Iniciado em	domingo, 15 ago 2021, 12:44
Estado	Finalizada
Concluída em	domingo, 15 ago 2021, 13:56
Tempo empregado	1 hora 11 minutos
Avaliar	**9,86** de um máximo de 10,00(**99**%)

Aviso!

?

Questão 1

Correto

Atingiu 1,00 de 1,00

O problema do Brines

Paulo é um aluno de Ciência da Computação de dia, mas de noite ele é mais conhecido como DJ Brines, isso mesmo ! O famoso DJ Brines! Mas infelizmente Brines está muito atarefado na matéria de Estrutura de Dados e não está tendo tempo para compor suas obras de arte. Brines teve então a brilhante ideia de contratar um produtor musical, que avisou Brines que a duração de sua música deveria ter no mínimo a minutos e no máximo b minutos para ser um sucesso nacional. Mais formalmente, para a música fazer sucesso, devemos ter $(0 < a <= x <= b < 10000)$. Como Brines está sem tempo, escreva um programa que ajude-o a identificar se essa música será um sucesso ou se é melhor ele começar a pensar na próxima música.

Entrada

A entrada consiste em três linhas. Cada linha possui um inteiro positivo a, b $(0 < a < b < 10000)$ e x respectivamente, informados pelo usuário, como descritos no enunciado. Não é necessário validar se os números estão dentro do intervalo definido.

Saída

Imprima na tela "É O BRINES, NÃO ADIANTA!", caso a música esteja dentro do tempo, ou "não foi dessa vez Brines :(" caso contrário.

Observações

- No primeiro caso de teste, o resultado é "É O BRINES, NÃO ADIANTA!", pois o x está no intervalo.
- No segundo caso de teste, o resultado é "não foi dessa vez Brines :(", pois o x não está no intervalo.
- No terceiro caso de teste, o resultado é "É O BRINES, NÃO ADIANTA", pois o x está no intervalo.

For example:

Input	Result
10 30 15	É O BRINES, NÃO ADIANTA!

Aviso!

Input	Result
34 35 1	não foi dessa vez Brines :(
10 50 39	É O BRINES, NÃO ADIANTA!

Answer: (penalty regime: 0, 0, 10, 20, ... %)

```
MIN = int(input())
MAX = int(input())
tam = int(input())

if tam >= MIN and tam <= MAX:
    print('É O BRINES, NÃO ADIANTA!')
else:
    print('não foi dessa vez Brines :(')
```

	Input	Expected	Got	
✔	10 30 15	É O BRINES, NÃO ADIANTA!	É O BRINES, NÃO ADIANTA!	✔

Aviso!

	Input	Expected	Got	
✔	34 35 1	não foi dessa vez Brines :(	não foi dessa vez Brines :(	✔
✔	10 50 39	É O BRINES, NÃO ADIANTA!	É O BRINES, NÃO ADIANTA!	✔
✔	37 56 56	É O BRINES, NÃO ADIANTA!	É O BRINES, NÃO ADIANTA!	✔
✔	21 43 20	não foi dessa vez Brines :(	não foi dessa vez Brines :(	✔
✔	3 9000 36	É O BRINES, NÃO ADIANTA!	É O BRINES, NÃO ADIANTA!	✔

Passou em todos os teste! ✔

Para resolver o problema, deve-se implementar um if verificando se o x dado está no intervalo $[a, b]$.

Question author's solution (Python3):

```
a = int(input())
b = int(input())
x = int(input())

if a <= x and x <= b:
    print("É O BRINES, NÃO ADIANTA!")
else:
    print("não foi dessa vez Brines :(")
```

Aviso!

?

Notas para este envio: 1,00/1,00.

Aviso!

?

Questão **2**

Correto

Atingiu 1,00 de 1,00

Área de Figuras Planas III

Implemente uma função chamada **area** que imprime a área de uma determina figura geométrica. A função deve receber 3 parâmetros, sendo dois deles valores numéricos e uma string representando a forma geométrica. A área da figura deve ser do tipo inteiro. As formas geométricas permitidas são retangulo, losango, triangulo e circulo.

Entrada

Os parâmetros da função são dois inteiros $arg1, arg2 \geq 1$ e uma string $forma$. Em caso de retângulo ou triângulo, os argumentos representam a base e a altura da forma, caso a figura seja um losango, os argumentos representam o valor das duas diagonais, e caso a figura geométrica seja círculo, a variável $arg1$ será **sempre** o raio, e a variável $arg2$ **sempre** será igual a 3.

Saída

A função deve imprimir a frase O $forma$ tem $area$ de area

Onde $forma$ é a string inserida, que pode tomar o nome de quatro formas geométricas: **retangulo, losango, triangulo, circulo** e $area$ é somente o valor **inteiro** do cálculo da área da forma geométrica dada na string $forma$, com $arg1, arg2$ assumindo as incógnitas de cada cálculo de área.

Observação

- Caso a figura seja um círculo, utilize $arg2$ no lugar do valor de π, para calcular a área.
- Submeta somente o que foi solicitado.

Particularidade do Tópico

Atenção, a criação de uma função com o nome determinado pelo enunciado é fundamental para a prática do aluno e o Moodle irá descontar pontos caso a criação não tenha sido feita corretamente (sendo *case-sensitive* o nome da função).

