(https://careerkarma.com/)

CAREER KARMA. (https://careerkarma.com/)

Python indentationerror: unindent does not match any outer indentation level Solution

BY 👤 JAMES GALLAGHER (HTTPS://CAREERKARMA.COM/BLOG/AUTHOR/JAMESGALLAGHER/)

UPDATED DECEMBER 1, 2023

Indenting your code can be a messy business if you try to use both spaces and tabs. In Python, using both methods of indentation results in an error. This error is "indentationerror: unindent does not match any outer indentation level".

In this guide, we talk about what this error means and when it is raised. We walk through an example of this error in action to help you learn how to fix it.
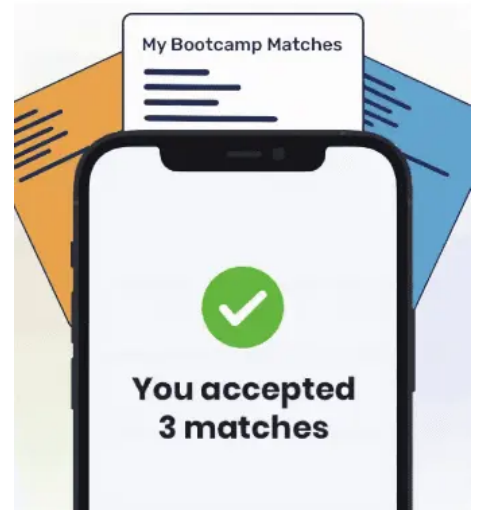
(http://www.facebook.com/sharer.php?
(https://twitter.com/intent/tweet?

Find your bootcamp match

text=Python+indentationerror%3A+unindent+does+not+match+any+outer+indentation+level+Solution&url=https%3A%2F%2Fcareerkarma.com%2Fblog%2Fpython-

Select Your Interest	Your Experience	Time To Start	
Artificial Intelligence	Beginner	Now	GET MATCHED

indentationerror%2F&via=career_karma)

By completing and submitting this form, you agree that Career Karma Platform, LLC may deliver or cause to be delivered information, advertisements, and telemarketing messages regarding their services by email, call, text, recording, and message using a telephone system, dialer, automated technology or system, artificial or prerecorded voice or message device to your email and/or telephone number(s) (and not any other person's email or telephone number) that you entered. Consent is not a condition of receiving information, receiving Career Karma services, or using the website, and you may obtain information by emailing info@careerkarma.com (mailto:info@careerkarma.com). Message & Data rates may apply. Message frequency may vary. Text STOP to unsubscribe. Terms of Service (https://careerkarma.com/terms-of-use) and Privacy Policy (https://privacy.careerkarma.com/privacy-policy) govern the processing and handling of your data.

indentationerror: unindent does not match any outer indentation level

Python code (https://careerkarma.com/blog/how-to-code-in-python/) can be indented with tabs or spaces. It's up to you.

Python only has an objection when you use both spaces and tabs to indent your code. Python requires you use only one method of indentation. This is because the language is statically typed. Statically typed programming languages are sticklers when it comes to syntax.

Indentation errors are a difficult one to understand because it involves the invisible: mixing spaces and tabs. Depending on the code editor you are using, you may not even be able to see whether spaces or tabs have been used until you delete your indents from your code.

IndentationErrors are common when you copy code snippets from the internet. Every developer has their own preference when it comes to indentation and you'll often find code snippets do not adhere to your own preferences. Some snippets will indent with spaces.

If a code snippet you copy into your program uses a different type of indentation, you may see an IndentationError in your code. This is because you mix tabs and spaces.

An Example Scenario

Write a program that finds the factors of a number.

Start by defining a function (https://careerkarma.com/blog/python-functions/) to calculate the factors of a number:

```
def get_factors(number):
    for i in range(1, number + 1):
        if number % i == 0:
            print("{} is a factor of {}.".format(i, number))
```

This function uses a for loop (https://careerkarma.com/blog/python-for-loop/) to iterate through every number in the range of 1 and the number we have specified plus 1.

