

## Câu hỏi 1

Đúng

Đạt điểm 10,00 trên 10,00

## Yêu cầu

Nhập vào số nguyên dương  $n$  và in ra dãy  $1\ 2\ 3\ \dots\ n$  gồm các số dương nhỏ hơn hoặc bằng  $n$ , cách nhau bởi dấu cách.

## Input

- Một dòng văn bản chứa số  $n > 0$

## Output

- Một dòng văn bản chứa  $n$  số nguyên dương  $1\ 2\ 3\ \dots\ n$ , hai số liền nhau cách nhau bởi dấu cách

## Gợi ý

- Sử dụng vòng lặp for, while hoặc do ... while
- Khởi tạo biến đếm bằng 1 (số đầu tiên cần in ra)
- In ra biến đếm và dấu cách rồi tăng biến đếm lên 1

## Lưu ý

- Riêng vòng lặp do ... while sẽ thực hiện khối lệnh trước khi kiểm tra điều kiện, nên khối lệnh sẽ được thực hiện ít nhất một lần
- Đối với bài này, các vòng lặp sau là tương đương

```
for (int i = 1; i <= n; i++) {  
    // lệnh in  
}
```

```
int i = 1;  
while (i <= n) {  
    // lệnh in  
    ++i;  
}
```

```
int i = 1;  
do {  
    // lệnh in  
    ++i;  
} while (i <= n);
```

For example:

Input	Result
3	1 2 3

Answer: (penalty regime: 0 %)

```
1 #include <iostream>  
2 using namespace std;  
3  
4 signed main() {  
5     int n;  
6     cin >> n;  
7     for (int i = 1; i <= n; ++i) {  
8         cout << i << ' ';  
9     }  
10 }
```

	Input	Expected	Got	
✓	3	1 2 3	1 2 3	✓
✓	5	1 2 3 4 5	1 2 3 4 5	✓
✓	10	1 2 3 4 5 6 7 8 9 10	1 2 3 4 5 6 7 8 9 10	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu hỏi 2

Đúng

Đạt điểm 10,00 trên 10,00

### Yêu cầu

Nhập một số nguyên dương  $n$  từ bàn phím.

Kiểm tra và in ra màn hình tất cả các số nguyên dương nhỏ hơn hoặc bằng  $n$  và chia hết cho 7.

### Input

- Một dòng văn bản chứa số nguyên  $n$ , biết  $0 < n < 10000$

### Output

- Một dòng văn bản chứa các số nguyên dương nhỏ hơn hoặc bằng  $n$  và chia hết cho 7 theo thứ tự từ nhỏ đến lớn, hai số cách nhau bởi dấu cách

### Gợi ý

- Có thể dùng cả ba loại vòng lặp `for`, `while` và `do ... while`
- Biến đếm được khởi tạo bằng số đầu tiên chia hết cho 7 (số 7), sau đó mỗi lần lặp tăng biến đếm thêm 7 đơn vị để bảo đảm nó luôn chia hết cho 7.

### Lưu ý

- Nếu tăng biến đếm 1 đơn vị mỗi lần lặp thì cần kiểm tra tính chia hết cho 7 trước khi in
- Cũng có thể tăng biến đếm mỗi lần 1 đơn vị nhưng in ra 7 lần biến đếm (`i*7`) và kiểm tra `i*7 <= n`

For example:

Input	Result
15	7 14

Answer: (penalty regime: 0 %)

```
1 #include <iostream>
2 using namespace std;
3
4 signed main() {
5     int n;
6     cin >> n;
7     for (int i = 7; i <= n; i += 7) {
8         cout << i << ' ';
9     }
10 }
```

	Input	Expected	Got	
✓	15	7 14	7 14	✓
✓	1000	7 14 21 28 35 42 49 56 63 70 77 84 91 98 105 112 119 126 133 140 147 154 161 168 175 182 189 196 203 210 217 224 231 238 245 252 259 266 273 280 287 294 301 308 315 322 329 336 343 350 357 364 371 378 385 392 399 406 413 420 427 434 441 448 455 462 469 476 483 490 497 504 511 518 525 532 539 546 553 560 567 574 581 588 595 602 609 616 623 630 637 644 651 658 665 672 679 686 693 700 707 714 721 728 735 742 749 756 763 770 777 784 791 798 805 812 819 826 833 840 847 854 861 868 875 882 889 896 903 910 917 924 931 938 945 952 959 966 973 980 987 994	7 14 21 28 35 42 49 56 63 70 77 84 91 98 105 112 119 126 133 140 147 154 161 168 175 182 189 196 203 210 217 224 231 238 245 252 259 266 273 280 287 294 301 308 315 322 329 336 343 350 357 364 371 378 385 392 399 406 413 420 427 434 441 448 455 462 469 476 483 490 497 504 511 518 525 532 539 546 553 560 567 574 581 588 595 602 609 616 623 630 637 644 651 658 665 672 679 686 693 700 707 714 721 728 735 742 749 756 763 770 777 784 791 798 805 812 819 826 833 840 847 854 861 868 875 882 889 896 903 910 917 924 931 938 945 952 959 966 973 980 987 994	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

//

### Câu hỏi 3

Đúng

Đạt điểm 10,00 trên 10,00

## Yêu cầu

Viết chương trình liệt kê tất cả các số chính phương nằm trong khoảng từ cận dưới  $lowerBound$  đến cận trên  $upperBound$ .

