Kezdés ideje	2020. június 18., csütörtök, 13:17
Állapot	Befejezte
Befejezés dátuma	2020. június 18., csütörtök, 14:45
Felhasznált idő	1 óra 27 perc
Pont	**5** a maximum 5 közül (**100**%)

1 kérdés

Helyes

1 közül 1 leosztályozva

Írjon egy `maxoszlop` nevű függvényt, ami paraméterként kap egy dinamikusan, soronként foglalt int mátrixot (a sorok kezdőcímeinek tömbjét), a sorok számát, és oszlopok számát; és visszaadja annak az oszlopnak az indexét, amelyikben az elemek összege a legnagyobb. Ha több maximális összegű oszlop is van, ezek közül a legnagyobb indexű oszlop indexét adja vissza.

Megjegyzés: A tesztekben található letrehoz, feltolt, és felszabadit függvényeket nem kell megírnia.

Szintaktikai hiba esetén a fordító által jelzett sor számából vonjon ki 40-et, hogy megkapja a szerkesztőben olvasható sorszámot!

For example:

Test	Result

`int sorok = 4, oszlopok = 5;`
`int **mtx = letrehoz(sorok, oszlopok);`
`feltolt(mtx, sorok, oszlopok, 0);`
`cout << "A legnagyobb osszegu oszlop indexe: " << maxoszlop(mtx, sorok, oszlopok) << "\n";`
`felszabadit(mtx, sorok);`
	`A legnagyobb osszegu oszlop indexe: 2`

Answer: (penalty regime: 0 %)

Reset answer

```
#include <climits>

int maxoszlop(int **mtx, int s, int o) {
    int maxsum =INT_MIN, max=1;
    for(int i=0 ;i<o;i++){
        int sum =0;
        for(int j=0;j<s;j++){
            sum += mtx[j][i];
        }
        if(sum>maxsum){
            maxsum=sum;
            max=i;
        }
    }
    return max;
}
```

	Test	Expected	Got	

✔	`int sorok = 4, oszlopok = 5;`
	`int **mtx = letrehoz(sorok, oszlopok);`
	`feltolt(mtx, sorok, oszlopok, 0);`
	`cout << "A legnagyobb osszegu oszlop indexe: " << maxoszlop(mtx, sorok, oszlopok) << "\n";`
	`felszabadit(mtx, sorok);`
		A legnagyobb osszegu oszlop indexe: 2	A legnagyobb osszegu oszlop indexe: 2	✔

✔	`int sorok = 16, oszlopok = 8;`
	`int **mtx = letrehoz(sorok, oszlopok);`
	`feltolt(mtx, sorok, oszlopok, 4);`
	`cout << "A legnagyobb osszegu oszlop indexe: " << maxoszlop(mtx, sorok, oszlopok) << "\n";`
	`felszabadit(mtx, sorok);`
		A legnagyobb osszegu oszlop indexe: 2	A legnagyobb osszegu oszlop indexe: 2	✔

✔	`int sorok = 10, oszlopok = 15;`
	`int **mtx = letrehoz(sorok, oszlopok);`
	`feltolt(mtx, sorok, oszlopok, 3);`
	`cout << "A legnagyobb osszegu oszlop indexe: " << maxoszlop(mtx, sorok, oszlopok) << "\n";`
	`felszabadit(mtx, sorok);`
		A legnagyobb osszegu oszlop indexe: 13	A legnagyobb osszegu oszlop indexe: 13	✔

✔	`int sorok = 6, oszlopok = 5;`
	`int **mtx = letrehoz(sorok, oszlopok);`
	`feltolt(mtx, sorok, oszlopok, 2);`
	`cout << "A legnagyobb osszegu oszlop indexe: " << maxoszlop(mtx, sorok, oszlopok) << "\n";`
	`felszabadit(mtx, sorok);`
		A legnagyobb osszegu oszlop indexe: 3	A legnagyobb osszegu oszlop indexe: 3	✔

Passed all tests! ✔

Question author's solution:

```
#include <climits>

int maxoszlop(int **mtx, int s, int o) {
    int maxsum = 0, maxi = 0;
    for (int i=0; i<o; i++) {
        int sum = 0;
        for(int j=0; j<s; j++) {
            sum += mtx[j][i];
        }
        if (sum > maxsum) {
            maxsum = sum;
            maxi = i;
        }
    }
    return maxi;
}
```

Helyes

Leadásra bejelöl: 1/1.

2 kérdés

Helyes

Adott a következő struktúra, amely országot, az ország Nobel-dijasainak számát és a lakosságot (millió főben) tartalmazza:

1 közül 1 leosztályozva

```
struct NobelDij{
 string country;
 unsigned int numWinnners;
 double millionPeople;
}
```

Írjon meg két függvényt:

- `void sortByRate(bool rate, NobelDij *nobels, int n)` - a rendezést valósítja meg, ha `rate` `true`, akkor Nobel díj / millió lakos, ha `false` akkor pedig a Nobel dij alapján csökkenő sorrendbe
- `void statistics(NobelDij *nobels, int n)` - írja ki orszagonként átlagosan hány millió lakos és átlagosan mennyi a Nobel-díjas szam

Szintaktikai hiba esetén a fordító által jelzett sor számából vonjon ki 40-at, hogy megkapja a szerkesztőben olvasható sorszámot!

For example:

