

The Art of Googling

Learning Python is a lot like learning a *natural language* like Japanese or French in a number of important ways.

- We learn best by **doing**; by trying to solve particular problems.
- Reading the dictionary (or the [Python documentation](#)) is only useful when combined with trying to solve problems in the language.
- Different students will need to look up different things. An English student learning French will have to look up different words than an Italian student learning French.

Sometimes the *best* way of learning is also the quickest way of solving the problem: a well-worded **Google search**. This might seem like a hacker's thing to be doing, but in fact, all professional and experienced programmers and data scientists use Google [even more than beginners do](#).

The lesson is clear: **being able to Google properly** is a sign of the *pragmatic ingenuity* integral to being a competent programmer and data scientist.

But what counts as Googling properly? Over time, you'll certainly get an intuition for how to word good Google searches and easily identify what constitutes a good resource. Plus, in this article and in the coming Python lessons, we'll ensure that each time we give you a resource we recommend, we'll tell you what would have been the Google search to find it. Sometimes, we'll also tell you what you would have needed to Google

to find out certain tips and insights we give you. In this way, you'll develop your own judgment, self-reliance, and programming abilities.

Here are some great [Googling tips](#) to get you started:

- Use words like *get*, *set* and *check* in your search, e.g: 'get last two characters of string python', or 'check if list is empty python'.
- Always specify the language in question (Python)
- Less is more: avoid words that don't matter. For simple searches, don't worry about punctuation. For example, here's a bad way to Google search the above: 'how do I retrieve the last two remaining characters of my string 'aardvark'".
- When you find a solution, *abstract* from the lessons you learn or the examples you find to your case. This may entail copying and pasting sample code and changing variable names.

Googling Example

Let's look at an example. Suppose we have a variable storing the number of musical instruments in our instrument rental store:

```
instruments = 6
```

Suppose we need to check whether we have an even number of instruments for rental bookings, and we didn't know how to use Python to do this.

Well we *could* open up a Python textbook, or go straight to the Python documentation. But the chances are that these methods would only give us what we need after a lot of tedious research.

Instead, we can solve this quickly with a well-worded Google search. Following the tips we gave above, if we wrote simply: [check if number is even python](#) then we get a list of very useful pages:



[All](#)
[Images](#)
[Videos](#)
[News](#)
[Shopping](#)
[More](#)
[Settings](#)
[Tools](#)

About 235,000,000 results (0.64 seconds)

If you divide a **number** by 2 and it gives a remainder of 0 then it is known as **even number**, otherwise an odd **number**.

...

Python Program to Check if a Number is Odd or Even

1. num = int(input("Enter a **number**: "))
2. if (num % 2) == 0:
3. print("{0} is **Even number**".format(num))
4. else:
5. print("{0} is Odd **number**".format(num))

Python Program to Check if a Number is Odd or Even

<https://www.javatpoint.com/python-check-number-is-odd-or-even>



[About Featured Snippets](#)
[Feedback](#)

Python Program to Check if a Number is Odd or Even

<https://www.programiz.com/python-programming/examples/odd-even>

The first resource we retrieved from this search was a www.javatpoint.com page. Now, even though this website started life as a resource for Java programmers, it does contain helpful pages for other programming languages too. The entry here is no exception and would suffice for our purposes.

Let's take a quick look at the second resource that's output by our search. The website here is www.programiz.com, which has plenty of articles, tutorials, and tips on Python problems from the general to the specific. Again, this resource would be adequate for our needs.

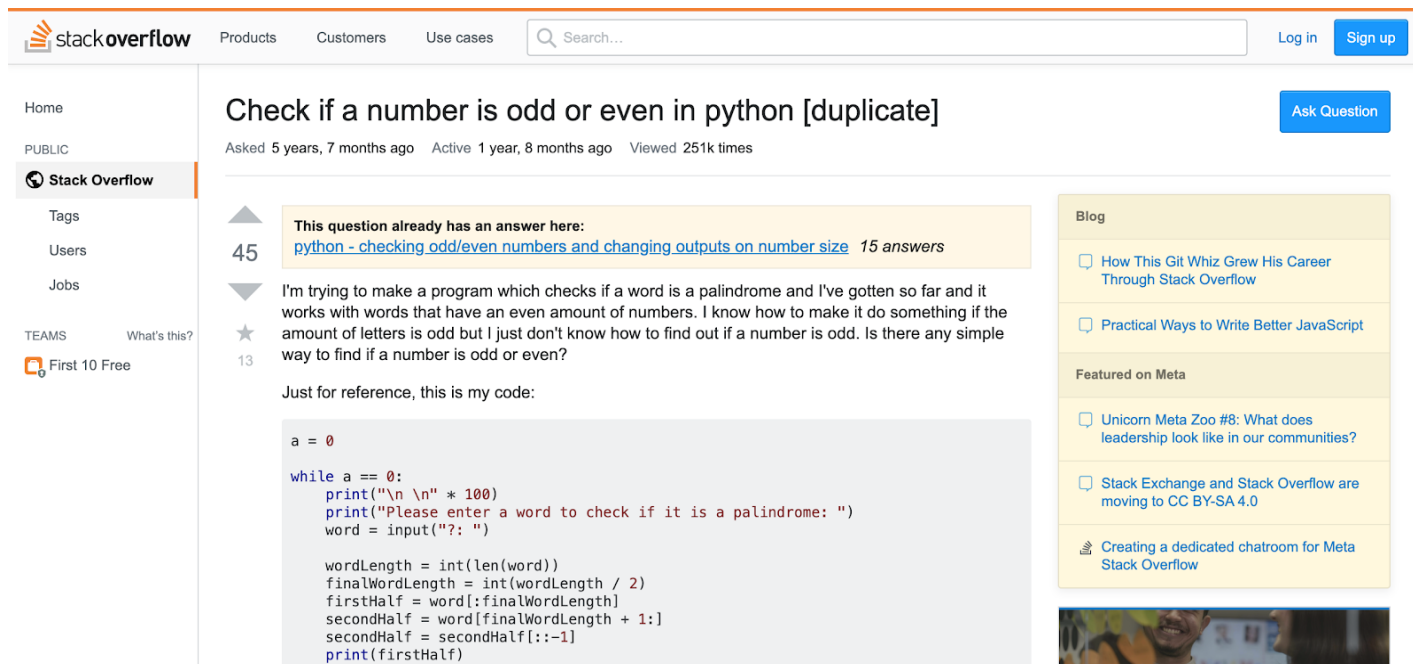
The next resource is a 'w3resource', which is an offshoot of an excellent website called www.w3schools.com. It provides handy and succinct introductions to topics in Python, data analysis, and programming, as well as examples to illustrate the points.