For example:

Test	Result

Aviso!

?

Test	Result
area(10, 2, 'losango')	O losango tem 10 de area
area(20, 4, 'retangulo')	O retangulo tem 80 de area
area(15, 3, 'circulo')	O circulo tem 675 de area

Answer: (penalty regime: 0, 0, 10, 20, ... %)

```
1
2  def area(arg1, arg2, forma):
3      area = 0
4      if forma == 'retangulo':
5          area = arg1*arg2
6      elif forma == 'losango' or forma == 'triangulo':
7         area = int(arg1*arg2*0.5)
8      elif forma == 'circulo':
9          area = arg2*arg1**2 #pi*r^2
10     print(f'O {forma} tem {area} de area')
```

	Test	Expected	Got	
✔	area(10, 2, 'losango')	O losango tem 10 de area	O losango tem 10 de area	✔
✔	area(20, 4, 'retangulo')	O retangulo tem 80 de area	O retangulo tem 80 de area	✔
	area(15, 3, 'circulo')	O circulo tem 675 de area	O circulo tem 675 de area	✔

Aviso!

	Test	Expected	Got	
✔	`area(144, 198, 'losango')`	`O losango tem 14256 de area`	`O losango tem 14256 de area`	✔
✔	`area(72, 30, 'triangulo')`	`O triangulo tem 1080 de area`	`O triangulo tem 1080 de area`	✔
✔	`area(194, 193, 'retangulo')`	`O retangulo tem 37442 de area`	`O retangulo tem 37442 de area`	✔

Passou em todos os teste! ✔

Question author's solution (Python3):

```
1 def area(arg1, arg2, forma):
2     area = arg1*arg2
3     if forma == 'triangulo' or forma == 'losango':
4         area //= 2
5     elif forma == 'circulo':
```

Correto

Notas para este envio: 1,00/1,00.

Aviso!

?

Questão 3

Correto

Atingiu 1,00 de 1,00

Termos da sequência v1

Implemente uma função recursiva chamada **`imprimeTermos`** que receba um número inteiro n e imprima os termos da sequência $n, n-2, n-4, \ldots, 0$, isto é, os termos da sequência que começa pelo valor n e termina em 0, decrescendo de 2 em 2 valores.

Entrada

A função recebe como parâmetro um único número inteiro n ($2 \leq n \leq 10^9$).

Saída

Imprima os termos que compõem a sequência estabelecida, sendo que cada termo da sequência $n, n-2, n-4, \ldots, 0$ deve ser impresso em uma única linha.

Observações

Submeta somente o que foi solicitado.

Particularidade do Tópico

Atenção, a criação de uma função com o nome determinado pelo enunciado é fundamental para a prática do aluno e o Moodle irá descontar pontos caso a criação não tenha sido feita corretamente (sendo *case-sensitive* o nome da função).

For example:

Test	Result

`imprimeTermos(8)`	`8`
	`6`
	`4`
	`2`
	`0`

Aviso!

Test	Result
`imprimeTermos(11)`	11 9 7 5 3 1 0
`imprimeTermos(15)`	15 13 11 9 7 5 3 1 0

Answer: (penalty regime: 0, 0, 10, 20, ... %)

```
def imprimeTermos(numero):
    if numero > 0:
        print(numero)
        imprimeTermos(numero-2)
    if numero <= 0:
        print(0)
```

Aviso!

?

	Test	Expected	Got	
✔	`imprimeTermos(8)`	8 6 4 2 0	8 6 4 2 0	✔
✔	`imprimeTermos(11)`	11 9 7 5 3 1 0	11 9 7 5 3 1 0	✔
✔	`imprimeTermos(15)`	15 13 11 9 7 5 3 1 0	15 13 11 9 7 5 3 1 0	✔
✔	`imprimeTermos(123)`	123 121 119 117 115 113 111 109 107 105 103 101 99 97 95 93	123 121 119 117 115 113 111 109 107 105 103 101 99 97 95 93	✔

Aviso!

?

	Test	Expected	Got	
		91 89 87 85 83 81 79 77 75 73 71 69 67 65 63 61 59 57 55 53 51 49 47 45 43 41 39 37 35 33 31 29 27 25 23 21 19 17 15 13 11	91 89 87 85 83 81 79 77 75 73 71 69 67 65 63 61 59 57 55 53 51 49 47 45 43 41 39 37 35 33 31 29 27 25 23 21 19 17 15 13 11	

Aviso!

?

	Test	Expected	Got	
		9 7 5 3 1 0	9 7 5 3 1 0	
✔	imprimeTermos(635)	635 633 631 629 627 625 623 621 619 617 615 613 611 609 607 605 603 601 599 597 595 593 591 589 587 585 583 581 579 577 575 573 571 569	635 633 631 629 627 625 623 621 619 617 615 613 611 609 607 605 603 601 599 597 595 593 591 589 587 585 583 581 579 577 575 573 571 569	✔

Aviso!

?

	Test	Expected	Got	
		567 565 563 561 559 557 555 553 551 549 547 545 543 541 539 537 535 533 531 529 527 525 523 521 519 517 515 513 511 509 507 505 503 501 499 497 495 493 491 489 487	567 565 563 561 559 557 555 553 551 549 547 545 543 541 539 537 535 533 531 529 527 525 523 521 519 517 515 513 511 509 507 505 503 501 499 497 495 493 491 489 487	

Aviso!

