递归获取数组的大小。

这是独立写出的第一个递归，纪念一下。终于有点感觉了。

注意：

- 1、空数组是没有第一个元素。即不能这样写：

```
$arr = [];
$arr[0];
会报错，Undefined offset: 0...
```

- 2、**先保存**原数组的第一个元素，**再出队**。即：

```
$first = $arr[0];
array_shift($arr);
return $first + getSize($arr);
```

错误的写法：

```
array_shift($arr);
return $arr[0] + getSize($arr);
假如$arr为[10000,20000];
经过array_shift($arr)处理之后，$arr为[20000];
这时候$arr[0]就为20000，而不是预期的10000；
```

- 3、数组$arr = [[ ],[ ]];不为空。

代码

```php
<?php
function getSize($arr)
{
    //三问之第一问：参数是否为空。
    if (empty($arr)) {
        return 0;
    }
    //三问之第二问：参数的第一个元素是否为数组。
    if (!is_array($arr[0])) {
        // 注意这里的顺序，先保存参数的第一个元素。
        $first = $arr[0];
        // 再出队。
        array_shift($arr);
        return $first + getSize($arr);
    } else {
        // 三问之第三问：else。
        $e = array_shift($arr);
        return getSize($e) + getSize($arr);
    }
}
$arr = [
            [
                10000,
                20000
                ],
            [],
            []
        ];
$re = getSize($arr);
echo $re;
// 输出：30000;
```

推演：

```
getSize([10000, 20000]) + getSize([[],[]]);

(10000 + getSize([20000])) + (getSize([]) + getSize([]));

(10000 + 20000 + getSize([])) + (0 + 0);
 (10000 + 20000 + 0) + (0 + 0);
```

note路径：

- D:\phpStudy\PHPTutorial\WWW\index\01\pattern\Composite\note\getSize.md