Test	Result

`NobelDij nobels[] = {`
`{"AUS", 12, 24.4},`
`{"BEL", 10, 11.42},`
`{"CZE", 5, 10.6},`
`{"DNK", 13, 5.73},`
`{"FRA", 68, 64.97},`
`{"GRE", 2, 11.1},`
`{"HUN", 13, 9.7},`
`{"GER", 107, 82.11},`
`{"SPA", 8, 46.35},`
`{"ROM", 4, 19.67},`
`{"CHE", 26, 8.47},`
`{"SLN", 1, 2.},`
`{"SLK", 0, 5.45},`
`{"USA", 368, 324.45},`
`{"UKR", 2, 44.222},`
`};`
`sortByRate(true, nobels, 15);`
`printAll(nobels, 15);`
`statistics(nobels, 15);`
	`1. hely: CHE (3.07 Nobel dij / millio lakos) [Osszesen 26 Nobel]`
	`2. hely: DNK (2.27 Nobel dij / millio lakos) [Osszesen 13 Nobel]`
	`3. hely: HUN (1.34 Nobel dij / millio lakos) [Osszesen 13 Nobel]`
	`4. hely: GER (1.30 Nobel dij / millio lakos) [Osszesen 107 Nobel]`
	`5. hely: USA (1.13 Nobel dij / millio lakos) [Osszesen 368 Nobel]`
	`6. hely: FRA (1.05 Nobel dij / millio lakos) [Osszesen 68 Nobel]`
	`7. hely: BEL (0.88 Nobel dij / millio lakos) [Osszesen 10 Nobel]`
	`8. hely: SLN (0.50 Nobel dij / millio lakos) [Osszesen 1 Nobel]`
	`9. hely: AUS (0.49 Nobel dij / millio lakos) [Osszesen 12 Nobel]`
	`10. hely: CZE (0.47 Nobel dij / millio lakos) [Osszesen 5 Nobel]`
	`11. hely: ROM (0.20 Nobel dij / millio lakos) [Osszesen 4 Nobel]`
	`12. hely: GRE (0.18 Nobel dij / millio lakos) [Osszesen 2 Nobel]`
	`13. hely: SPA (0.17 Nobel dij / millio lakos) [Osszesen 8 Nobel]`
	`14. hely: UKR (0.05 Nobel dij / millio lakos) [Osszesen 2 Nobel]`
	`15. hely: SLK (0.00 Nobel dij / millio lakos) [Osszesen 0 Nobel]`
	`Orszagonkent atlagosan: 44.71 millio lakos`
	`Atlagos Nobel-dijas szam orszagonkent: 2.84`

Answer: (penalty regime: 0 %)

Reset answer

```
1  void sortByRate(bool rate, NobelDij *nobels, int n)
2  {
3      for (int i = 0; i < n-1; i++)
4          for (int j = i + 1; j < n; j++)
5          {
6              if (rate)
7              {
8                  if ((double)nobels[i].numWinnners / nobe
9                  {
10                     NobelDij seg = nobels[i];
11                     nobels[i] = nobels[j];
12                     nobels[j] = seg;
13                 }
14             }
15             else
16             {
17                 if (nobels[i].numWinnners < nobels[j].nu
18                 {
19                     NobelDij seg = nobels[i];
20                     nobels[i] = nobels[j];
21                     nobels[j] = seg;
22
```

	Test	Input	Expected	Got	

✔	NobelDij nobels[] = {
	
	{"Ausztralia", 12, 24.4},
	{"Belgium", 10, 11.42},
	
	{"Csehorszag", 5, 10.6},
	{"Dania", 13, 5.73},
	
	{"Franciaorszag", 68, 64.97},
	
	{"Gorogorszag", 2, 11.1},
	
	{"Magyaroszag", 13, 9.7},
	
	{"Nemetorszag", 107, 82.11},
	
	{"Spanyolorszag", 8, 46.35},
	
	{"Romania", 4, 19.67},
	{"Svajc", 26, 8.47},
	
	{"Szlovenia", 1, 2.},
	
	{"Szlovakia", 0, 5.45},
	{"USA", 368, 324.45},
	
	{"Ukrajna", 2, 44.222},
	};
	sortByRate(true, nobels, 15);
	printAll(nobels, 15);
			1. hely: Svajc (3.07 Nobel dij / millio lakos) [Osszesen 26 Nobel]
			2. hely: Dania (2.27 Nobel dij / millio lakos) [Osszesen 13 Nobel]
			3. hely: Magyaroszag (1.34 Nobel dij / millio lakos) [Osszesen 13 Nobel]
			4. hely: Nemetorszag (1.30 Nobel dij / millio lakos) [Osszesen 107 Nobel]
			5. hely: USA (1.13 Nobel dij / millio lakos) [Osszesen 368 Nobel]
			6. hely: Franciaorszag (1.05 Nobel dij / millio lakos) [Osszesen 68 Nobel]
			7. hely: Belgium (0.88 Nobel dij / millio lakos) [Osszesen 10 Nobel]
			8. hely: Szlovenia (0.50 Nobel dij / millio lakos) [Osszesen 1 Nobel]
			9. hely: Ausztralia (0.49 Nobel dij / millio lakos) [Osszesen 12 Nobel]
			10. hely: Csehorszag (0.47 Nobel dij / millio lakos) [Osszesen 5 Nobel]
			11. hely: Romania (0.20 Nobel dij / millio lakos) [Osszesen 4 Nobel]
			12. hely: Gorogorszag (0.18 Nobel dij / millio lakos) [Osszesen 2 Nobel]
			13. hely: Spanyolorszag (0.17 Nobel dij / millio lakos) [Osszesen 8 Nobel]
			14. hely: Ukrajna (0.05 Nobel dij / millio lakos) [Osszesen 2 Nobel]
			15. hely: Szlovakia (0.00 Nobel dij / millio lakos) [Osszesen 0 Nobel]
				1. hely: Svajc (3.07 Nobel dij / millio lakos) [Osszesen 26 Nobel]
				2. hely: Dania (2.27 Nobel dij / millio lakos) [Osszesen 13 Nobel]
				3. hely: Magyaroszag (1.34 Nobel dij / millio lakos) [Osszesen 13 Nobel]
				4. hely: Nemetorszag (1.30 Nobel dij / millio lakos) [Osszesen 107 Nobel]
				5. hely: USA (1.13 Nobel dij / millio lakos) [Osszesen 368 Nobel]
				6. hely: Franciaorszag (1.05 Nobel dij / millio lakos) [Osszesen 68 Nobel]
				7. hely: Belgium (0.88 Nobel dij / millio lakos) [Osszesen 10 Nobel]
				8. hely: Szlovenia (0.50 Nobel dij / millio lakos) [Osszesen 1 Nobel]
				9. hely: Ausztralia (0.49 Nobel dij / millio lakos) [Osszesen 12 Nobel]
				10. hely: Csehorszag (0.47 Nobel dij / millio lakos) [Osszesen 5 Nobel]
				11. hely: Romania (0.20 Nobel dij / millio lakos) [Osszesen 4 Nobel]
				12. hely: Gorogorszag (0.18 Nobel dij / millio lakos) [Osszesen 2 Nobel]
				13. hely: Spanyolorszag (0.17 Nobel dij / millio lakos) [Osszesen 8 Nobel]
				14. hely: Ukrajna (0.05 Nobel dij / millio lakos) [Osszesen 2 Nobel]
				15. hely: Szlovakia (0.00 Nobel dij / millio lakos) [Osszesen 0 Nobel]
					✔