?

	Test	Expected	Got	
		485 483 481 479 477 475 473 471 469 467 465 463 461 459 457 455 453 451 449 447 445 443 441 439 437 435 433 431 429 427 425 423 421 419 417 415 413 411 409 407 405	485 483 481 479 477 475 473 471 469 467 465 463 461 459 457 455 453 451 449 447 445 443 441 439 437 435 433 431 429 427 425 423 421 419 417 415 413 411 409 407 405	

Aviso!

?

	Test	Expected	Got	
		403 401 399 397 395 393 391 389 387 385 383 381 379 377 375 373 371 369 367 365 363 361 359 357 355 353 351 349 347 345 343 341 339 337 335 333 331 329 327 325 323	403 401 399 397 395 393 391 389 387 385 383 381 379 377 375 373 371 369 367 365 363 361 359 357 355 353 351 349 347 345 343 341 339 337 335 333 331 329 327 325 323	

Aviso!

?

	Test	Expected	Got	
		321 319 317 315 313 311 309 307 305 303 301 299 297 295 293 291 289 287 285 283 281 279 277 275 273 271 269 267 265 263 261 259 257 255 253 251 249 247 245 243 241	321 319 317 315 313 311 309 307 305 303 301 299 297 295 293 291 289 287 285 283 281 279 277 275 273 271 269 267 265 263 261 259 257 255 253 251 249 247 245 243 241	

Aviso!

?

	Test	Expected	Got	
		239 237 235 233 231 229 227 225 223 221 219 217 215 213 211 209 207 205 203 201 199 197 195 193 191 189 187 185 183 181 179 177 175 173 171 169 167 165 163 161 159	239 237 235 233 231 229 227 225 223 221 219 217 215 213 211 209 207 205 203 201 199 197 195 193 191 189 187 185 183 181 179 177 175 173 171 169 167 165 163 161 159	

Aviso!

?

	Test	Expected	Got	
		157 155 153 151 149 147 145 143 141 139 137 135 133 131 129 127 125 123 121 119 117 115 113 111 109 107 105 103 101 99 97 95 93 91 89 87 85 83 81 79 77	157 155 153 151 149 147 145 143 141 139 137 135 133 131 129 127 125 123 121 119 117 115 113 111 109 107 105 103 101 99 97 95 93 91 89 87 85 83 81 79 77	

Aviso!

?

	Test	Expected	Got	
		75 73 71 69 67 65 63 61 59 57 55 53 51 49 47 45 43 41 39 37 35 33 31 29 27 25 23 21 19 17 15 13 11 9 7 5 3 1 0	75 73 71 69 67 65 63 61 59 57 55 53 51 49 47 45 43 41 39 37 35 33 31 29 27 25 23 21 19 17 15 13 11 9 7 5 3 1 0	

Aviso!

?

	Test	Expected	Got	
✔	`imprimeTermos(2)`	`2` `0`	`2` `0`	✔

Passou em todos os teste! ✔

A função tem como parâmetro um número inteiro. Para imprimir a sequência, deve-se garantir dentro da função que esse inteiro seja maior do que zero e, assim, imprimir o termo e realizar nova chamada dessa mesma função passando como parâmetro valor daquele inteiro subtraído de duas unidades. Caso o valor do número inteiro da função seja menor ou igual a zero, deve-se apenas imprimir zero (e não realizar novas chamadas da função).

Question author's solution (Python3):

```
1 def imprimeTermos(n):
2     if n <= 0:
3         print("0")
4     else:
5         print(n)
```

Correto

Notas para este envio: 1,00/1,00.

Aviso!

?

Questão **4**

Correto

Atingiu 1,00 de 1,00

Área de Figuras Planas I

Implemente uma função chamada **area** que imprime a área de uma determina figura geométrica. A função deve receber 3 parâmetros, sendo dois deles valores numéricos e uma string representando a forma geométrica. A área da figura deve ser do tipo inteiro. As formas geométricas permitidas são "retangulo" e "losango".

Entrada

Os parâmetros da função são dois inteiros $arg1, arg2 \geq 1$ e uma string $forma$. Em caso de retângulo, os argumentos representam a base e a altura da forma e caso a figura seja um losango, os argumentos representam o valor das duas diagonais.

Saída

A função deve imprimir a frase "O $forma$ tem $area$ de area", conforme os exemplos.

$forma$ é a string que pode ter as formas geométricas **retangulo** ou **losango** e $area$ é o valor **inteiro** do cálculo da área da forma geométrica dada na string $forma$, com $arg1, arg2$ assumindo as incógnitas de cada cálculo de área.

Observações

Submeta somente o que foi solicitado.

Particularidade do Tópico

Atenção, a criação de uma função com o nome determinado pelo enunciado é fundamental para a prática do aluno e o Moodle irá descontar pontos caso a criação não tenha sido feita corretamente (sendo *case-sensitive* o nome da função).

For example:

Test	Result
`area(10, 2, 'losango')`	`O losango tem 10 de area`
`4, 'retangulo')`	`O retangulo tem 80 de area`

Aviso!