## Input

- Một dòng chứa hai số nguyên dương  $lowerBound$  và  $upperBound$

## Output

- Một dòng chứa tất cả các số chính phương nằm trong khoảng  $[lowerBound, upperBound]$  theo thứ tự tăng dần, hai số cách nhau bởi một dấu cách

## Gợi ý

- Cách nhanh nhất để giải bài này là dùng một vòng lặp có biến đếm chạy từ  $\sqrt{lowerBound}$  đến  $\sqrt{upperBound}$
- In ra bình phương của số đếm sau khi kiểm tra có nằm trong khoảng  $[lowerBound, upperBound]$

## Lưu ý

- Nếu dùng biến đếm chạy từ  $lowerBound$  đến  $upperBound$  thì phải kiểm tra số đó có phải là số chính phương trước khi in. Cách này chậm vì hầu hết các con số được duyệt không phải là số chính phương. Ví dụ  $lowerBound = 1$  và  $upperBound = 10^9$  thì phải lặp 1 tỷ lần. Trong khi đó với cách ở phần gợi ý, vòng lặp chỉ lặp  $\sqrt{10^9}$  lần.

For example:

Input	Result
4 20	4 9 16

Answer: (penalty regime: 10, 20, ... %)

```
1 #include <iostream>
2 #include <cmath>
3 using namespace std;
4
5 signed main() {
6     int a, b;
7     cin >> a >> b;
8
9     int lo = ceil(sqrt(a));
10    int hi = sqrt(b);
11    for (int i = lo; i <= hi; ++i) {
12        cout << i * i << ' ';
13    }
14 }
```

	Input	Expected	Got	
✓	4 20	4 9 16	4 9 16	✓

	Input	Expected	Got	
✓	100 1000000	100 121 144 169 196 225 256 289 324 361 400 441 484 529 576 625 676 729 784 841 900 961 1024 1089 1156 1225 1296 1369 1444 1521 1600 1681 1764 1849 1936 2025 2116 2209 2304 2401 2500 2601 2704 2809 2916 3025 3136 3249 3364 3481 3600 3721 3844 3969 4096 4225 4356 4489 4624 4761 4900 5041 5184 5329 5476 5625 5776 5929 6084 6241 6400 6561 6724 6889 7056 7225 7396 7569 7744 7921 8100 8281 8464 8649 8836 9025 9216 9409 9604 9801 10000 10201 10404 10609 10816 11025 11236 11449 11664 11881 12100 12321 12544 12769 12996 13225 13456 13689 13924 14161 14400 14641 14884 15129 15376 15625 15876 16129 16384 16641 16900 17161 17424 17689 17956 18225 18496 18769 19044 19321 19600 19881 20164 20449 20736 21025 21316 21609 21904 22201 22500 22801 23104 23409 23716 24025 24336 24649 24964 25281 25600 25921 26244 26569 26896 27225 27556 27889 28224 28561 28900 29241 29584 29929 30276 30625 30976 31329 31684 32041 32400 32761 33124 33489 33856 34225 34596 34969 35344 35721 36100 36481 36864 37249 37636 38025 38416 38809 39204 39601 40000 40401 40804 41209 41616 42025 42436 42849 43264 43681 44100 44521 44944 45369 45796 46225 46656 47089 47524 47961 48400 48841 49284 49729 50176 50625 51076 51529 51984 52441 52900 53361 53824 54289 54756 55225 55696 56169 56644 57121 57600 58081 58564 59049 59536 60025 60516 61009 61504 62001 62500 63001 63504 64009 64516 65025 65536 66049 66564 67081 67600 68121 68644 69169 69696 70225 70756 71289 71824 72361 72900 73441 73984 74529 75076 75625 76176 76729 77284 77841 78400 78961 79524 80089 80656 81225 81796 82369 82944 83521 84100 84681 85264 85849 86436 87025 87616 88209 88804 89401 90000 90601 91204 91809 92416 93025 93636 94249 94864 95481 96100 96721 97344 97969 98596 99225 99856 100489 101124 101761 102400 103041 103684 104329 104976 105625 106276 106929 107584 108241 108900 109561 110224 110889 111556 112225 112896 113569 114244 114921 115600 116281 116964 117649 118336 119025 119716 120409 121104 121801 122500 123201 123904 124609 125316 126025 126736 127449 128164 128881 129600 130321 131044 131769 132496 133225 133956 134689 135424 136161 136900 137641 138384 139129 139876 140625 141376 142129 142884 143641 144400 145161 145924 146689 147456 148225 148996 149769 150544 151321 152100 152881 153664 154449 155236 156025 156816 157609 158404 159201 160000 160801 161604 162409 163216 164025 164836 165649 166464 167281 168100 168921 169744 170569 171396 172225 173056 173889 174724 175561 176400 177241 178084 178929 179776 180625 181476 182329 183184 184041 184900 185761 186624 187489 188356 189225 190096 190969 191844 192721 193600 194481 195364 196249 197136 198025 198916 199809 200704 201601 202500 203401 204304	100 121 144 169 196 225 256 289 324 361 400 441 484 529 576 625 676 729 784 841 900 961 1024 1089 1156 1225 1296 1369 1444 1521 1600 1681 1764 1849 1936 2025 2116 2209 2304 2401 2500 2601 2704 2809 2916 3025 3136 3249 3364 3481 3600 3721 3844 3969 4096 4225 4356 4489 4624 4761 4900 5041 5184 5329 5476 5625 5776 5929 6084 6241 6400 6561 6724 6889 7056 7225 7396 7569 7744 7921 8100 8281 8464 8649 8836 9025 9216 9409 9604 9801 10000 10201 10404 10609 10816 11025 11236 11449 11664 11881 12100 12321 12544 12769 12996 13225 13456 13689 13924 14161 14400 14641 14884 15129 15376 15625 15876 16129 16384 16641 16900 17161 17424 17689 17956 18225 18496 18769 19044 19321 19600 19881 20164 20449 20736 21025 21316 21609 21904 22201 22500 22801 23104 23409 23716 24025 24336 24649 24964 25281 25600 25921 26244 26569 26896 27225 27556 27889 28224 28561 28900 29241 29584 29929 30276 30625 30976 31329 31684 32041 32400 32761 33124 33489 33856 34225 34596 34969 35344 35721 36100 36481 36864 37249 37636 38025 38416 38809 39204 39601 40000 40401 40804 41209 41616 42025 42436 42849 43264 43681 44100 44521 44944 45369 45796 46225 46656 47089 47524 47961 48400 48841 49284 49729 50176 50625 51076 51529 51984 52441 52900 53361 53824 54289 54756 55225 55696 56169 56644 57121 57600 58081 58564 59049 59536 60025 60516 61009 61504 62001 62500 63001 63504 64009 64516 65025 65536 66049 66564 67081 67600 68121 68644 69169 69696 70225 70756 71289 71824 72361 72900 73441 73984 74529 75076 75625 76176 76729 77284 77841 78400 78961 79524 80089 80656 81225 81796 82369 82944 83521 84100 84681 85264 85849 86436 87025 87616 88209 88804 89401 90000 90601 91204 91809 92416 93025 93636 94249 94864 95481 96100 96721 97344 97969 98596 99225 99856 100489 101124 101761 102400 103041 103684 104329 104976 105625 106276 106929 107584 108241 108900 109561 110224 110889 111556 112225 112896 113569 114244 114921 115600 116281 116964 117649 118336 119025 119716 120409 121104 121801 122500 123201 123904 124609 125316 126025 126736 127449 128164 128881 129600 130321 131044 131769 132496 133225 133956 134689 135424 136161 136900 137641 138384 139129 139876 140625 141376 142129 142884 143641 144400 145161 145924 146689 147456 148225 148996 149769 150544 151321 152100 152881 153664 154449 155236 156025 156816 157609 158404 159201 160000 160801 161604 162409 163216 164025 164836 165649 166464 167281 168100 168921 169744 170569 171396 172225 173056 173889 174724 175561 176400 177241 178084 178929 179776 180625 181476 182329 183184 184041 184900 185761 186624 187489 188356 189225 190096 190969 191844 192721 193600 194481 195364 196249 197136 198025 198916 199809 200704 201601 202500 203401	✓

