مسائل تمرینی توابع

مسائل به ترتیب افزایش درجه سختی مرتب شده اند:

- دست گرمی - اینها می توانند با استفاده از مقایسه های پایه و روش های ساده حل شوند
- سطح 1 - اینها ممکن است شامل عبارات شرطی if / then و روش های ساده باشد
- سطح 2 - اینها ممکن است نیاز به تکرار بر روی دنباله ها داشته باشد ، معمولاً با نوعی حلقه
- چالش برانگیز - حل این مسائل نیاز به خلاقیت دارد

دست گرمی:

کمتر از دو زوج: تابعی بنویسید که از میان دو عدد داده را بعنوان ورودی دریافت می کند و * اگر * هر دو عدد زوج باشند مقدار کوچکتر را بر میگرداند ، اما اگر حداقل یکی از اعداد فرد باشد مقدار بزرگتر را برمی گرداند

```
lesser_of_two_evens (2،4) -> 2
lesser_of_two_evens (2،5) -> 5
```

In [ ]:

```
1
```

In [ ]:

```
1 # Check
2 lesser_of_two_evens(2, 4)
```

In [ ]:

```
1 # Check
2 lesser_of_two_evens(2, 5)
```

ANIMAL CRACKERS: تابعی بنویسید که یک رشته دو کلمه ایی از اسامی حیوانات را بعنوان ورودی دریافت کند و اگر حرف اول دو کلمات داخل رشته با یکدیگر برابر باشد، مقدار `True` را برمیگرداند

```
animal_crackers('Levelheaded Llama') --> True
animal_crackers('Crazy Kangaroo') --> False
```

In [ ]:

```
1
```

In [ ]:

```
# Check
animal_crackers('Levelheaded Llama')
```

In [ ]:

```
# Check
animal_crackers('Crazy Kangaroo')
```

MAKES TWENTY: تابعی بنویسید که دو عدد را بعنوان ورودی دریافت می کند و اگر یکی از اعداد 20 باشد یا مجموعه دو عدد 20 باشد مقدار `True` را برگرداند.

```
makes_twenty(20,10) --> True
makes_twenty(12,8) --> True
makes_twenty(2,3) --> False
```

In [ ]:

```

```

In [ ]:

```
# Check
makes_twenty(20,10)
```

In [ ]:

```
# Check
makes_twenty(12,8)
```

In [ ]:

```
#Check
makes_twenty(2,3)
```

مسائل سطح 1

OLD MACDONALD: تابعی بنویسید تا یک رشته را بعنوان ورودی دریافت کند و اگر طول رشته بزرگتر از 3 باشد حرف اول و چهارم را با حروف بزرگ انگلیسی بنویسد

```
old_macdonald('macdonald') --> MacDonald
```

'Note: `'macdonald'.capitalize()` returns 'Macdonald

In [ ]:

```
1
```

In [ ]:

```
1 # Check
2 old_macdonald('macdonald')
```

In [ ]:

```
1 old_macdonald('ma')
```

MASTER YODA: تابعی بنویسید تا یک رشته را دریافت نمایید و ترتیب کلمات را در آن مطابق مثال های زیر معکوس نماید.

```
master_yoda('I am home') --> 'home am I'
master_yoda('We are ready') --> 'ready are We'
```

In [ ]:

```
1
```

In [ ]:

```
1 # Check
2 master_yoda('I am home')
```

In [ ]:

```
1 # Check
2 master_yoda('We are ready')
```

ALMOST THERE: تابع بنویسید که یک عدد را دریافت کند و اگر در فاصله 10 تا از 100 یا 200 باشد `True` و در غیر اینصورت `False` را برگرداند

```
almost_there(90) --> True
almost_there(104) --> True
almost_there(150) --> False
almost_there(209) --> True
```

NOTE: `abs(num)` returns the absolute value of a number

In [ ]:

```
1
```

In [ ]:

```
# Check
almost_there(90)
```

In [ ]:

```
# Check
almost_there(104)
```

In [ ]:

```
# Check
almost_there(150)
```

In [ ]:

```
# Check
almost_there(209)
```

مسائل سطح 2

FIND 33:

تابعی بنویسید که لیستی از اعداد صحیح را دریافت نماید و اگر دو عضو با مقدار 3 پشت سرهم پیدا نماید مقدار `True` را برگرداند و در غیر اینصورت مقدار `False`

```
has_33([1, 3, 3]) → True
has_33([1, 3, 1, 3]) → False
has_33([3, 1, 3]) → False
```