Test	Result
area(15, 3, 'losango')	O losango tem 22 de area

Answer: (penalty regime: 0, 0, 10, 20, ... %)

```
def area(arg1, arg2, forma):
    area = 0
    if forma == 'retangulo':
        area = arg1*arg2
    elif forma == 'losango':
       area = int(arg1*arg2*0.5)
    print(f'O {forma} tem {area} de area')
```

	Test	Expected	Got	
✔	area(10, 2, 'losango')	O losango tem 10 de area	O losango tem 10 de area	✔
✔	area(20, 4, 'retangulo')	O retangulo tem 80 de area	O retangulo tem 80 de area	✔
✔	area(15, 3, 'losango')	O losango tem 22 de area	O losango tem 22 de area	✔
✔	area(144, 198, 'losango')	O losango tem 14256 de area	O losango tem 14256 de area	✔

Aviso!

?

	Test	Expected	Got	
✔	`area(72, 30, 'retangulo')`	`O retangulo tem 2160 de area`	`O retangulo tem 2160 de area`	✔
✔	`area(194, 193, 'retangulo')`	`O retangulo tem 37442 de area`	`O retangulo tem 37442 de area`	✔

Passou em todos os teste! ✔

Question author's solution (Python3):

```
1 def area(arg1, arg2, forma):
2     area = arg1 * arg2
3     if forma == 'losango':
4         area //= 2
```

Correto

Notas para este envio: 1,00/1,00.

Aviso!

?

Questão **5**

Correto

Atingiu 1,00 de 1,00

O Açaí do Ceubinho

Antes da pandemia, durante uma das maiores secas da história de Brasília, quatro calourinhos decidiram contratar Dêivis para embarcar em uma jornada épica em busca de uma iguaria divina, conhecida apenas por lendas urbanas como O Açaí do Ceubinho.

Como todos os calourinhos ainda não são muito amigos, e portanto não possuem tanta intimidade, eles pediram para Dêivis comprar uma quantidade de copos que fosse divisível por 4, para que cada um pudesse comer o seu sem ter que dividir a refrescância com o coleguinha. O problema é que Dêivis não é muito bom com matemática, ou com realizar tarefas corretamente, então antes de levar O Açaí do Ceubinho, ele decidiu levar pra você decidir se a quantidade de copos de açaí está correta!

Escreva a função **`qtdcopos(n)`** que faça o pedido.

Entrada
A entrada consiste de um inteiro $0 \leq n \leq 100$ que indica quantos copos ele levou para você avaliar.

Saída
Imprima "Pode levar pros calourinhos, deivis!" se a quantidade de copos for divisível por 4. Imprima "Pode voltar pro ceubinho, deivis! Falta(m) x copo(s)!" onde x é o número de copos restantes para que o número se torne divisível por 4, caso não seja possível dar copos para todos os calourinhos.

Observações
Submeta somente o que foi solicitado.

Particularidade do Tópico
Atenção, a criação de uma função com o nome determinado pelo enunciado é fundamental para a prática do aluno e o Moodle irá descontar pontos caso a criação não tenha sido feita corretamente (sendo *case-sensitive* o nome da função).

For example:

Test	Result
`qtdcopos(8)`	`Pode levar pros calourinhos, deivis!`
`(15)`	`Pode voltar pro ceubinho, deivis! Falta(m) 1 copo(s)!`

Aviso!

?

Test	Result
qtdcopos(0)	Pode voltar pro ceubinho, deivis! Falta(m) 4 copo(s)!

Answer: (penalty regime: 0, 0, 10, 20, ... %)

```
def qtdcopos(n):
    if n % 4 == 0 and n > 0:
        print('Pode levar pros calourinhos, deivis!')
    else:
        qtd = 4 * (n//4 + 1) - n
        print(f'Pode voltar pro ceubinho, deivis! Falta(m) {qtd} copo(s)!')

```

	Test	Expected	Got	
✔	qtdcopos(8)	Pode levar pros calourinhos, deivis!	Pode levar pros calourinhos, deivis!	✔
✔	qtdcopos(15)	Pode voltar pro ceubinho, deivis! Falta(m) 1 copo(s)!	Pode voltar pro ceubinho, deivis! Falta(m) 1 copo(s)!	✔
✔	qtdcopos(0)	Pode voltar pro ceubinho, deivis! Falta(m) 4 copo(s)!	Pode voltar pro ceubinho, deivis! Falta(m) 4 copo(s)!	✔
✔	qtdcopos(32)	Pode levar pros calourinhos, deivis!	Pode levar pros calourinhos, deivis!	✔

Aviso!

?

	Test	Expected	Got	
✔	`qtdcopos(45)`	`Pode voltar pro ceubinho, deivis! Falta(m) 3 copo(s)!`	`Pode voltar pro ceubinho, deivis! Falta(m) 3 copo(s)!`	✔
✔	`qtdcopos(1)`	`Pode voltar pro ceubinho, deivis! Falta(m) 3 copo(s)!`	`Pode voltar pro ceubinho, deivis! Falta(m) 3 copo(s)!`	✔

Passou em todos os teste! ✔

Question author's solution (Python3):

```
1 def qtdcopos(x):
2     if x > 0 and x % 4 == 0:
3         print('Pode levar pros calourinhos, deivis!')
4     else:
```

Correto

Notas para este envio: 1,00/1,00.

Aviso!

?