Input	Expected	Got
	205209 206116 207025 207936 208849 209764 210681 211600 212521 213444 214369 215296 216225 217156 218089 219024 219961 220900 221841 222784 223729 224676 225625 226576 227529 228484 229441 230400 231361 232324 233289 234256 235225 236196 237169 238144 239121 240100 241081 242064 243049 244036 245025 246016 247009 248004 249001 250000 251001 252004 253009 254016 255025 256036 257049 258064 259081 260100 261121 262144 263169 264196 265225 266256 267289 268324 269361 270400 271441 272484 273529 274576 275625 276676 277729 278784 279841 280900 281961 283024 284089 285156 286225 287296 288369 289444 290521 291600 292681 293764 294849 295936 297025 298116 299209 300304 301401 302500 303601 304704 305809 306916 308025 309136 310249 311364 312481 313600 314721 315844 316969 318096 319225 320356 321489 322624 323761 324900 326041 327184 328329 329476 330625 331776 332929 334084 335241 336400 337561 338724 339889 341056 342225 343396 344569 345744 346921 348100 349281 350464 351649 352836 354025 355216 356409 357604 358801 360000 361201 362404 363609 364816 366025 367236 368449 369664 370881 372100 373321 374544 375769 376996 378225 379456 380689 381924 383161 384400 385641 386884 388129 389376 390625 391876 393129 394384 395641 396900 398161 399424 400689 401956 403225 404496 405769 407044 408321 409600 410881 412164 413449 414736 416025 417316 418609 419904 421201 422500 423801 425104 426409 427716 429025 430336 431649 432964 434281 435600 436921 438244 439569 440896 442225 443556 444889 446224 447561 448900 450241 451584 452929 454276 455625 456976 458329 459684 461041 462400 463761 465124 466489 467856 469225 470596 471969 473344 474721 476100 477481 478864 480249 481636 483025 484416 485809 487204 488601 490000 491401 492804 494209 495616 497025 498436 499849 501264 502681 504100 505521 506944 508369 509796 511225 512656 514089 515524 516961 518400 519841 521284 522729 524176 525625 527076 528529 529984 531441 532900 534361 535824 537289 538756 540225 541696 543169 544644 546121 547600 549081 550564 552049 553536 555025 556516 558009 559504 561001 562500 564001 565504 567009 568516 570025 571536 573049 574564 576081 577600 579121 580644 582169 583696 585225 586756 588289 589824 591361 592900 594441 595984 597529 599076 600625 602176 603729 605284 606841 608400 609961 611524 613089 614656 616225 617796 619369 620944 622521 624100 625681 627264 628849 630436 632025 633616 635209 636804 638401 640000 641601 643204 644809 646416 648025 649636 651249 652864 654481 656100 657721 659344 660969 662596 664225 665856 667489 669124 670761 672400 674041 675684 677329 678976 680625 682276 683929 685584 687241 688900 690561 692224 693889 695556 697225 698896 700569	204304 205209 206116 207025 207936 208849 209764 210681 211600 212521 213444 214369 215296 216225 217156 218089 219024 219961 220900 221841 222784 223729 224676 225625 226576 227529 228484 229441 230400 231361 232324 233289 234256 235225 236196 237169 238144 239121 240100 241081 242064 243049 244036 245025 246016 247009 248004 249001 250000 251001 252004 253009 254016 255025 256036 257049 258064 259081 260100 261121 262144 263169 264196 265225 266256 267289 268324 269361 270400 271441 272484 273529 274576 275625 276676 277729 278784 279841 280900 281961 283024 284089 285156 286225 287296 288369 289444 290521 291600 292681 293764 294849 295936 297025 298116 299209 300304 301401 302500 303601 304704 305809 306916 308025 309136 310249 311364 312481 313600 314721 315844 316969 318096 319225 320356 321489 322624 323761 324900 326041 327184 328329 329476 330625 331776 332929 334084 335241 336400 337561 338724 339889 341056 342225 343396 344569 345744 346921 348100 349281 350464 351649 352836 354025 355216 356409 357604 358801 360000 361201 362404 363609 364816 366025 367236 368449 369664 370881 372100 373321 374544 375769 376996 378225 379456 380689 381924 383161 384400 385641 386884 388129 389376 390625 391876 393129 394384 395641 396900 398161 399424 400689 401956 403225 404496 405769 407044 408321 409600 410881 412164 413449 414736 416025 417316 418609 419904 421201 422500 423801 425104 426409 427716 429025 430336 431649 432964 434281 435600 436921 438244 439569 440896 442225 443556 444889 446224 447561 448900 450241 451584 452929 454276 455625 456976 458329 459684 461041 462400 463761 465124 466489 467856 469225 470596 471969 473344 474721 476100 477481 478864 480249 481636 483025 484416 485809 487204 488601 490000 491401 492804 494209 495616 497025 498436 499849 501264 502681 504100 505521 506944 508369 509796 511225 512656 514089 515524 516961 518400 519841 521284 522729 524176 525625 527076 528529 529984 531441 532900 534361 535824 537289 538756 540225 541696 543169 544644 546121 547600 549081 550564 552049 553536 555025 556516 558009 559504 561001 562500 564001 565504 567009 568516 570025 571536 573049 574564 576081 577600 579121 580644 582169 583696 585225 586756 588289 589824 591361 592900 594441 595984 597529 599076 600625 602176 603729 605284 606841 608400 609961 611524 613089 614656 616225 617796 619369 620944 622521 624100 625681 627264 628849 630436 632025 633616 635209 636804 638401 640000 641601 643204 644809 646416 648025 649636 651249 652864 654481 656100 657721 659344 660969 662596 664225 665856 667489 669124 670761 672400 674041 675684 677329 678976 680625 682276 683929 685584 687241 688900 690561 692224 693889 695556 697225 698896