✔	NobelDij nobels[] = {
	{"AUS", 12, 24.4},
	{"BEL", 10, 11.42},
	{"CZE", 5, 10.6},
	{"DNK", 13, 5.73},
	{"FRA", 68, 64.97},
	{"GRE", 2, 11.1},
	{"HUN", 13, 9.7},
	{"GER", 107, 82.11},
	{"SPA", 8, 46.35},
	{"ROM", 4, 19.67},
	{"CHE", 26, 8.47},
	{"SLN", 1, 2.},
	{"SLK", 0, 5.45},
	{"USA", 368, 324.45},
	{"UKR", 2, 44.222},
	};
	sortByRate(true, nobels, 15);
	printAll(nobels, 15);
	statistics(nobels, 15);
			1. hely: CHE (3.07 Nobel dij / millio lakos) [Osszesen 26 Nobel]
			2. hely: DNK (2.27 Nobel dij / millio lakos) [Osszesen 13 Nobel]
			3. hely: HUN (1.34 Nobel dij / millio lakos) [Osszesen 13 Nobel]
			4. hely: GER (1.30 Nobel dij / millio lakos) [Osszesen 107 Nobel]
			5. hely: USA (1.13 Nobel dij / millio lakos) [Osszesen 368 Nobel]
			6. hely: FRA (1.05 Nobel dij / millio lakos) [Osszesen 68 Nobel]
			7. hely: BEL (0.88 Nobel dij / millio lakos) [Osszesen 10 Nobel]
			8. hely: SLN (0.50 Nobel dij / millio lakos) [Osszesen 1 Nobel]
			9. hely: AUS (0.49 Nobel dij / millio lakos) [Osszesen 12 Nobel]
			10. hely: CZE (0.47 Nobel dij / millio lakos) [Osszesen 5 Nobel]
			11. hely: ROM (0.20 Nobel dij / millio lakos) [Osszesen 4 Nobel]
			12. hely: GRE (0.18 Nobel dij / millio lakos) [Osszesen 2 Nobel]
			13. hely: SPA (0.17 Nobel dij / millio lakos) [Osszesen 8 Nobel]
			14. hely: UKR (0.05 Nobel dij / millio lakos) [Osszesen 2 Nobel]
			15. hely: SLK (0.00 Nobel dij / millio lakos) [Osszesen 0 Nobel]
			Orszagonkent atlagosan: 44.71 millio lakos
			Atlagos Nobel-dijas szam orszagonkent: 2.84
				1. hely: CHE (3.07 Nobel dij / millio lakos) [Osszesen 26 Nobel]
				2. hely: DNK (2.27 Nobel dij / millio lakos) [Osszesen 13 Nobel]
				3. hely: HUN (1.34 Nobel dij / millio lakos) [Osszesen 13 Nobel]
				4. hely: GER (1.30 Nobel dij / millio lakos) [Osszesen 107 Nobel]
				5. hely: USA (1.13 Nobel dij / millio lakos) [Osszesen 368 Nobel]
				6. hely: FRA (1.05 Nobel dij / millio lakos) [Osszesen 68 Nobel]
				7. hely: BEL (0.88 Nobel dij / millio lakos) [Osszesen 10 Nobel]
				8. hely: SLN (0.50 Nobel dij / millio lakos) [Osszesen 1 Nobel]
				9. hely: AUS (0.49 Nobel dij / millio lakos) [Osszesen 12 Nobel]
				10. hely: CZE (0.47 Nobel dij / millio lakos) [Osszesen 5 Nobel]
				11. hely: ROM (0.20 Nobel dij / millio lakos) [Osszesen 4 Nobel]
				12. hely: GRE (0.18 Nobel dij / millio lakos) [Osszesen 2 Nobel]
				13. hely: SPA (0.17 Nobel dij / millio lakos) [Osszesen 8 Nobel]
				14. hely: UKR (0.05 Nobel dij / millio lakos) [Osszesen 2 Nobel]
				15. hely: SLK (0.00 Nobel dij / millio lakos) [Osszesen 0 Nobel]
				Orszagonkent atlagosan: 44.71 millio lakos
				Atlagos Nobel-dijas szam orszagonkent: 2.84
					✔