Questão **6**

Correto

Atingiu 1,00 de 1,00

Piscininha

Agora que o Prof. Nerynho já construiu sua piscina, ele está testando um programa de localização pessoal que diz o que Nerynho está fazendo na área de sua piscina, baseado em uma representação cartesiana vista de um satélite!!!! Ajude o Prof Nerynho a desenvolver seu programa, escrevendo a função **`piscininha(x, y, w, h, a, b)`** cujas variáveis são:

- As primeiras quatro variáveis, $x, y, w, h \mid w, h \geq 2$ representam a piscina, que é um retângulo de altura h e largura w, alinhado aos eixos cartesianos **X** e **Y**, cujo vértice inferior esquerdo está no ponto (x, y).
- As duas ultimas variáveis, a e b, representam as coordenadas cartesianas (a, b) de onde o Prof Nerynho está.

Entrada
Os parâmetros da função são seis inteiros x, y, w, h, a, b.

Saída
A saída depende da posição do Prof. Nerynho baseado em onde está sua piscina e:

- Caso o professor esteja dentro da piscina, o programa deverá imprimir a frase **Dando um tchibum**;
- Caso o professor esteja fora da piscina, o programa deverá imprimir a frase **Tomando um solzin**;
- Caso o professor esteja na borda da piscina, o programa deverá imprimir a frase **So com os pezin dentro da agua**.

Observações
Submeta somente o que foi solicitado.

Particularidade do Tópico
Atenção, a criação de uma função com o nome determinado pelo enunciado é fundamental para a prática do aluno e o Moodle irá descontar pontos caso a criação não tenha sido feita corretamente (sendo *case-sensitive* o nome da função).

For example:

Test	Result
`piscininha(0, 0, 10, 10, 0, 5)`	`So com os pezin dentro da agua`
`piscininha(0, 0, 10, 10, 1, 1)`	`Dando um tchibum`

Aviso!

?

Test	Result
`piscininha(-10, -15, 5, 4, 0, 0)`	`Tomando um solzin`

Answer: (penalty regime: 0, 0, 10, 20, ... %)

```
def piscininha(x, y, w, h, a, b):

    #borda esquerda: x, borda direita: x+w
    #borda inferior: y, borda superior: y+h
    if (a >= x and a <= x + w) and (b >= y and b <= y + h): #verifica se está nos limites
        if a == x or a == x + w or b == y or b == y + h: # verifica se está na borda
            print('So com os pezin dentro da agua')
        else:
            print('Dando um tchibum')
    else:
        print('Tomando um solzin')

```

	Test	Expected	Got	
✔	`piscininha(0, 0, 10, 10, 0, 5)`	`So com os pezin dentro da agua`	`So com os pezin dentro da agua`	✔
✔	`piscininha(0, 0, 10, 10, 1, 1)`	`Dando um tchibum`	`Dando um tchibum`	✔
✔	`piscininha(-10, -15, 5, 4, 0, 0)`	`Tomando um solzin`	`Tomando um solzin`	✔
✔	`piscininha(-1, -11, 961103, 2, -1, -10)`	`So com os pezin dentro da agua`	`So com os pezin dentro da agua`	✔

Aviso!

	Test	Expected	Got	
✔	`piscininha(-1, -1847, 2, 7, -189999982, -1846)`	`Tomando um solzin`	`Tomando um solzin`	✔
✔	`piscininha(100000000, -100000000, 2, 1843, 100000002, 472000855)`	`Tomando um solzin`	`Tomando um solzin`	✔

Passou em todos os teste! ✔

Question author's solution (Python3):

```
1 def piscininha(x, y, w, h, a, b):
2     if (a > x) and (a < w + x) and (b > y) and (b < y + h):
3         print('Dando um tchibum')
4     elif (a < x) or (a > w + x) or (b < y) or (b > y + h):
5         print('Tomando um solzin')
```

Correto

Notas para este envio: 1,00/1,00.

Aviso!

?

Questão 7

Parcialmente correto

Atingiu 0,86 de 1,00

Múltiplos

Uma operação bastante comum dados dois números a e b é saber se a é um múltiplo de b ou não. Formalmente, um inteiro a é múltiplo de um inteiro b se existe um inteiro k tal que $b * k = a$.

Nesse exercício, você deve criar uma função chamada **`multiple`** , que recebe dois inteiros, a e b. A função deve imprimir na tela "a eh multiplo de b", caso a seja multiplo de b; "b eh multiplo de a" caso b seja multiplo de a; "nao multiplos" em quaisquer outros casos. Substitua a e b pelo valor dos argumentos para imprimir. Imprima SEM as aspas. Veja os exemplos.

Dica: Python possui o operador '%' (módulo). Ela retorna o resto da divisão inteira de um número por outro (e.g. `2 % 10` dá como resultado o resto da divisão de 2 por 10).

Entrada

A entrada consiste nos parâmetros da função **`multiple,`** que são os dois números inteiros a e b ($-10000 \leq a, b \leq 10000$).

Saída

A função deve imprimir na tela "a eh multiplo de b", caso a seja multiplo de \(b\); "b eh multiplo de a" caso b seja multiplo de a; "nao multiplos" em quaisquer outros casos. Substitua a e b pelo valor dos argumentos para imprimir. Imprima SEM as aspas.

