakującego elementu.
Przykład:
Input: $0, 1, 2, 3, 4, 5, 7, 8, 9$
Output: 6

Kod w C++

```
arturlaskowski@lab-seq-2 ~/Documents/github/WdA-lectures cat binary_search.cpp
#include <iostream>

using namespace std;

int tab[1000];

int bin_search(int beg, int end) {
        cout << "Szukam w przedziale (" << beg << ", " << end << ")" << endl;
        if(tab[beg] != beg) {
                cout << "Mam go: " << beg << endl;
                return beg;
        }
        int mid = (beg + end) / 2;
        if(tab[mid] != mid) {
                return bin_search(beg, mid);
        } else {
                return bin_search(mid + 1, end);
        }
}

int main() {
        for(int i = 0, j = 0; i < 1000; ++i) {
                if(i == 33) {
                        ++j;
                }
                tab[i] = j++;
        }

        cout << bin_search(0, 999) << endl;

        return 0;
}
arturlaskowski@lab-seq-2 ~/Documents/github/WdA-lectures g++ binary_search.cpp -o C.out
arturlaskowski@lab-seq-2 ~/Documents/github/WdA-lectures ./C.out
Szukam w przedziale (0, 999)
Szukam w przedziale (0, 499)
Szukam w przedziale (0, 249)
Szukam w przedziale (0, 124)
Szukam w przedziale (0, 62)
Szukam w przedziale (32, 62)
Szukam w przedziale (32, 47)
Szukam w przedziale (32, 39)
Szukam w przedziale (32, 35)
Szukam w przedziale (32, 33)
Szukam w przedziale (33, 33)
Mam go: 33
33
arturlaskowski@lab-seq-2 ~/Documents/github/WdA-lectures
```

Obserwacja

Jeśli dane trzeba wcześniej wczytać.
Można to zrobić w trakcie wczytywania, bez dodatkowego wyszukiwania.

```
 arturlaskowski@lab-seq-2 ~/Documents/github/WdA-lectures cat binary_search_skipped.cpp
#include <iostream>

using namespace std;

int arr[10];

int main() {
        for(int i = 0; i < 10; ++i) {
                cin >> arr[i];
                if(arr[i] != i) {
                        cout << i << endl;
                        return 0;
                }
        }
}
 arturlaskowski@lab-seq-2 ~/Documents/github/WdA-lectures
```

Liczby pierwsze

Liczba pierwsza to taka liczba, którą dzielą dokładnie dwie liczby: 1 oraz ona sama.

$$2, 3, 5, 7, 11, 13, ...$$

Rozkład liczby na czynniki pierwsze

$$x = p_1^{n_1} \cdot p_2^{n_2} \cdot \ldots \cdot p_q^{n_q}$$

Czynnik *pi* wystepuje n_i razy w rozkładzie liczby x.
Liczba wszystkich dzielników liczby x:

$$y = (n_1 + 1) \cdot (n_2 + 1) \cdot \ldots \cdot (n_q + 1)$$

Przykład:

$$12 = 2^2 \cdot 3^1$$

$$(2 + 1) \cdot (1 + 1) = 3 \cdot 2 = 6$$

Dzielniki:

$$1, 2, 3, 4, 6, 12$$

Spostrzeżenie

$$n = p \cdot q$$

$$min(p, q) \leqslant \sqrt{n}$$

Test pierwszości

```
bool is_prime(int n) {
    if(n < 2) {
        return false;
    }
    for(int i = 2; i * i <= n; ++i) {
        if(n % i == 0) {
            return false;
        }
    }
    return true;
}
```

Eratostenes

```
void sito() {
    for(int i = 0; i <= MAX; ++i) {
        prime[i] = true;
    }
    prime[0] = prime[1] = false;
    for(int i = 2; i * i <= MAX; ++i) {
        if(prime[i]) {
            j = i * i;
            while(j <= MAX) {
                prime[j] = false;
                j += i;
            }
        }
    }
}
```

Zadanie

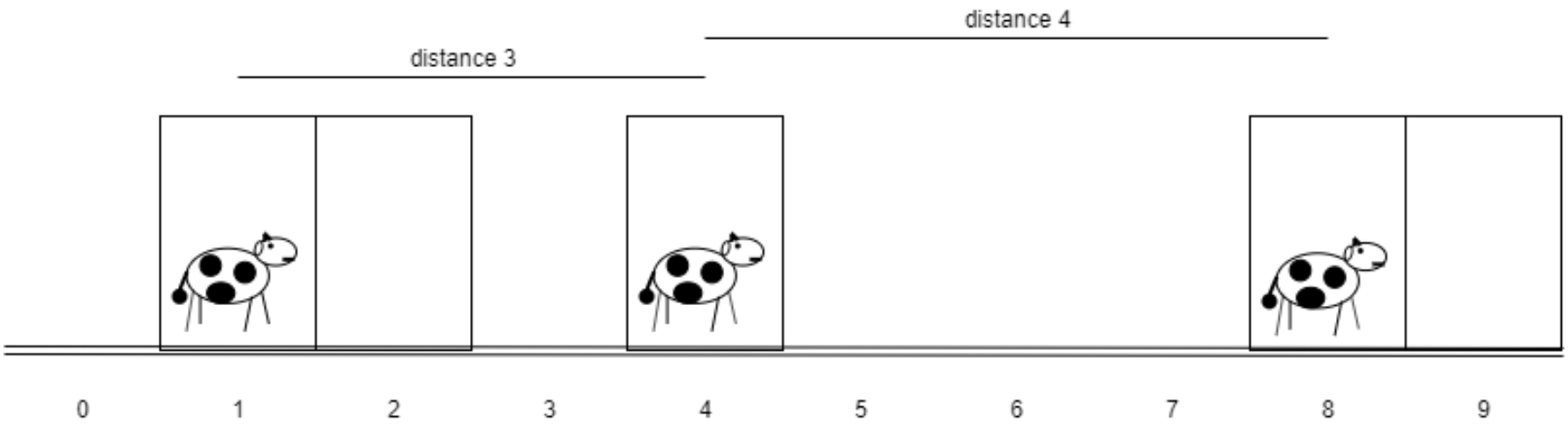
Farmer Jan wybudował długą oborę, która ma N ($2 \leqslant N \leqslant 100,000$) stanowisk. Wszystkie stanowiska są ustawione w linii na pozycjach $x_1, ..., x_N$ ($0 \leqslant x_i \leqslant 1,000,000,000$).
Posiada on C ($2 \leqslant C \leqslant N$) krów, które nie są agresywne wobec siebie.
Farmer chce ustawić krowy w taki sposób, żeby zminimalizować szansę na to, że zrobią sobie krzywdę.
Jan chce, aby dystans pomiędzy krowami, które są najbliżej siebie był jak największy.
Jaki największy dystans może w ten sposób uzyskać?

Zadanie

Przykładowy test.

```
5 3

1

2

8

4

9
```

Zadanie



Laboratoria

https://www.hackerrank.com/wda-01-2021