✔	NobelDij nobels[] = {		1. hely: USA (1.13	1. hely: USA (1.13	✔

	= {
	{"AUS", 12, 24.4},
	{"BEL ", 10, 11.42},
	{"CZE", 5, 10.6},
	{"DNK", 13, 5.73},
	{"FRA", 68, 64.97},
	{"GRE", 2, 11.1},
	{"HUN", 13, 9.7},
	{"GER", 107, 82.11},
	{"SPA", 8, 46.35},
	{"ROM", 4, 19.67},
	{"CHE", 26, 8.47},
	{"SLN", 1, 2.},
	{"SLK", 0, 5.45},
	{"USA", 368, 324.45},
	{"UKR", 2, 44.222},
	};
	sortByRate(false, nobels, 15);
	printAll(nobels, 15);
	statistics(nobels, 15);
			Nobel dij / millio lakos) [Osszesen 368 Nobel]
			2. hely: GER (1.30 Nobel dij / millio lakos) [Osszesen 107 Nobel]
			3. hely: FRA (1.05 Nobel dij / millio lakos) [Osszesen 68 Nobel]
			4. hely: CHE (3.07 Nobel dij / millio lakos) [Osszesen 26 Nobel]
			5. hely: HUN (1.34 Nobel dij / millio lakos) [Osszesen 13 Nobel]
			6. hely: DNK (2.27 Nobel dij / millio lakos) [Osszesen 13 Nobel]
			7. hely: AUS (0.49 Nobel dij / millio lakos) [Osszesen 12 Nobel]
			8. hely: BEL (0.88 Nobel dij / millio lakos) [Osszesen 10 Nobel]
			9. hely: SPA (0.17 Nobel dij / millio lakos) [Osszesen 8 Nobel]
			10. hely: CZE (0.47 Nobel dij / millio lakos) [Osszesen 5 Nobel]
			11. hely: ROM (0.20 Nobel dij / millio lakos) [Osszesen 4 Nobel]
			12. hely: GRE (0.18 Nobel dij / millio lakos) [Osszesen 2 Nobel]
			13. hely: UKR (0.05 Nobel dij / millio lakos) [Osszesen 2 Nobel]
			14. hely: SLN (0.50 Nobel dij / millio lakos) [Osszesen 1 Nobel]
			15. hely: SLK (0.00 Nobel dij / millio lakos) [Osszesen 0 Nobel]
			Orszagonkent atlagosan: 44.71 millio lakos
			Atlagos Nobel-dijas szam orszagonkent: 2.84
				Nobel dij / millio lakos) [Osszesen 368 Nobel]
				2. hely: GER (1.30 Nobel dij / millio lakos) [Osszesen 107 Nobel]
				3. hely: FRA (1.05 Nobel dij / millio lakos) [Osszesen 68 Nobel]
				4. hely: CHE (3.07 Nobel dij / millio lakos) [Osszesen 26 Nobel]
				5. hely: HUN (1.34 Nobel dij / millio lakos) [Osszesen 13 Nobel]
				6. hely: DNK (2.27 Nobel dij / millio lakos) [Osszesen 13 Nobel]
				7. hely: AUS (0.49 Nobel dij / millio lakos) [Osszesen 12 Nobel]
				8. hely: BEL (0.88 Nobel dij / millio lakos) [Osszesen 10 Nobel]
				9. hely: SPA (0.17 Nobel dij / millio lakos) [Osszesen 8 Nobel]
				10. hely: CZE (0.47 Nobel dij / millio lakos) [Osszesen 5 Nobel]
				11. hely: ROM (0.20 Nobel dij / millio lakos) [Osszesen 4 Nobel]
				12. hely: GRE (0.18 Nobel dij / millio lakos) [Osszesen 2 Nobel]
				13. hely: UKR (0.05 Nobel dij / millio lakos) [Osszesen 2 Nobel]
				14. hely: SLN (0.50 Nobel dij / millio lakos) [Osszesen 1 Nobel]
				15. hely: SLK (0.00 Nobel dij / millio lakos) [Osszesen 0 Nobel]
				Orszagonkent atlagosan: 44.71 millio lakos
				Atlagos Nobel-dijas szam orszagonkent: 2.84	

✔	NobelDij nobels[] = { {"AUS", 12,		1. hely: USA (1.13 Nobel dij / millio lakos) [Osszesen 368	1. hely: USA (1.13 Nobel dij / millio lakos) [Osszesen 368	✔

	24.4},
	{"BEL", 10, 11.42},
	{"CZE", 5, 10.6},
	{"DNK", 13, 5.73},
	{"FRA", 68, 64.97},
	{"GRE", 2, 11.1},
	{"HUN", 13, 9.7},
	{"GER", 107, 82.11},
	{"SPA", 8, 46.35},
	{"ROM", 4, 19.67},
	{"CHE", 26, 8.47},
	{"SLN", 1, 2.},
	{"SLK", 0, 5.45},
	{"USA", 368, 324.45},
	{"UKR", 2, 44.222},
	};
	sortByRate(false, nobels, 15);
	printAll(nobels, 15);
	statistics(nobels, 15);
			Nobel]
			2. hely: GER (1.30 Nobel dij / millio lakos) [Osszesen 107 Nobel]
			3. hely: FRA (1.05 Nobel dij / millio lakos) [Osszesen 68 Nobel]
			4. hely: CHE (3.07 Nobel dij / millio lakos) [Osszesen 26 Nobel]
			5. hely: HUN (1.34 Nobel dij / millio lakos) [Osszesen 13 Nobel]
			6. hely: DNK (2.27 Nobel dij / millio lakos) [Osszesen 13 Nobel]
			7. hely: AUS (0.49 Nobel dij / millio lakos) [Osszesen 12 Nobel]
			8. hely: BEL (0.88 Nobel dij / millio lakos) [Osszesen 10 Nobel]
			9. hely: SPA (0.17 Nobel dij / millio lakos) [Osszesen 8 Nobel]
			10. hely: CZE (0.47 Nobel dij / millio lakos) [Osszesen 5 Nobel]
			11. hely: ROM (0.20 Nobel dij / millio lakos) [Osszesen 4 Nobel]
			12. hely: GRE (0.18 Nobel dij / millio lakos) [Osszesen 2 Nobel]
			13. hely: UKR (0.05 Nobel dij / millio lakos) [Osszesen 2 Nobel]
			14. hely: SLN (0.50 Nobel dij / millio lakos) [Osszesen 1 Nobel]
			15. hely: SLK (0.00 Nobel dij / millio lakos) [Osszesen 0 Nobel]
			Orszagonkent atlagosan: 44.71 millio lakos
			Atlagos Nobel-dijas szam orszagonkent: 2.84
				Nobel]
				2. hely: GER (1.30 Nobel dij / millio lakos) [Osszesen 107 Nobel]
				3. hely: FRA (1.05 Nobel dij / millio lakos) [Osszesen 68 Nobel]
				4. hely: CHE (3.07 Nobel dij / millio lakos) [Osszesen 26 Nobel]
				5. hely: HUN (1.34 Nobel dij / millio lakos) [Osszesen 13 Nobel]
				6. hely: DNK (2.27 Nobel dij / millio lakos) [Osszesen 13 Nobel]
				7. hely: AUS (0.49 Nobel dij / millio lakos) [Osszesen 12 Nobel]
				8. hely: BEL (0.88 Nobel dij / millio lakos) [Osszesen 10 Nobel]
				9. hely: SPA (0.17 Nobel dij / millio lakos) [Osszesen 8 Nobel]
				10. hely: CZE (0.47 Nobel dij / millio lakos) [Osszesen 5 Nobel]
				11. hely: ROM (0.20 Nobel dij / millio lakos) [Osszesen 4 Nobel]
				12. hely: GRE (0.18 Nobel dij / millio lakos) [Osszesen 2 Nobel]
				13. hely: UKR (0.05 Nobel dij / millio lakos) [Osszesen 2 Nobel]
				14. hely: SLN (0.50 Nobel dij / millio lakos) [Osszesen 1 Nobel]
				15. hely: SLK (0.00 Nobel dij / millio lakos) [Osszesen 0 Nobel]
				Orszagonkent atlagosan: 44.71 millio lakos
				Atlagos Nobel-dijas szam orszagonkent: 2.84	