Observações

- Nos 2 primeiros casos de teste, 6 é múltiplo de 2, então a função imprime "6 eh multiplo de 2".
- No terceiro caso de teste, nem 21 é múltiplo de 22, nem 22 de 21, então a função imprime "nao multiplos".
- Submeta somente o que foi solicitado.

For example:

Test	Result
`multiple(2, 6)`	`6 eh multiplo de 2`
`(6, 2)`	`6 eh multiplo de 2`

Aviso!

?

Test	Result
multiple(21, 22)	nao multiplos

Answer: (penalty regime: 0, 0, 10, 20, ... %)

```
def multiple(a, b):
    if a >= b and a % b == 0:
        print(f'{a} eh multiplo de {b}')
    elif a < b and b % a == 0:
        print(f'{b} eh multiplo de {a}')
    else:
        print('nao multiplos')
```

	Test	Expected	Got	
✔	multiple(2, 6)	6 eh multiplo de 2	6 eh multiplo de 2	✔
✔	multiple(6, 2)	6 eh multiplo de 2	6 eh multiplo de 2	✔
✔	multiple(21, 22)	nao multiplos	nao multiplos	✔
✔	multiple(-2, 6)	6 eh multiplo de -2	6 eh multiplo de -2	✔

Aviso!

?

	Test	Expected	Got	
✔	`multiple(-3, 41)`	`nao multiplos`	`nao multiplos`	✔
✔	`multiple(200, 10000)`	`10000 eh multiplo de 200`	`10000 eh multiplo de 200`	✔

Your code failed one or more hidden tests.

Se a é múltiplo de b, então a divisão inteira de a por b (e.g. $\frac{a}{b}$ também é um número inteiro, e portanto tem resto 0.
Logo, basta testar o resto da divisão de a por \(b\) com o operador módulo (%). Se o resultado de a for 0, então a é múltiplo de b.

Temos 3 possibilidades, conforme diz o enunciado. Então criamos uma estrutura if / elif / else para acomodar todas elas. Na primeira testamos se a é múltiplo de b, em seguida testamos se b é múltiplo de a. Se nenhum desses for verdade, então não são múltiplos.

Question author's solution (Python3):

```
#Colocar aqui a solucao
def multiple(a, b):
    if a % b == 0:
        print(f'{a} eh multiplo de {b}')
    elif b % a == 0:
        print(f'{b} eh multiplo de {a}')
```

Parcialmente correto

Notas para este envio: 0,86/1,00.

Aviso!

?

Questão 8

Correto

Atingiu 1,00 de 1,00

Triângulo Válido 2

Crie uma função chamada **valid_triangle**. Ela deve receber como argumentos 3 números naturais a, b e c, os lados de um triângulo.

Lembre que triângulos equiláteros são aqueles que possuem 3 lados iguais; isósceles possuem 2 lados iguais; escalenos possuem os 3 lados com medidas diferentes.

Entrada

A entrada consiste nos parâmetros da função **`valid_triangle,`** que são os três números naturais a, b e c.

Saída

- Sua função deve imprimir **equilatero** caso o triângulo seja válido e um triângulo equilátero.
- Imprima **isosceles** caso o triângulo seja válido e isósceles.
- Imprima **escaleno** caso o triângulo seja válido e escaleno.
- Em qualquer outro caso, imprima **invalido**.

Observações

- No primeiro caso de teste, é possível criar um triângulo com lados 4, 3, 5. Como o triângulo é válido e todos os lados são diferentes, a função imprime **escaleno**.
- No segundo caso não é possível criar um triângulo com lados 3, 1, 2. Logo a função imprime **invalido.**
- No terceiro caso o triângulo com lados 5, 5, 5 é válido e equilátero. Então a função imprime **equilatero.**
- Imprima somente o que foi solicitado

Particularidade do Tópico

Atenção, a criação de uma função com o nome determinado pelo enunciado é fundamental para a prática do aluno e o Moodle irá descontar pontos caso a criação não tenha sido feita corretamente (sendo *case-sensitive* o nome da função).

For example:

Test	Result
`iangle(4,3,5)`	`escaleno`

Aviso!

Test	Result
valid_triangle(3,1,2)	invalido
valid_triangle(5,5,5)	equilatero

Answer: (penalty regime: 0, 0, 10, 20, ... %)

```
def valid_triangle(a, b, c):
    isValido = (a<b+c) and (b<a+c) and (c<a+b)
    if not isValido:
        print('invalido')
    else:
        if a == b and b == c:
            print('equilatero')
        elif a == b or a == c or b== c:
            print('isosceles')
        else:
            print('escaleno')
```

	Test	Expected	Got	
✔	valid_triangle(4,3,5)	escaleno	escaleno	✔
✔	valid_triangle(3,1,2)	invalido	invalido	✔
✔	valid_triangle(5,5,5)	equilatero	equilatero	✔
	lid_triangle(6,12,6)	invalido	invalido	✔

Aviso!

?

	Test	Expected	Got	
✔	`valid_triangle(100,200,250)`	`escaleno`	`escaleno`	✔
✔	`valid_triangle(12, 12, 15)`	`isosceles`	`isosceles`	✔

Passou em todos os teste! ✔

O primeiro passo é testar se o triângulo é válido ou não.
Um triângulo é válido se a soma de 2 lados é maior do que o terceiro lado.
Logo, temos que testar 3 condições:

```
1.a + b > c
2.a + c > b
3.b + c > a
```

Apenas se TODAS essas condições forem verdadeiras temos um triângulo válido. Caso seja inválido, já imprimimos inválido.