	Input	Expected	Got	
		702244 703921 705600 707281 708964 710649 712336 714025 715716 717409 719104 720801 722500 724201 725904 727609 729316 731025 732736 734449 736164 737881 739600 741321 743044 744769 746496 748225 749956 751689 753424 755161 756900 758641 760384 762129 763876 765625 767376 769129 770884 772641 774400 776161 777924 779689 781456 783225 784996 786769 788544 790321 792100 793881 795664 797449 799236 801025 802816 804609 806404 808201 810000 811801 813604 815409 817216 819025 820836 822649 824464 826281 828100 829921 831744 833569 835396 837225 839056 840889 842724 844561 846400 848241 850084 851929 853776 855625 857476 859329 861184 863041 864900 866761 868624 870489 872356 874225 876096 877969 879844 881721 883600 885481 887364 889249 891136 893025 894916 896809 898704 900601 902500 904401 906304 908209 910116 912025 913936 915849 917764 919681 921600 923521 925444 927369 929296 931225 933156 935089 937024 938961 940900 942841 944784 946729 948676 950625 952576 954529 956484 958441 960400 962361 964324 966289 968256 970225 972196 974169 976144 978121 980100 982081 984064 986049 988036 990025 992016 994009 996004 998001 1000000	700569 702244 703921 705600 707281 708964 710649 712336 714025 715716 717409 719104 720801 722500 724201 725904 727609 729316 731025 732736 734449 736164 737881 739600 741321 743044 744769 746496 748225 749956 751689 753424 755161 756900 758641 760384 762129 763876 765625 767376 769129 770884 772641 774400 776161 777924 779689 781456 783225 784996 786769 788544 790321 792100 793881 795664 797449 799236 801025 802816 804609 806404 808201 810000 811801 813604 815409 817216 819025 820836 822649 824464 826281 828100 829921 831744 833569 835396 837225 839056 840889 842724 844561 846400 848241 850084 851929 853776 855625 857476 859329 861184 863041 864900 866761 868624 870489 872356 874225 876096 877969 879844 881721 883600 885481 887364 889249 891136 893025 894916 896809 898704 900601 902500 904401 906304 908209 910116 912025 913936 915849 917764 919681 921600 923521 925444 927369 929296 931225 933156 935089 937024 938961 940900 942841 944784 946729 948676 950625 952576 954529 956484 958441 960400 962361 964324 966289 968256 970225 972196 974169 976144 978121 980100 982081 984064 986049 988036 990025 992016 994009 996004 998001 1000000	
✓	100 1000	100 121 144 169 196 225 256 289 324 361 400 441 484 529 576 625 676 729 784 841 900 961	100 121 144 169 196 225 256 289 324 361 400 441 484 529 576 625 676 729 784 841 900 961	✓

Passed all tests! ✓

Dúng

Marks for this submission: 10,00/10,00.