✔	NobelDij nobels[] = { {"CZE", 5, 10.6}, {"FRA", 68,	NobelDij nobels[] = { {"CZE", 5, 10.6}, {"FRA", 68,	1. hely: CHE (3.07 Nobel dij / millio lakos) [Osszesen 26 Nobel] 2. hely: HUN (1.34	1. hely: CHE (3.07 Nobel dij / millio lakos) [Osszesen 26 Nobel] 2. hely: HUN (1.34	✔

	64.97},
	{"HUN", 13, 9.7},
	{"GER", 107, 82.11},
	{"SPA", 8, 46.35},
	{"ROM", 4, 19.67},
	{"CHE", 26, 8.47},
	{"SLN", 1, 2.},
	{"SLK", 0, 5.45},
	{"UKR", 2, 44.222},
	};
	sortByRate(true, nobels, 10);
	printAll(nobels, 10);
	statistics(nobels, 10);
		64.97},
		{"HUN", 13, 9.7},
		{"GER", 107, 82.11},
		{"SPA", 8, 46.35},
		{"ROM", 4, 19.67},
		{"CHE", 26, 8.47},
		{"SLN", 1, 2.},
		{"SLK", 0, 5.45},
		{"UKR", 2, 44.222},
		};
		sortByRate(false, nobels, 10);
		printAll(nobels, 10);
		statistics(nobels, 10);
			Nobel dij / millio lakos) [Osszesen 13 Nobel]
			3. hely: GER (1.30 Nobel dij / millio lakos) [Osszesen 107 Nobel]
			4. hely: FRA (1.05 Nobel dij / millio lakos) [Osszesen 68 Nobel]
			5. hely: SLN (0.50 Nobel dij / millio lakos) [Osszesen 1 Nobel]
			6. hely: CZE (0.47 Nobel dij / millio lakos) [Osszesen 5 Nobel]
			7. hely: ROM (0.20 Nobel dij / millio lakos) [Osszesen 4 Nobel]
			8. hely: SPA (0.17 Nobel dij / millio lakos) [Osszesen 8 Nobel]
			9. hely: UKR (0.05 Nobel dij / millio lakos) [Osszesen 2 Nobel]
			10. hely: SLK (0.00 Nobel dij / millio lakos) [Osszesen 0 Nobel]
			Orszagonkent atlagosan: 29.35 millio lakos
			Atlagos Nobel-dijas szam orszagonkent: 2.34
				Nobel dij / millio lakos) [Osszesen 13 Nobel]
				3. hely: GER (1.30 Nobel dij / millio lakos) [Osszesen 107 Nobel]
				4. hely: FRA (1.05 Nobel dij / millio lakos) [Osszesen 68 Nobel]
				5. hely: SLN (0.50 Nobel dij / millio lakos) [Osszesen 1 Nobel]
				6. hely: CZE (0.47 Nobel dij / millio lakos) [Osszesen 5 Nobel]
				7. hely: ROM (0.20 Nobel dij / millio lakos) [Osszesen 4 Nobel]
				8. hely: SPA (0.17 Nobel dij / millio lakos) [Osszesen 8 Nobel]
				9. hely: UKR (0.05 Nobel dij / millio lakos) [Osszesen 2 Nobel]
				10. hely: SLK (0.00 Nobel dij / millio lakos) [Osszesen 0 Nobel]
				Orszagonkent atlagosan: 29.35 millio lakos
				Atlagos Nobel-dijas szam orszagonkent: 2.34	

Passed all tests! ✔

Question author's solution:

```
void sortByRate(bool rate, NobelDij *nobels, int n)
{
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            if (rate){
                if (nobels[i].numWinnners / nobels[i].millionPeople > nobels[j].numWinnners /
nobels[j].millionPeople)
                {
                    NobelDij t = nobels[i];
                    nobels[i] = nobels[j];
                    nobels[j] = t;
                }
            }
            else{
                if (nobels[i].numWinnners > nobels[j].numWinnners)
                {
                    NobelDij t = nobels[i];
                    nobels[i] = nobels[j];
                    nobels[j] = t;
                }
```

```
        }
    }
}

void printAll(NobelDij *nobels, int n)
{
    for (int i = 0; i < n; i++)
    {
        cout << i+1 << ". hely: "<< nobels[i].country << " (" << fixed << setprecision(2) <<
nobels[i].numWinnners / nobels[i].millionPeople << " Nobel dij / millio lakos) [Osszesen " <<
nobels[i].numWinnners << " Nobel]\n";
    }
}

void statistics(NobelDij *nobels, int n)
{
    double meanN = 0.0, meanM = 0.0;
    for (int i = 0; i < n; i++)
    {
        meanN += nobels[i].numWinnners;
        meanM += nobels[i].millionPeople;
    }
    meanN /= n;
    cout << "Orszagonkent atlagosan: " << meanM / n << " millio lakos\n";
    cout << "Atlagos Nobel-dijas szam orszagonkent: " << meanN / n << "\n";
}
```

Helyes

Leadásra bejelöl: 1/1.

3 kérdés

Helyes

1 közül 1 leosztályozva

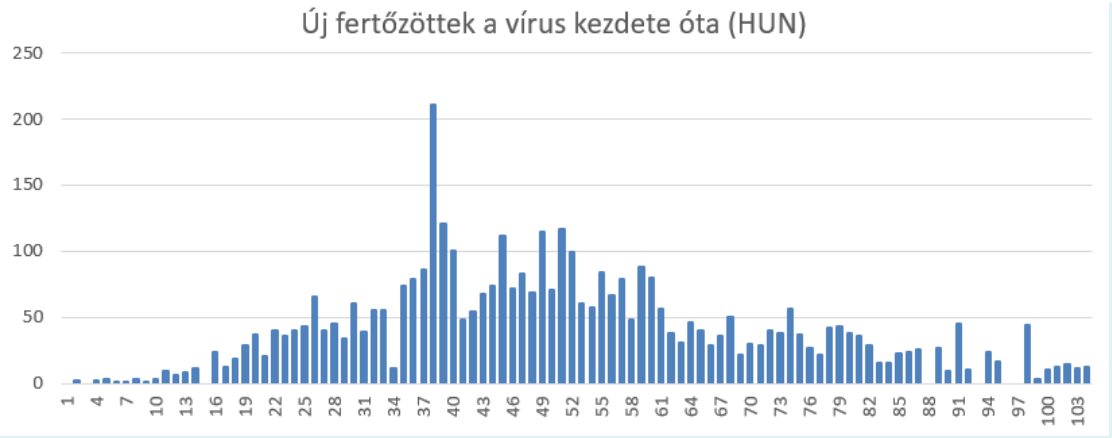
Készítsen függvényeket, amelyekkel fájlból olvas be egész számokat és azokat egy tömbbe illeszti. A számok a Magyarországon regisztrált COVID fertőzötteket jelentik a vírus megjelenésétől kezdve.