Em seguida temos que testar se o triângulo é equilátero, isósceles ou escaleno.

Para isso vemos se os 3 lados são iguais (equilátero), se 2 lados quaisquer são iguais (isósceles). Se nenuma dessas alternativas for verdadeira, o triângulo é escaleno.

Dica: você pode utilizar operações lógicas para testar múltiplas expressões em um mesmo `if`. Por exemplo, para testar se 2 lados quaisquer são iguais pode-se usar `a == b or a == c or b == c`

. As operações lógicas em Python são `and`, `or` e `not`.

Question author's solution (Python3):

```python
#Colocar aqui a solucao

def valid_triangle(a, b, c):
    if a + b > c and a + c > b and b + c > a:
        if a == b and a == c:
            print('equilatero')
        elif a == b or a == c or b == c:
            print('isosceles')
        else:
```

Aviso!

```
10              print('escaleno')
```

Correto

Notas para este envio: 1,00/1,00.

Aviso!

?

Questão **9**

Correto

Atingiu 1,00 de 1,00

Gondim

Gondim está aprendendo geometria. Ele fez uma grande lista de triplas de números e agora está tentando classificar cada tripla, determinando suas propriedades em relação a triângulos. Para terminar mais rápido, ele pediu a sua ajuda. Ele precisa que você escreva uma função chamada **`classificador`** que tem como parâmetro três inteiros.

Entrada

Os parâmetros da função são três inteiros a, b, c.

Saída

Imprima uma linha para cada propriedade que a tripla de números satisfaz, na ordem indicada abaixo.

- triangulo – A soma de qualquer par é maior que o número restante da tripla.
- gondim sendo gondim – Não satisfaz a propriedade triangulo.
- escaleno – Satisfaz a propriedade triangulo e os três lados são distintos entre si.
- isosceles – Satisfaz a propriedade triangulo e pelo menos dois lados são iguais.
- equilatero – Satisfaz a propriedade triangulo e os três lados são iguais.
- retangulo – Satisfaz a propriedade triangulo e o quadrado do maior lado é a soma dos quadrados dos outros lados.

Caso o triângulo possua mais de uma das propriedades estas devem ser listadas **na ordem apresentada acima**, caso o triângulo seja um triangulo isosceles e retangulo a seguinte saída é considerada errada:

```
triangulo
retangulo
isosceles
```

Enquanto a seguinte saída está correta:

```
triangulo
isosceles
retangulo
```

(ões

Aviso!

?

- No primeiro caso de teste, a figura geométrica tem lados a = 3, b = 5 e c = 4. Gondim chegou a conclusão q a figura é um triângulo e satisfaz as propriedades para ser classificado como escaleno e também como retângulo.
- No segundo caso de teste, a figura geométrica tem lados a = 2, b = 1 e c = 1. Gondim percebeu que essa figura não pode ser um triângulo!
- No terceiro caso de teste, a figura geométrica tem lados a = 3, b = 3 e c = 3. Gondim chegou a conclusão q a figura é um triângulo e satisfaz as propriedades para ser classificado como isósceles e também como equilátero.
- Submeta somente o que foi solicitado.

Particularidade do Tópico

Atenção, a criação de uma função com o nome determinado pelo enunciado é fundamental para a prática do aluno e o Moodle irá descontar pontos caso a criação não tenha sido feita corretamente (sendo *case-sensitive* o nome da função).

For example:

Test	Result

`classificador(3, 5, 4)`	`triangulo`
	`escaleno`
	`retangulo`

`classificador(2, 1, 1)`	`gondim sendo gondim`

`classificador(3, 3, 3)`	`triangulo`
	`isosceles`
	`equilatero`

Answer: (penalty regime: 0, 0, 10, 20, ... %)

```
1
2 def classificador(a, b, c):
3     isTriangulo = (a<b+c) and (b<a+c) and (c<a+b)
4     if not isTriangulo:
5         print('gondim sendo gondim')
6     else:
7         print('triangulo')
8         if a != b and b!= c and a!= c:
9             print('escaleno')
10        if a == b or a == c or b == c:
```

Aviso!

```
11              print('isosceles')
12          if a == b and b == c:
13              print('equilatero')
14          if max(a, b, c) == a:
15              if a**2 == b**2 + c**2:
```

	Test	Expected	Got	
✔	`classificador(3, 5, 4)`	triangulo escaleno retangulo	triangulo escaleno retangulo	✔
✔	`classificador(2, 1, 1)`	gondim sendo gondim	gondim sendo gondim	✔
✔	`classificador(3, 3, 3)`	triangulo isosceles equilatero	triangulo isosceles equilatero	✔
✔	`classificador(5, 1, 5)`	triangulo isosceles	triangulo isosceles	✔
✔	`classificador(209, 1, 1)`	gondim sendo gondim	gondim sendo gondim	✔
✔	`classificador(604317, 593745, 1000000)`	triangulo escaleno	triangulo escaleno	✔

Passou em todos os teste! ✔

Question author's solution (Python3):

```
1  def classificador(a, b, c):
2      if a + b > c and a + c >  b and b + c  > a:
3          print("triangulo")
4
5          if a != b and b != c and c != a:
6              print("escaleno")
```

Aviso!

```
    elif a == b or b == c or c == a:
        print("isosceles")

    if a == b and b == c:
        print("equilatero")

    if a*a == b*b+c*c or b*b==a*a+c*c or c*c==a*a+b*b:
```

Correto

Notas para este envio: 1,00/1,00.