The [final resource](#), though, is one whose usefulness and structure we'll now go into in more depth.

[StackOverflow](#)

The website is **StackOverflow**: this is a kind of forum for programmers, software developers, data analysts, and data scientists in which questions can be asked and answered by the participating community. Answers are peer-reviewed and each user is assigned a rating determined by the value of their contributions to the community. There is not a programmer or data scientist in the world who won't tell you how useful StackOverflow is, so we'll spend a little time looking at how we can get the most from it.

The page initially will look very foreign:



The screenshot shows the StackOverflow website interface. At the top, there's a navigation bar with links for Products, Customers, Use cases, a search bar, and Log in / Sign up buttons. The left sidebar contains navigation links for Home, PUBLIC, Stack Overflow, Tags, Users, Jobs, TEAMS, and What's this? Below these are links for First 10 Free. The main content area displays a question titled "Check if a number is odd or even in python [duplicate]". The question is marked as "duplicate" and has 45 votes. It includes a code snippet for checking a palindrome in Python. The right sidebar features a Blog section with links to "How This Git Whiz Grew His Career Through Stack Overflow", "Practical Ways to Write Better JavaScript", and "Creating a dedicated chatroom for Meta Stack Overflow".

We can see the question that someone has asked in bold, which is almost exactly what we want to know. We can see that 45 other people have thought this a valuable thing to ask. We can see that the question has already been asked, from the 'duplicate' word in square brackets: StackOverflow cleverly tries to flag question-answer threads that already exist in order to reduce the number of superfluous questions. And we can see the problem that our inquirer was trying to solve: checking whether a string is a palindrome in Python (i.e, the same forward as backward, like 'ABBA'). This is significantly more complex than what we're trying to do, so the first thing we do is **scroll down to the answers**.

At the top of the answer section is the top-rated answer. This one has been up-voted at least 123 times:

6 Answers

active

oldest


votes

▲

123

▼


```
if num % 2 == 0:
    pass # Even
else:
    pass # Odd
```



The `%` sign is like division only it checks for the remainder, so if the number divided by `2` has a remainder of `0` it's even otherwise odd.

[share](#) [improve this answer](#)


edited Jan 2 '18 at 22:20



phoenix

2,140 ● 1 ● 22 ● 27


answered Feb 17 '14 at 19:05



DeadChex

3,015 ● 1 ● 21 ● 31

1

is 0(Zero) considered an even in python or does it need a separate (or) statement in the condition? e.g. if num % 2 == 0: or num == 0 – [trustory](#) Feb 15 at 4:32 

3

0 would be considered even. Since 0/2 = 0 Remainder 0, 0%2 is 0 – [DeadChex](#) Feb 16 at 17:32

[add a comment](#)

The responder has also given a short and useful reminder to us of the modulo operator `%`. They've offered a neat and elegant way of checking for evenhood: if the remainder you get when you divide the relevant number by 2 is 0, then it's even; otherwise, it's odd.

So what do we do now? **Almost uniformly, we do this:**

1. We try to understand *why* the solution offered works. This is desirable, but not required. Sometimes, we only really understand later in our journeys. Sometimes, we can only find out why a solution works with further research.
2. We copy and paste the solution offered and put it into our code.
3. At this point, unless we are *extremely* lucky, we will need to change the variable names involved so that the solution works for our code. We're not using a variable called `num`; our variable is called `instruments`.
4. We might even need to change the *type* of the variables at this point.
5. We change parts of the code in the solution so that it matches our purposes.

Here, in the *if* and *else* blocks of the solution, there is the keyword `pass`. This simply asks the interpreter to do nothing. The responder probably put this there so that their

solution was as general as possible. We don't want to do nothing, so we'll modify the responder's code into something like:

```
if (num % 2 == 0):  
    print("All is as expected. The number of instruments we have is even.")  
else:  
    print("We have an odd number of instruments. Something's wrong.")
```

This has been a brief introduction to the art of the Google search, but it could also be called the art of programming.