#### Câu hỏi 4

Đúng

Đạt điểm 10,00 trên 10,00

### Yêu cầu:

"Vừa gà vừa chó,

Bó lại cho tròn,

Ba mươi sáu con,

Một trăm chân chẵn"

Từ bài toán dân gian trên, mở rộng thành chương trình nhận đầu vào là tổng số con và tổng số chân của gà và chó. Nếu tìm được số phù hợp, in ra số lượng gà và số lượng chó. Ngược lại, in ra "invalid"

### Input:

- Gồm 2 số nguyên dương: tổng số con và tổng số chân

### Output:

- Nếu tìm ra được phương án phù hợp, in ra số lượng gà và số lượng chó theo mẫu sau: "chicken = <số gà>, dog = <số chó>".
- Nếu không tìm được phương án phù hợp, in ra "invalid"

### Gợi ý:

- Khai báo biến `int total, totalLegs;` và nhập dữ liệu `cin >> total >> totalLegs;`
- Khai báo biến `bool flag = false` để xác định bài toán có lời giải hay không, có giá trị ban đầu là false có nghĩa là chưa tìm ra được lời giải.
- Sử dụng vòng lặp for kiểm tra hết các trường hợp. Khai báo biến chạy `int numChicken` là biến chạy của vòng `for` (bạn cũng có thể để biến chạy là `numDog`, cần chú ý điều kiện)
- Trong vòng for dùng câu lệnh rẽ nhánh `if` để kiểm tra điều kiện `numChicken*2+(total-numChicken)*4 == totalLegs` . Nếu đúng in ra như mẫu output, chuyển biến `flag = true` do chúng ta đã tìm được phương án thích hợp.
- Nếu sau khi vòng for kết thúc, kiểm tra giá trị `flag`, nếu `flag == false` có nghĩa là không tìm ra phương án phù hợp, in ra "invalid".

### Lưu ý:

Trong vòng for khi tìm được phương án phù hợp, in ra và đổi giá trị `flag`; chúng ta có thể dùng câu lệnh `break;` để kết thúc vòng for sớm

#### For example:

Input	Result
36 100	chicken = 22, dog = 14

Answer: (penalty regime: 0 %)

```
1 #include <iostream>
2 using namespace std;
3
4 signed main() {
5     // ga + cho = x
6     // 2ga + 4cho = y
7     // ga = (4x - y) / 2
8     // cho = (2x - y) / -2
9
10    int x, y;
11    cin >> x >> y;
12    if ((4*x - y) % 2 != 0 || (2*x - y) % 2 != 0) {
13        cout << "invalid" << endl;
14        return 0;
15    }
16    int ga = (4*x - y) / 2;
17    int cho = (2*x - y) / -2;
18    if (ga < 0 || cho < 0) {
19        cout << "invalid" << endl;
20        return 0;
21    }
22    cout << "chicken = " << ga << ", dog = " << cho << endl;
23    return 0;
24 }
```

```

13     cout << invalid ;
14 } else {
15     cout << "chicken = " << (4*x - y) / 2 << ", dog = " << (y - 2*x) / 2;
16 }
17 }

```

	Input	Expected	Got	
✓	36 100	chicken = 22, dog = 14	chicken = 22, dog = 14	✓
✓	37 100	chicken = 24, dog = 13	chicken = 24, dog = 13	✓
✓	37 101	invalid	invalid	✓

Passed all tests! ✓

**Đúng**

Marks for this submission: 10,00/10,00.

Câu hỏi 5

Đúng

Đạt điểm 10,00 trên 10,00

Yêu cầu

Hãy viết chương trình nhận vào từ bàn phím một số nguyên dương  $n$  và in ra màn hình một chữ  $X$  có kích thước  $(2n+1) \times (2n+1)$ .

Input

- Một dòng văn bản chứa số  $n > 0$

Output

- Hình một chữ  $X$  có kích thước  $(2n+1) \times (2n+1)$ .

Gợi ý

- sử dụng 2 vòng lặp for lồng nhau và một câu lệnh rẽ nhánh if-else.
- In hình chữ X chính là in ra 2 đường chéo của hình vuông.

Lưu ý

- Các điểm nằm trên đường chéo chính (từ trái qua phải) có chỉ số hoành độ bằng tung độ ( $i = j$ ).
- Các điểm nằm trên đường chéo phụ (từ phải qua trái) có tổng chỉ số hoành độ và tung độ bằng kích thước hình vuông + 1. ( $i + j = (2*n+1)+1$ )
- Tham khảo cách sử dụng 2 vòng lặp lồng nhau kết hợp vs câu lệnh rẽ nhánh như sau:

```
for (int i = 1; i <= 2*n + 1; i++) {
    for (int j = 1; j <= 2*n + 1; j++){
        if ( ... ) { // Kiểm tra điều kiện
            cout<<"* ";
        }else{
            cout<<". ";
        }
    }
    cout<<endl;
}
```

For example:

Input	Result
3	* . . . . * . * . . . * . . . * . * . . . . . * . . . . . * . * . . . * . . . * . * . . . . *

Answer: (penalty regime: 0 %)

```
1 #include <iostream>
2 using namespace std;
3
4 void drawLine(int i, int n) {
5     int len = 4*n + 1;
6     int pos1 = 2*i;
7     int pos2 = len - pos1 - 1;
8     for (int j = 0; j < len; ++j) {
9         if (j == pos1 || j == pos2) {
```

```

10      cout << " ";
11  } else if (j % 2 == 1) {
12      cout << " ";
13  } else {
14      cout << ".";
15  }
16  }
17  cout << "\n";
18  }
19
20  signed main() {
21      int n;
22      cin >> n;
23
24      for (int i = 0; i < n; ++i)
25          drawLine(i, n);
26  }
27  for (int i = n; i >= 0; --i)
28      drawLine(i, n);
29  }
30  }

```

[illegible]

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu hỏi 6

Đúng

Đạt điểm 10,00 trên 10,00

## Yêu cầu

Hãy viết chương trình nhận vào từ bàn phím một số nguyên  $n$  và in ra một viên kim cương có kích thước  $(2n+1) \times (2n+1)$ .

## Input

- Một dòng văn bản chứa số  $n > 0$

## Output

- In ra một viên kim cương có kích thước  $(2n+1) \times (2n+1)$ .

## Gợi ý

- sử dụng 2 vòng lặp for lồng nhau và câu lệnh rẽ nhánh if-else.
- Cách 1 for 0 đến  $2n-1$  bình thường => khó hiểu
- Cách 2 dùng for x, y chạy từ -n đến n kết hợp biểu thức phương trình hình thoi trong hệ trục tọa độ.