In each iteration, our program checks if there is a remainder after dividing "number" by "i". We do this using the modulo operator (https://careerkarma.com/blog/python-modulo/). If there is not a remainder, "i" is a factor of "number".

If a number is a factor of "number", a message is printed to the console.

We to call our function:

```
get_factors(32)
```

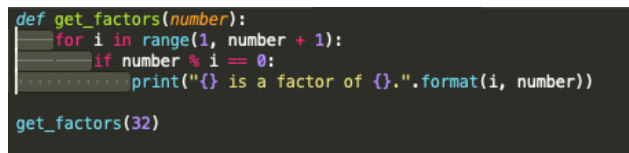This code calculates all the factors of 32. Run our code and see what happens:

```
File "test.py", line 4
    print("{} is a factor of {}.".format(i, number))
                    ^
IndentationError: unindent does not match any outer indentation level
```

Our code returns an IndentationError.

The Solution

We have made a mistake with the styling of our code. We know this because IndentationErrors are raised when there is a problem with how your code is indented.

Take a look at how our code is styled. In Sublime Text, we can see the styles of our code by hovering over each line:

Each line represents a tab. Each dot represents a space. You can see that we have mixed up both spaces and tabs in our code snippet. Python likes consistency in indents and so the interpreter returns an error when we run our code.

If you use a text editor that does not support this behavior, check whether your code uses spaces or tabs by backspacing your indentation. If your code removes a tab when you press the backspace key, that part of your code is using tabs.

Spaces are the preferred method of indentation in Python (https://www.python.org/dev/peps/pep-0008/) but you can use tabs if you want.

Let's revise our code:

```
def get_factors(number):
        for i in range(1, number + 1):
                if number % i == 0:
                        print("{} is a factor of {}.".format(i, number))
```

We have replaced all the spaces with tabs. Run our program again:

```
1 is a factor of 32.
2 is a factor of 32.
4 is a factor of 32.
8 is a factor of 32.
16 is a factor of 32.
32 is a factor of 32.
```

Our code successfully returns a list of all the factors of 32. This shows us that our code was logically accurate all along. It was our indentation that caused the problem.

Conclusion

The "indentationerror: unindent does not match any outer indentation level" error is raised when you use both spaces and tabs to indent your code.

To solve this error, check to make sure that all your code uses either spaces or tabs in a program. Now you're ready to resolve this common Python error like a professional software developer (https://medium.com/careerkarma/python-vs-java-web-development-career-karma-6e098c9744a4)!

About us: Career Karma is a platform designed to help job seekers find, research, and connect with job training programs to advance their careers. Learn about the CK publication (https://careerkarma.com/blog/about/).

What's Next?

WANT TO TAKE ACTION?
Get matched with top bootcamps

>

WANT TO DIVE DEEPER?
Ask a question to our community

>

(https://careerkarma.com/?utm_campaign=ck-blog&utm_source=whatsnext)

WANT TO EXPLORE TECH CAREERS?
Take our careers quiz

>

(https://careerkarma.com/quiz?utm_campaign=ck-blog&utm_source=whatsnext)

About the Author

James Gallagher (https://careerkarma.com/blog/author/jamesgallagher/)
Technical Content Manager at **Career Karma (https://careerkarma.com/)**

James Gallagher is a self-taught programmer and the technical content manager at Career Karma. He has experience in range of programming languages and extensive expertise in Python, HTML, CSS, and JavaScript. James has written hundreds of programming tuto... read more about the author (https://careerkarma.com/blog/author/jamesgallagher/)

COMMENTS (1) ⌄

Jobs (https://careerkarma.com/company/jobs/)

Mission (https://careerkarma.com/company/values/)

Resource Center (https://careerkarma.com/blog/)

Press (https://careerkarma.com/company/press/)

Terms & Conditions (https://careerkarma.com/terms-of-use/)

Sitemap (https://careerkarma.com/blog/html-sitemap/)