Aviso!

?

Questão **10**

Correto

Atingiu 1,00 de 1,00

Sequências de pares v1

Implemente uma função recursiva chamada **paresDeNumeros** que receba dois números inteiros n e m, em que $n < m$. A função deve imprimir pares de números inteiros $n_i \quad m_j$, em que $n_i = n_{i-1} + 1$ e $m_j = m_{j+1} - 1$, até que $n_i \leq m_j$. Considere que $n_0 = n$ e $m_0 = m$.

Entrada

A função **paresDeNumeros** recebe como parâmetros dois números inteiros positivos n e m ($1 \leq n \leq 10^9 - 1, n < m \leq 10^9$).

Saída

Imprima os termos que compõem a sequência de pares de números supracitada, sendo que cada par de números deve ser impresso em uma única linha e esses números devem estar separados por um espaço em branco.

Observações

Submeta somente o que foi solicitado.

Particularidade do Tópico

Atenção, a criação de uma função com o nome determinado pelo enunciado é fundamental para a prática do aluno e o Moodle irá descontar pontos caso a criação não tenha sido feita corretamente (sendo *case-sensitive* o nome da função).

For example:

Test	Result

`paresDeNumeros(1,6)`	`1 6`
	`2 5`
	`3 4`

`paresDeNumeros(3,7)`	`3 7`
	`4 6`
	`5 5`

Aviso!

Test	Result
paresDeNumeros(8,10)	8 10 9 9

Answer: (penalty regime: 0, 0, 10, 20, ... %)

```
def paresDeNumeros(n, m):
    if m >= n:
        print(n,m)
        paresDeNumeros(n+1, m-1)

```

	Test	Expected	Got	
✔	paresDeNumeros(1,6)	1 6 2 5 3 4	1 6 2 5 3 4	✔
✔	paresDeNumeros(3,7)	3 7 4 6 5 5	3 7 4 6 5 5	✔
	paresDeNumeros(8,10)	8 10 9 9	8 10 9 9	✔

Aviso!

?

	Test	Expected	Got	
✔	paresDeNumeros(200,213)	200 213 201 212 202 211 203 210 204 209 205 208 206 207	200 213 201 212 202 211 203 210 204 209 205 208 206 207	✔
✔	paresDeNumeros(513,688)	513 688 514 687 515 686 516 685 517 684 518 683 519 682 520 681 521 680 522 679 523 678 524 677 525 676 526 675 527 674 528 673 529 672 530 671 531 670 532 669 533 668 534 667 535 666 536 665 537 664 538 663 539 662 540 661 541 660 542 659 543 658 544 657	513 688 514 687 515 686 516 685 517 684 518 683 519 682 520 681 521 680 522 679 523 678 524 677 525 676 526 675 527 674 528 673 529 672 530 671 531 670 532 669 533 668 534 667 535 666 536 665 537 664 538 663 539 662 540 661 541 660 542 659 543 658 544 657	✔

Aviso!

?

	Test	Expected	Got	
		545 656 546 655 547 654 548 653 549 652 550 651 551 650 552 649 553 648 554 647 555 646 556 645 557 644 558 643 559 642 560 641 561 640 562 639 563 638 564 637 565 636 566 635 567 634 568 633 569 632 570 631 571 630 572 629 573 628 574 627 575 626 576 625 577 624 578 623 579 622 580 621 581 620 582 619 583 618 584 617 585 616	545 656 546 655 547 654 548 653 549 652 550 651 551 650 552 649 553 648 554 647 555 646 556 645 557 644 558 643 559 642 560 641 561 640 562 639 563 638 564 637 565 636 566 635 567 634 568 633 569 632 570 631 571 630 572 629 573 628 574 627 575 626 576 625 577 624 578 623 579 622 580 621 581 620 582 619 583 618 584 617 585 616	

Aviso!

?

	Test	Expected	Got	
		586 615 587 614 588 613 589 612 590 611 591 610 592 609 593 608 594 607 595 606 596 605 597 604 598 603 599 602 600 601	586 615 587 614 588 613 589 612 590 611 591 610 592 609 593 608 594 607 595 606 596 605 597 604 598 603 599 602 600 601	
✔	paresDeNumeros(2,4)	2 4 3 3	2 4 3 3	✔

Passou em todos os teste! ✔

A chamada da função deve receber dois números inteiros como parâmetros: n e m. Deve-se imprimir os valores de n e m somente se n for menor ou igual a m para depois realizar uma nova chamada da função, passando os valores $n + 1$ e $m - 1$ como argumentos.

Question author's solution (Python3):

```
1 def paresDeNumeros(n,m):
2     if(n <= m):
3         print(str(n)+" "+str(m))
```

Correto

Notas para este envio: 1,00/1,00.

Aviso!

?

Aviso!