## Lưu ý

- Tham khảo cách cách in ra nửa trên hình thoi như sau:

```
int temp;
for(int x = -n; x <= n; x++){
    for(int y = -n; y <= n; y++){
        temp = x;
        // Nếu muốn in ra nửa dưới hình thoi chỉ cần xét điều kiện của x
        // và cập nhật biến temp ở đây.
        if (y >= -(n + temp) && y <= n + temp){
            cout<<"* ";
        }else{
            cout<<". ";
        }
    }
    cout<<endl;
}
```

For example:

Input	Result
4	. . . . * . . . . . . . * * * . . . . . * * * * . . . * * * * * . * * * * * * . * * * * * . . . * * * * . . . . . * * * . . . . . . * . . . .

Answer: (penalty regime: 0 %)

1	#include <iostream>	
2	using namespace std;	
3		

```

3
4 void drawLine(int i, int n) {
5     int len = 4*n + 1;
6     int pos1 = 2 * (n-i);
7     int pos2 = len - pos1 - 1;
8     for (int j = 0; j < len; ++j) {
9         if (j % 2) {
10             cout << ' ';
11         } else if (pos1 <= j && j <= pos2) {
12             cout << '*';
13         } else {
14             cout << '.';
15         }
16     }
17     cout << '\n';
18 }
19
20 signed main() {
21     int n;
22     cin >> n;
23
24     for (int i = 0; i < n; ++i) {
25         drawLine(i, n);
26     }
27     for (int i = n; i >= 0; --i) {
28         drawLine(i, n);
29     }
30 }

```

	Input	Expected	Got	
✓	4	<pre> . . . . * . . . . . . . * * * . . . . . * * * * * . . . * * * * * * . * * * * * * * . * * * * * * . . . * * * * * . . . . * * * . . . . . . . * . . . . </pre>	<pre> . . . . * . . . . . . . * * * . . . . . * * * * * . . . * * * * * * . * * * * * * * . * * * * * * . . . * * * * * . . . . * * * . . . . . . . * . . . . </pre>	✓
✓	2	<pre> . . * . . . * * . * * * * . * * . . . * . . </pre>	<pre> . . * . . . * * . * * * * . * * . . . * . . </pre>	✓
✓	3	<pre> . . . * . . . . . * * * . . . * * * * . * * * * * . * * * * . . . * * * . . . . . * . . . </pre>	<pre> . . . * . . . . . * * * . . . * * * * . * * * * * . * * * * . . . * * * . . . . . * . . . </pre>	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu hỏi 7

Đúng

Đạt điểm 10,00 trên 10,00

### Yêu cầu:

Viết chương trình sử dụng khai triển Taylor để tính giá trị hàm mũ theo công thức sau:

$$e^x = 1 + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \dots$$

### Input:

- Một số nguyên không âm  $x$  duy nhất ( $0 \leq x \leq 50$ ).

### Output:

- Kết quả  $e^x$  được tính theo công thức với sai số tới 0.001. Kết quả làm tròn đến số thập phân thứ tư.

### Gợi ý:

- Sai số tới 0.001 có nghĩa là tổng trước và sau khi cộng thêm một lượng  $\frac{x^i}{i!}$  không vượt quá 0.001.
- Bắt đầu với biến đếm  $i = 1$ .
- Sử dụng vòng lặp **while** lặp lại quá trình sau với điều kiện thực hiện là **sum - oldSum > 0.001**:
  - Lưu lại giá trị của **sum** vào **oldSum**.
  - (\*) Tính giá trị  $\frac{x^i}{i!}$ .
  - Cộng thêm giá trị vừa tính được vào **sum**.
  - Tăng giá trị biến đếm lên 1.

### Lưu ý:

- Cần lưu ý đến giá trị khởi tạo của **sum** và **oldSum** tương ứng là 1 và 0.
- Liệu có cách làm nào để bước (\*) không cần tính lại giá trị mũ và giai thừa từ đầu không?
- Ở bước i, ta đã tính được  $x^i$  và  $i!$  rồi. Để tận dụng được giá trị này, ta khởi tạo chúng từ trước vòng lặp.
- Tới bước i+1, các giá trị này vẫn giữ nguyên từ bước i, chỉ cần nhân thêm  $x$  vào  $x^i$  là có kết quả  $x^{i+1}$ , tương tự với  $i!$
- Đọc thêm: Nếu cần tính  $e^x$  với  $x$  là một số thực dương bất kỳ, ta có thể làm theo các bước sau

- Tính  $e^f$  với  $f = \lfloor x \rfloor$  là phần nguyên của  $x$

$$e^f = \begin{cases} (e^{f/2})^2 & \text{nếu } f \text{ chẵn} \\ e \times (e^{(f-1)/2})^2 & \text{nếu } f \text{ lẻ} \end{cases}$$

```
double exp_int(int f) {
    if (f == 0)
        return 1.0;

    const double e = 2.718281828459045235360287471352;

    double e2 = exp_int(f/2);

    if (f % 2 == 0)
        return e2*e2;

    else
        return e2*e2*e;
}
```

- Tính  $e^r$  với  $r = x - \lfloor x \rfloor$  là phần lẻ của  $x$  sử dụng khai triển Taylor. Do  $r < 1$  nên sai số của khai triển Taylor đến số hạng  $\frac{x^n}{n!}$  bé hơn  $\frac{3}{(n+1)!}$ . Vòng lặp có thể dừng khi đại lượng này bé hơn 0,0001

For example:

Input	Result
1	2.7183
25	72004899337.3855

Answer: (penalty regime: 0 %)

```
1 #include <iostream>
```



```

1 #include <iostream>
2 #include <iomanip>
3 using namespace std;
4
5 signed main() {
6     int x;
7     cin >> x;
8
9     double val = 1;
10    double sum = 1;
11    for (int i = 1; ; ++i) {
12        val *= 1.0 * x / i;
13        sum += val;
14        if (val <= 0.001) {
15            break;
16        }
17    }
18    cout << fixed << setprecision(4);
19    cout << sum;
20 }

```

	Input	Expected	Got	
✓	1	2.7183	2.7183	✓
✓	25	72004899337.3855	72004899337.3855	✓
✓	10	22026.4656	22026.4656	✓
✓	6	403.4286	403.4286	✓
✓	20	485165195.4096	485165195.4096	✓
✓	8	2980.9577	2980.9577	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

### Câu hỏi 8

Đúng

Đạt điểm 10,00 trên 10,00

## Yêu cầu:

Viết chương trình nhập vào một số nguyên  $n$  và in ra màn hình số chữ số của nó.

## Input:

- Một số nguyên  $n$  duy nhất nhập vào từ bàn phím ( $-10^{18} \leq n \leq 10^{18}$ ).

## Output:

- Một số nguyên duy nhất là số chữ số của  $n$ .

## Gợi ý:

- Dùng vòng lặp `do {...} while (...)`; để giải quyết bài tập này.
- Khởi tạo một biến đếm `count`.
- Xoá bớt một chữ số trong  $n$  bằng cách chia lấy phần nguyên với 10.
- Cộng thêm `count` 1 đơn vị.
- Nếu  $n$  **chưa bằng** 0, quay lại bước 3.

## Lưu ý:

- Sinh viên không dùng dấu ký tự để làm bài tập này.
- Kiểu dữ liệu `long long` (tương đương `long long int`) sẽ lưu được giá trị nguyên từ  $\approx -10^{19}$  đến  $\approx 10^{19}$ .
- Tại bước 5, cần lưu ý điều kiện là  $n$  **chưa bằng** 0 ( $n \neq 0$ ). Rất nhiều bạn sẽ sử dụng điều kiện  $n > 0$ , như vậy sẽ sai với các số âm.
- Sinh viên xem lại bài đã làm và trả lời câu hỏi: Tại sao dùng vòng lặp `do {...} while (...)`; mà không dùng `while (...)` `{...}`?

### For example:

Input	Result
-1593	4

Answer: (penalty regime: 0 %)

```
1 #include <iostream>
2 using namespace std;
3
4 signed main() {
5     long long n;
6     cin >> n;
7
8     // dùng do-while vì nếu dùng while thì
9     // count=0 khi n=0 -> sai
10    int count = 0;
11    do {
12        ++count;
13        n /= 10;
14    } while (n);
15    cout << count;
16 }
```

	Input	Expected	Got	
✓	12	2	2	✓
✓	-1593	4	4	✓
✓	74359	5	5	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

### Câu hỏi 9

Đúng

Đạt điểm 10,00 trên 10,00

## Yêu cầu:

Viết chương trình nhập vào một mảng số thực có độ dài  $n$  và in ra màn hình dãy đó theo thứ tự đảo ngược.  $n$  tối đa 1000

Kết quả làm tròn đến chữ số thập phân thứ hai.

## Input:

- Dòng đầu tiên là độ dài  $n$ ,  $0 < n \leq 1000$ .
- Dòng thứ 2 chứa  $n$  số thực cách nhau bởi dấu cách

## Output:

- In ra dãy đảo ngược đã được làm tròn, cách nhau bởi dấu cách

## Gợi ý:

- Khai báo biến `const int MAX_N = 1000;`
- Khai báo biến `int n;` và nhập dữ liệu `cin >> n;`
- Khởi tạo mảng `double arrayX[MAX_N];`
- Tạo vòng lặp để nhập vào từng phần tử của mảng.
- Tạo vòng lặp với biến chạy từ  $n$  giảm dần để in ra dãy đảo ngược với giá trị đã làm tròn.

## Lưu ý:

- Các hàm làm tròn là `fixed` và `setprecision` nằm trong thư viện `<iomanip>`
- SINH VIÊN KHÔNG ĐƯỢC PHÉP TẠO MẢNG CÓ KÍCH THƯỚC LÀ BIẾN:

```
int n;  
cin >> n;  
int arr[n];
```

## For example:

Input	Result
4 1 2 3 4	4.00 3.00 2.00 1.00

Answer: (penalty regime: 0 %)

```
1 #include <iostream>  
2 #include <iomanip>  
3 using namespace std;  
4  
5 signed main() {  
6     const int MAXN = 1000;  
7     double a[MAXN];  
8  
9     int n;  
10    cin >> n;  
11  
12    for (int i = 0; i < n; ++i) {  
13        cin >> a[i];  
14    }  
15    cout << fixed << setprecision(2);  
16    for (int i = n-1; i >=0; --i) {  
17        cout << a[i] << ' ';  
18    }  
19 }
```

	Input	Expected	Got	
✓	4 1 2 3 4	4.00 3.00 2.00 1.00	4.00 3.00 2.00 1.00	✓
✓	5 91 23 -12.324 21.12323 21	21.00 21.12 -12.32 23.00 91.00	21.00 21.12 -12.32 23.00 91.00	✓

Passed all tests! ✓

**Đúng**

Marks for this submission: 10,00/10,00.

//

## Câu hỏi 10

Đúng

Đạt điểm 10,00 trên 10,00

### Yêu cầu:

Giả sử  $A$  và  $B$  là 2 vectơ  $n$  chiều.