Valósítsa meg a 3 rövid függvényt:

- `void fajbolOlvas(string fajlnev, int adatok[], int db)` - Beolvassa *fajlnev*-ből az *adatok*-ba az *n* darabszámú adatot. A *fajlnev* nevű fájlnak legfeljebb az első *n* sorát, és az abban található adatokat helyezze el az adatok tömbben. Tesztesetként csak olyan eseteket adunk, ahol a fájl nagyobb vagy egyenlő *n*.
- `double atlag(const int adatok[], int db)` - Visszaadja a tömb átlag értékét.
- `double osszes(const int adatok[], int db)` - Visszaadja a tömb összértékét.

Például a `covid19.txt` tartalma:

2
0
2
3
1
1
3
1
3
9
6
8
11
0
23
12
....stb



Szintaktikai hiba esetén a fordító által jelzett sor számából vonjon ki 20-at, hogy megkapja a szerkesztőben olvasható sorszámot!

Answer: (penalty regime: 0 %)

Reset answer

```
1  void fajbolOlvas(string fajlnev, int adatok[], int db)
2  {
3      ifstream fajl(fajlnev.c_str());
4      for (int i = 0; i < db; i++)
5      {
6          string seg;
7          getline(fajl, seg);
8          adatok[i] = atoi(seg.c_str());
9      }
10 }
11
12 double atlag(const int adatok[], int db)
13 {
14     double atlag = 0;
15     for (int i = 0; i < db; i++)
16         atlag += (double)adatok[i];
17     atlag = atlag / db;
18     return atlag;
19 }
20
21 double osszes(const int adatok[], int db)
22 {
```

	Test	Expected	Got	

✔	`int s = 10;`
	`int napiFertozott[s];`
	`fajbolOlvas("covid19.txt", napiFertozott, s);`
	`kilistaz(napiFertozott, s);`
	`cout << "Atlag: " << atlag(napiFertozott, s) << "\n";`
	`cout << "Osszes: " << osszes(napiFertozott, s) << "\n";`
		0. nap 2 uj fertozott
		1. nap 0 uj fertozott
		2. nap 2 uj fertozott
		3. nap 3 uj fertozott
		4. nap 1 uj fertozott
		5. nap 1 uj fertozott
		6. nap 3 uj fertozott
		7. nap 1 uj fertozott
		8. nap 3 uj fertozott
		9. nap 9 uj fertozott
			0. nap 2 uj fertozott
			1. nap 0 uj fertozott
			2. nap 2 uj fertozott
			3. nap 3 uj fertozott
			4. nap 1 uj fertozott
			5. nap 1 uj fertozott
			6. nap 3 uj fertozott
			7. nap 1 uj fertozott
			8. nap 3 uj fertozott
			9. nap 9 uj fertozott
				✔

		Atlag: 2.5 Osszes: 25	Atlag: 2.5 Osszes: 25	

✔	int napiFertozott[1];
	fajbolOlvas("covid19.txt", napiFertozott, 1);
	kilistaz(napiFertozott, 1);
	cout << "Atlag: " << atlag(napiFertozott, 1) << "\n";
	cout << "Osszes: " << osszes(napiFertozott, 1) << "\n";
		0. nap 2 uj fertozott
		Atlag: 2
		Osszes: 2
			0. nap 2 uj fertozott
			Atlag: 2
			Osszes: 2
				✔

✔	int s = 2;
	int napiFertozott[s];
	fajbolOlvas("covid19.txt", napiFertozott, s);
	kilistaz(napiFertozott, s);
	cout << "Atlag: " << atlag(napiFertozott, s) << "\n";
	cout << "Osszes: " << osszes(napiFertozott, s) << "\n";
		0. nap 2 uj fertozott
		1. nap 0 uj fertozott
		Atlag: 1
		Osszes: 2
			0. nap 2 uj fertozott
			1. nap 0 uj fertozott
			Atlag: 1
			Osszes: 2
				✔

✔	int s = 3;
	int napiFertozott[s];
	fajbolOlvas("covid19.txt", napiFertozott, s);
	kilistaz(napiFertozott, s);
	cout << "Atlag: " << atlag(napiFertozott, s) << "\n";
	cout << "Osszes: " << osszes(napiFertozott, s) << "\n";
		0. nap 2 uj fertozott
		1. nap 0 uj fertozott
		2. nap 2 uj fertozott
		Atlag: 1.33333
		Osszes: 4
			0. nap 2 uj fertozott
			1. nap 0 uj fertozott
			2. nap 2 uj fertozott
			Atlag: 1.33333
			Osszes: 4
				✔

✔	int s = 4;
	int napiFertozott[s];
	fajbolOlvas("covid19.txt", napiFertozott, s);
	kilistaz(napiFertozott, s);
	cout << "Atlag: " << atlag(napiFertozott, s) << "\n";
	cout << "Osszes: " << osszes(napiFertozott, s) << "\n";
		0. nap 2 uj fertozott
		1. nap 0 uj fertozott
		2. nap 2 uj fertozott
		3. nap 3 uj fertozott
		Atlag: 1.75
		Osszes: 7
			0. nap 2 uj fertozott
			1. nap 0 uj fertozott
			2. nap 2 uj fertozott
			3. nap 3 uj fertozott
			Atlag: 1.75
			Osszes: 7
				✔

✔	int s = 100;
	int napiFertozott[s];
	fajbolOlvas("covid19.txt", napiFertozott, s);
	cout << "Atlag: " << atlag(napiFertozott, s) << "\n";
	cout << "Osszes: " << osszes(napiFertozott, s) << "\n";
		Atlag: 40.39
		Osszes: 4039
			Atlag: 40.39
			Osszes: 4039
				✔