In [ ]:

```

```

In [ ]:

```
# Check
has_33([1, 3, 3])
```

In [ ]:

```
# Check
has_33([1, 3, 1, 3])
```

In [ ]:

```
# Check
has_33([3, 1, 3])
```

PAPER DOLL: تابعی بنویسید که یک رشته را بعنوان ورودی دریافت نماید و یک رشته جدید بعنوان خروجی برگرداند که در آن به ازای هر حرف در رشته اصلی سه بار در رشته جدید تکرار شده باشد.

```
paper_doll('Hello') --> 'HHHeeellllllooo'
paper_doll('Mississippi') --> 'MMMiiissssssiiippppppiii'
```

In [ ]:

```

```

In [ ]:

```
# Check
paper_doll('Hello')
```

In [ ]:

```
# Check
paper_doll('Mississippi')
```

BLACKJACK: تابعی بنویسید که سه عدد صحیح در بازه 1 تا 11 دریافت کند در صورتیکه مجموعه اعداد کمتر مساوی 21 باشد مقدار مجموع را برگرداند، در صورتیکه مجموعه بیشتر از 21 باشد ؛ مجموع را منهای 10 کند اگر مقدار جدید کمتر از 21 باشد و در میان اعداد 11 باشد، مجموع جدید آن را گزارش کند و در غیراینصورت عبارت `Bust` را چاپ کند.

```
blackjack(5,6,7) --> 18
blackjack(9,9,9) --> 'BUST'
blackjack(9,9,11) --> 19
```

In [ ]:

```

```

In [ ]:

```
# Check
blackjack(5,6,7)
```

In [ ]:

```
# Check
blackjack(9,9,9)
```

In [ ]:

```
# Check
blackjack(9,9,11)
```

SUMMER OF '69: تابعی بنویسید که یک لیست از اعداد صحیح را بعنوان ورودی دریافت کند و مجموع اعداد را با در نظر گرفتن شرایط مقابل محاسبه نماید، اگر در هنگام محاسبه مجموع به عضوی در لیست با مقدار 6 برسیم خود آن عضو و اعضای بعدی را رسید به عضوی که مقدار آن 9 است در مجموع لحاظ نخواهیم کرد. (در لیست ورودی حتما پس از 6 باید حداقل یک 9 بیاید)

```
summer_69([1, 3, 5]) --> 9
summer_69([4, 5, 6, 7, 8, 9]) --> 9
summer_69([2, 1, 6, 9, 11]) --> 14
```

In [ ]:

```

```

In [ ]:

```
# Check
summer_69([1, 3, 5])
```

In [ ]:

```
# Check
summer_69([4, 5, 6, 7, 8, 9])
```

In [ ]:

```
# Check
summer_69([2, 1, 6, 9, 11])
```

مسائل چالش برانگیز

SPY GAME: تابعی بنویسید تا لیست از اعداد صحیح را بعنوان ورودی دریافت نماید و اگر در میان اعضا بتوان سه عضو پشت سرهم یا با فاصله پیدا نماید که الگوی `007` را ایجاد کند مقدار `True` را برگرداند در غیر اینصورت `False` را برگرداند

```
spy_game([1,2,4,0,0,7,5]) --> True
spy_game([1,0,2,4,0,5,7]) --> True
```

In [ ]:

```
1
```

In [ ]:

```
1 # Check
2 spy_game([1,2,4,0,0,7,5])
```

In [ ]:

```
1 # Check
2 spy_game([1,0,2,4,0,5,7])
```

In [ ]:

```
1 # Check
2 spy_game([1,7,2,0,4,5,0])
```

COUNT PRIMES: تابعی بنویسید که یک عدد صحیح را بعنوان ورودی دریافت کند و لیست اعداد اول تا خود عدد ورودی را چاپ وتعداد آنها را چاپ کند

```
count_primes(100) --> 25
```

.By convention, 0 and 1 are not prime

In [ ]:

```
1
2
```

In [ ]:

```
1 # Check
2 count_primes(100)
```

جایزه: راه حل سریعتر که اعداد اول را در طول مسیر پیمایش جمع آوری می نماید!

In [ ]:

```
1
```

In [ ]:

```
1 count_primes2(100)
```

صرفا جهت فان :)

PRINT BIG: تابعی بنویسید که یک حرف انگلیسی را دریافت کند و آن را با حرف بزرگ و با استفاده از علامت های * چاپ کند.

```
print_big('a')

out:   *
      * *
     *****
     *   *
     *   *
```

"راهنما: در نظر داشته باشید که یک فرهنگ لغت از الگوهای ممکن ایجاد کنید و حروف الفبا را به ترکیبات خاص 5 خطی از الگوهای مرتبط نگاشت کنید. برای اهداف این تمرین، اگر فرهنگ لغت شما در "E" متوقف شود، مشکلی نیست."

In [ ]:

```
1
```

In [ ]:

```
1 print_big('c')
```

عالی!