Hãy tính tích vô hướng của 2 vectơ trên bằng công thức:

$$A(x_1, x_2, \dots, x_n) * B(y_1, y_2, \dots, y_n) = x_1 * y_1 + x_2 * y_2 + \dots + x_n * y_n$$

### Input:

- Dòng đầu tiên là độ dài  $n$ ,  $0 < n \leq 1000$
- Dòng thứ 2 chứa tọa độ của vectơ  $A$  là  $n$  số thực cách nhau bởi dấu cách
- Dòng thứ 3 chứa tọa độ của vectơ  $B$  là  $n$  số thực cách nhau bởi dấu cách

### Output:

- Tích vô hướng của vectơ  $A$  và  $B$  với kết quả làm tròn đến chữ số thập phân thứ hai.

### Gợi ý:

1. Khai báo biến `const int MAX_N = 1000;`
2. Khai báo biến `int n;` và nhập dữ liệu `cin >> n;`
3. Khởi tạo 2 mảng `double vectorA[MAX_N]` và `double vectorB[MAX_N]`
4. Tạo vòng lặp để nhập vào từng tọa độ cho 2 mảng
5. Khai báo biến kết quả `double product = 0;`
6. Tạo vòng lặp chạy từ  $i = 0$  đến  $n - 1$ , tại mỗi lần lặp, tính `product += vectorA[i] * vectorB[i];`

### Lưu ý:

1. Khi khai báo biến với từ khóa `const`, biến đó được định nghĩa là hằng số và không thể thay đổi giá trị sau khi khởi tạo.
2. Có thể thay khai báo hai vector trên bằng lệnh  
`vector<double> vectorA, vectorB;`  
để sử dụng kiểu vector của thư viện STL được cung cấp sẵn. Nhớ khai báo thư viện bằng lệnh  
`#include <vector>`  
ở đầu chương trình.
3. Đọc thêm (khó): CPU của máy tính có thể có nhiều lõi (core) dùng để chạy song song các đoạn mã lệnh, tăng tốc độ chạy của chương trình. Tích vô hướng có thể thực hiện tính toán song song bằng các thư viện `<thread>`, `<atomic>`, `<vector>`

### For example:

Input	Result
2	11.00
1 2	
3 4	

Answer: (penalty regime: 0 %)

1	<code>#include &lt;iostream&gt;</code>	
2	<code>#include &lt;iomanip&gt;</code>	
3	<code>using namespace std;</code>	
4		
5	<code>signed main() {</code>	
6	<code>const int MAXN = 1000;</code>	
7	<code>double a[MAXN], b[MAXN];</code>	
8		
9	<code>int n;</code>	
10	<code>cin &gt;&gt; n;</code>	
11		
12	<code>for (int i = 0; i &lt; n; ++i) {</code>	
13	<code>cin &gt;&gt; a[i];</code>	
14	<code>}</code>	

```

15  for (int i = 0; i < n; ++i) {
16      cin >> b[i];
17  }
18  double dotProduct = 0;
19  for (int i = 0; i < n; ++i) {
20      dotProduct += a[i] * b[i];
21  }
22  cout << fixed << setprecision(2);
23  cout << dotProduct;
24  }

```

	Input	Expected	Got	
✓	2 1 2 3 4	11.00	11.00	✓
✓	3 1 1.2 3.5 -2 10 -4	-4.00	-4.00	✓
✓	4 1.23 24.1 230.9 -12.783 213.12 23.1 -1.2334 0.23234	531.09	531.09	✓
✓	5 12 3.4 0.21321 2.23 1.723 3.23 -13 8.32 12.89 2.1	28.70	28.70	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu hỏi 11

Đúng

Đạt điểm 10,00 trên 10,00

### Yêu cầu:

Phương sai của dãy  $X$  gồm  $n$  phần tử được tính theo công thức:

$$\text{var}(X) = \frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2$$

với  $\mu$  là giá trị trung bình (mean) của  $X$

Với công thức trên, hãy viết chương trình tính phương sai của một dãy số thực gồm  $n$  số.

### Input:

- Dòng đầu tiên là độ dài  $n$ ,  $0 < n \leq 1000$ .
- Dòng thứ 2 chứa  $n$  số thực cách nhau bởi dấu cách

### Output:

- Phương sai của dãy  $X$
- Kết quả làm tròn đến chữ số thập phân thứ hai.

### Gợi ý:

- Khai báo biến `const int MAX_N = 1000;`
- Khai báo biến `int n;` và nhập dữ liệu `cin >> n;`
- Khởi tạo mảng `double arrayX[MAX_N];`
- Khai báo biến tổng `double sum = 0;`
- Tạo vòng lặp để nhập vào từng phần tử của mảng, đồng thời cập nhật tổng tại mỗi lần lặp `sum += arrayX[i];`
- Tính giá trị trung bình `double mean = sum / n;`
- Tạo vòng lặp để tính phương sai theo công thức đã cho.

For example:

Input	Result
8 5 10 15 2 4 6 8 10	15.00

Answer: (penalty regime: 0 %)

```
1 #include <iostream>
2 #include <iomanip>
3 using namespace std;
4
5 signed main() {
6     const int MAXN = 1000;
7     double a[MAXN];
8
9     int n;
10    cin >> n;
11
12    double sum = 0;
13    for (int i = 0; i < n; ++i) {
14        cin >> a[i];
15        sum += a[i];
16    }
17    double mean = sum / n;
18    sum = 0;
19    for (int i = 0; i < n; ++i) {
20        sum += (a[i] - mean) * (a[i] - mean);
21    }
22    double var = sum / n;
23    cout << fixed << setprecision(2);
24    cout << var;
25 }
```



	Input	Expected	Got	
✓	8 5 10 15 2 4 6 8 10	15.00	15.00	✓
✓	10 -1 3 8374 23 -1392 34 234 -83 -263 10	6737917.69	6737917.69	✓
✓	3 2 4 5	1.56	1.56	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

Trở lại Khoá học

//