Passed all tests! ✔

Question author's solution:

```
// fájlból olvas
void fajbolOlvas(string fajlnev, int adatok[], int db)
{
    ifstream fajl(fajlnev.c_str());
    string adat;
    int i = 0;
    if (fajl.is_open())
    {
        while (getline(fajl, adat), !fajl.eof())
        {
            if (db > i)
            {
                adatok[i] = atoi(adat.c_str());
            }
            i++;
        }
    }
}
```

```
void kilistaz(const int adatok[], int db)
{
    for (int i = 0; i < db; i++)
    {
        cout << i << ". nap " << adatok[i] << " uj fertozott\n";
    }
}

double atlag(const int adatok[], int db)
{
    double atlag = 0.0;
    for (int i = 0; i < db; i++)
    {
        atlag += adatok[i];
    }
    return (atlag / db);
}

double osszes(const int adatok[], int db)
{
    double osszes = 0.0;
    for (int i = 0; i < db; i++)
    {
        osszes += adatok[i];
    }
    return osszes;
}
```

Helyes

Leadásra bejelöl: 1/1.

4 kérdés

Helyes

1 közül 1 leosztályozva

Készítsen `binTizesbe` függvényt, amely stringet vár bemenetként és decimális számot ad vissza. A bemeneti string egy bináris számsor, ennek megfelelő számot adjon vissza a függvény.

Szintaktikai hiba esetén a fordító által jelzett sor számából vonjon ki 10-et, hogy megkapja a szerkesztőben olvasható sorszámot!

For example:

Test	Result
`cout << binTizesbe("00000001") << "\n";`	`00000001 -- 1`
`cout << binTizesbe("00000010") << "\n";`	`00000010 -- 2`
`cout << binTizesbe("00000100") << "\n";`	`00000100 -- 4`
`cout << binTizesbe("00000101") << "\n";`	`00000101 -- 5`
`cout << binTizesbe("10000001") << "\n";`	`10000001 -- 129`

Answer: (penalty regime: 0 %)

Reset answer

```
int binTizesbe(string n)
{
    cout << n << " -- ";
    int szam = 0;
    int ertek = 1;
    int hossz = n.length();
    for (int i = hossz - 1; i >= 0; i--)
    {
        if (n[i] == '1')
            szam += ertek;
        ertek *= 2;
    }
    return szam;
}
```

	Test	Expected	Got	
✔	`cout << binTizesbe("10101001") << "\n";` `cout << binTizesbe("00000011") << "\n";`	`10101001 -- 169` `00000011 -- 3`	`10101001 -- 169` `00000011 -- 3`	✔

✔	`cout << binTizesbe("00000001") << "\n";`
	`cout << binTizesbe("00000010") << "\n";`
	`cout << binTizesbe("00000100") << "\n";`
	`cout << binTizesbe("00000101") << "\n";`
	`cout << binTizesbe("10000001") << "\n";`
		`00000001 -- 1`
		`00000010 -- 2`
		`00000100 -- 4`
		`00000101 -- 5`
		`10000001 -- 129`
			`00000001 -- 1`
			`00000010 -- 2`
			`00000100 -- 4`
			`00000101 -- 5`
			`10000001 -- 129`
				✔

✔	`cout << binTizesbe("11111111") << "\n";`
	`cout << binTizesbe("11111110") << "\n";`
	`cout << binTizesbe("00000010") << "\n";`
	`cout << binTizesbe("00000100") << "\n";`
		`11111111 -- 255`
		`11111110 -- 254`
		`00000010 -- 2`
		`00000100 -- 4`
			`11111111 -- 255`
			`11111110 -- 254`
			`00000010 -- 2`
			`00000100 -- 4`
				✔

✔	`cout << binTizesbe("00011") << "\n";`
	`cout << binTizesbe("11") << "\n";`
	`cout << binTizesbe("00001010110") << "\n";`
	`cout << binTizesbe("00000000100") << "\n";`
		`00011 -- 3`
		`11 -- 3`
		`00001010110 -- 86`
		`00000000100 -- 4`
			`00011 -- 3`
			`11 -- 3`
			`00001010110 -- 86`
			`00000000100 -- 4`
				✔

Passed all tests! ✔

Question author's solution:

```
int binTizesbe(string n) {
  int szam = 0;
  cout << n << " -- ";
  for(unsigned i = 0; i < n.length(); i++) {
    szam = szam * 2 + n[i]-'0';
  }
  return szam;
}
```

Helyes

Leadásra bejelöl: 1/1.

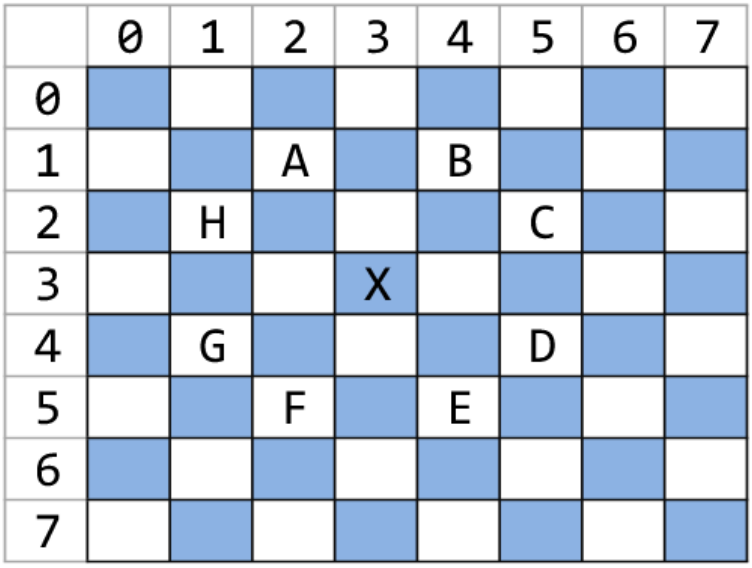
5 kérdés

Helyes

1 közül 1 leosztályozva

Írjon függvényt, ami egy 8x8-as sakktáblán fogadja el a ló lépéseit. A függvény neve legyen `lepesEllenor`. Az argumentunok: `bolS` jelölje a sort ahonnan, a `bolO` pedig az oszlopot ahonnan indul a lépés. A `baS` a sor, a `baO` pedig az oszlop ahova érkezik a ló. Térjen vissza `true`-val, ha szabályos a lépés `false`-al, minden más esetben (szabálytalan lépés, hibás input).

Segítség:



Szintaktikai hiba esetén a fordító által jelzett sor számából vonjon ki 10-et, hogy megkapja a szerkesztőben olvasható sorszámot!

For example:

Test	Result

`cout << (lepesEllenor(3,3,2,1) ? " helyes lepes" : " hibas lepes") << "\n";`
`cout << (lepesEllenor(3,3,1,2) ? " helyes lepes" : " hibas lepes") << "\n";`
`cout << (lepesEllenor(3,3,4,1) ? " helyes lepes" : " hibas lepes") << "\n";`
`cout << (lepesEllenor(3,3,5,2) ? " helyes lepes" : " hibas lepes") << "\n";`
`cout << (lepesEllenor(3,3,3,4) ? " helyes lepes" : " hibas lepes") << "\n";`
`cout << (lepesEllenor(9,9,8,7) ? " helyes lepes" : " hibas lepes") << "\n";`
	`[3,3] -> [2,1] helyes lepes`
	`[3,3] -> [1,2] helyes lepes`
	`[3,3] -> [4,1] helyes lepes`
	`[3,3] -> [5,2] helyes lepes`
	`[3,3] -> [3,4] hibas lepes`
	`[9,9] -> [8,7] helyes lepes`

Answer: (penalty regime: 0 %)

Reset answer

```
1  bool lepesEllenor(unsigned short bolS, unsigned short bol0
2  {
3      cout << "["<< bolS << "," << bol0 << "] -> [" << baS
4      bool szab = false;
5      if ((baS == bolS - 1 || baS == bolS + 1) && (ba0 == b
6          szab = true;
7      if ((ba0 == bol0 - 1 || ba0 == bol0 + 1) && (baS == b
8          szab = true;
9      return szab;
10 }
```

	Test	Expected	Got	

✔	`cout << (lepesEllenor(3,3,2,1) ? " helyes lepes" : " hibas lepes") << "\n";`
	`cout << (lepesEllenor(3,3,1,2) ? " helyes lepes" : " hibas lepes") << "\n";`
	`cout << (lepesEllenor(3,3,4,1) ? " helyes lepes" : " hibas lepes") << "\n";`
	`cout << (lepesEllenor(3,3,5,2) ? " helyes lepes" : " hibas lepes") << "\n";`
	`cout << (lepesEllenor(3,3,1,4) ? " helyes lepes" : " hibas lepes") << "\n";`
	`cout << (lepesEllenor(3,3,4,5) ? " helyes lepes" : " hibas lepes") << "\n";`
	`cout << (lepesEllenor(3,3,5,4) ? " helyes lepes" : " hibas lepes") << "\n";`
		`[3,3] -> [2,1] helyes lepes`
		`[3,3] -> [1,2] helyes lepes`
		`[3,3] -> [4,1] helyes lepes`
		`[3,3] -> [5,2] helyes lepes`
		`[3,3] -> [1,4] helyes lepes`
		`[3,3] -> [4,5] helyes lepes`
		`[3,3] -> [5,4] helyes lepes`
			`[3,3] -> [2,1] helyes lepes`
			`[3,3] -> [1,2] helyes lepes`
			`[3,3] -> [4,1] helyes lepes`
			`[3,3] -> [5,2] helyes lepes`
			`[3,3] -> [1,4] helyes lepes`
			`[3,3] -> [4,5] helyes lepes`
			`[3,3] -> [5,4] helyes lepes`
				✔

✔	`cout << (lepesEllenor(3,3,2,1) ? " helyes`	`[3,3] -> [2,1]`	`[3,3] -> [2,1]`	✔

	lepes" : " hibas lepes") << "\n"; cout << (lepesEllenor(3,3,1,2) ? " helyes lepes" : " hibas lepes") << "\n"; cout << (lepesEllenor(3,3,4,1) ? " helyes lepes" : " hibas lepes") << "\n"; cout << (lepesEllenor(3,3,5,2) ? " helyes lepes" : " hibas lepes") << "\n"; cout << (lepesEllenor(3,3,3,4) ? " helyes lepes" : " hibas lepes") << "\n"; cout << (lepesEllenor(9,9,8,7) ? " helyes lepes" : " hibas lepes") << "\n";	helyes lepes [3,3] -> [1,2] helyes lepes [3,3] -> [4,1] helyes lepes [3,3] -> [5,2] helyes lepes [3,3] -> [3,4] hibas lepes [9,9] -> [8,7] helyes lepes	helyes lepes [3,3] -> [1,2] helyes lepes [3,3] -> [4,1] helyes lepes [3,3] -> [5,2] helyes lepes [3,3] -> [3,4] hibas lepes [9,9] -> [8,7] helyes lepes	
✔	cout << (lepesEllenor(4,3,3,1) ? " helyes lepes" : " hibas lepes") << "\n"; cout << (lepesEllenor(5,5,3,4) ? " helyes lepes" : " hibas lepes") << "\n"; cout << (lepesEllenor(9,9,9,9) ? " helyes lepes" : " hibas lepes") << "\n"; cout << (lepesEllenor(1,7,2,5) ? " helyes lepes" : " hibas lepes") << "\n";	[4,3] -> [3,1] helyes lepes [5,5] -> [3,4] helyes lepes [9,9] -> [9,9] hibas lepes [1,7] -> [2,5] helyes lepes	[4,3] -> [3,1] helyes lepes [5,5] -> [3,4] helyes lepes [9,9] -> [9,9] hibas lepes [1,7] -> [2,5] helyes lepes	✔

Passed all tests! ✔

Question author's solution:

```
bool lepesEllenor(unsigned short bolS, unsigned short bolO, unsigned short baS, unsigned short
baO)
{
    cout << "["<< bolS << "," << bolO << "] -> [" << baS << "," << baO << "]";
    unsigned short sor = abs(bolS - baS);
    unsigned short oszlop = abs(bolO - baO);
    if((sor != 1 or oszlop != 2) and (oszlop != 1 or sor != 2)){
        return false;
    }
    else{
        return true;
    }
}
```

Helyes

Leadásra bejelöl: 1/1.

◀ Mintavizsga Ugrás... Előadás fóliák ▶