



# Here's Your Mistake...

## Taking a closer look at students' mistakes

Dr. Tobias Kohn

# What is a “syntax error”?

# Where do syntax errors come from?

# What can we do about syntax errors?

# I. UNDERSTANDING THE PROBLEM

# Counting Letters

Write a program that counts how often the letter e occurs in a string.

## Counting Letters

Write a program that counts how often the letter e occurs in a string.

```
count = 0
for c in my_text:
    if c == 'e':
        count += 1

print(count)
```

## Counting Letters

Write a program that counts how often the letter e occurs in a string.

```
count = 0
for c in my_text:
    if c = 'e':
        count += 1

print(count)
```

**SyntaxError: invalid syntax**



## Counting Letters

Write a program that counts how often the letter e occurs in a string.

```
count = 0
for 'e' in my_text:
    count += 1

print(count)
```

## Counting Letters

Write a program that counts how often the letter e occurs in a string.

```
count = 0
for 'e' in my_text:
    count += 1

print(count)
```

**SyntaxError: can't assign to literal**

# Counting Letters

## “Minor” mistake

```
count = 0
for c in my_text:
    if c == 'e':
        count += 1

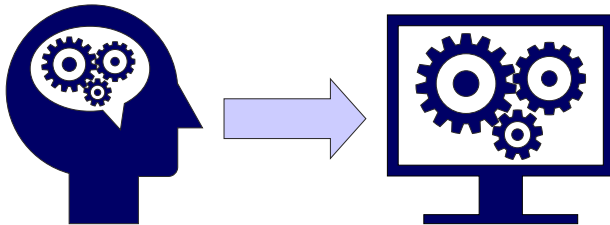
print(count)
```

## Misconception

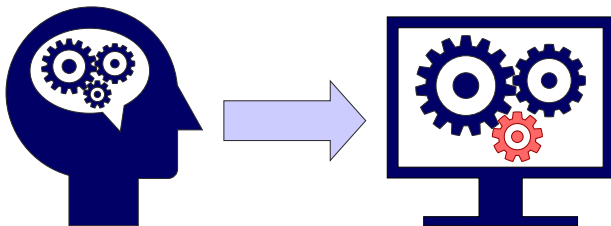
```
count = 0
for 'e' in my_text:
    count += 1

print(count)
```

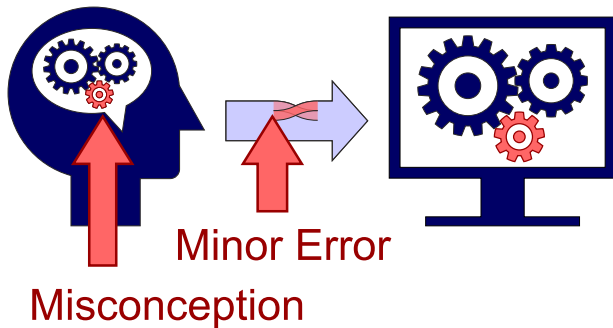
## Minor mistakes and misconceptions



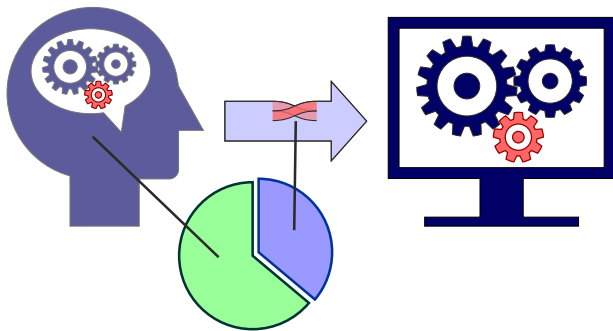
## Minor mistakes and misconceptions



## Minor mistakes and misconceptions



## Minor mistakes and misconceptions



# Some syntax errors are invisible



## Hidden syntax errors

```
count = 0
for c in my_text:
    if c == 'e':
        count =+ 1

print(count)
```

## Hidden syntax errors

```
count = 0
for c in my_text:
    if c == 'e':
        count =+ 1

print(count)
```

## Hidden syntax errors

```
def smallest(x, y, z):  
    if x and y > z:  
        return z  
    if x and z > y:  
        return y  
    if y and z > x:  
        return x
```

## Hidden syntax errors

```
def smallest(x, y, z):  
    if x and y > z:  
        return z  
    if x and z > y:  
        return y  
    if y and z > x:  
        return x
```

## Hidden syntax errors

```
while True:
    key = getKey()
    if key == LEFT:
        left(90)
    elif key == RIGHT:
        right(90)
else:
    forward(1)
```

## Hidden syntax errors

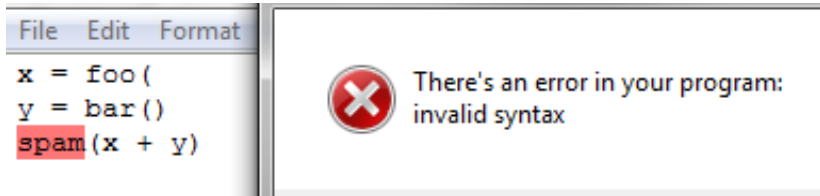
```
while True:
    key = getKey()
    if key == LEFT:
        left(90)
    elif key == RIGHT:
        right(90)
else:
    forward(1)
```

## II. FINDING SOLUTIONS

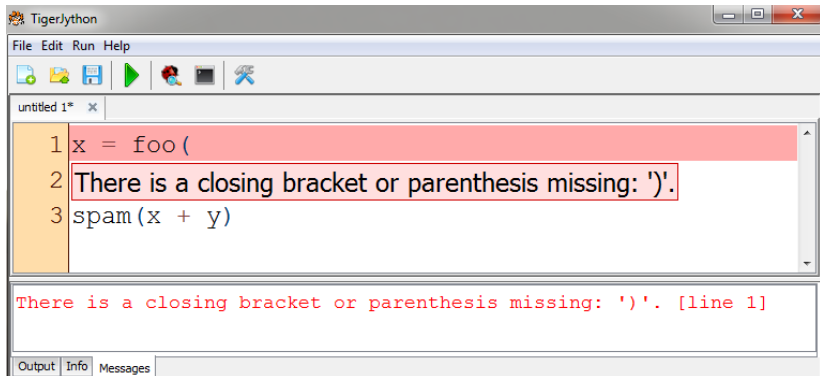
# Better error messages



## Better error messages



# Better error messages



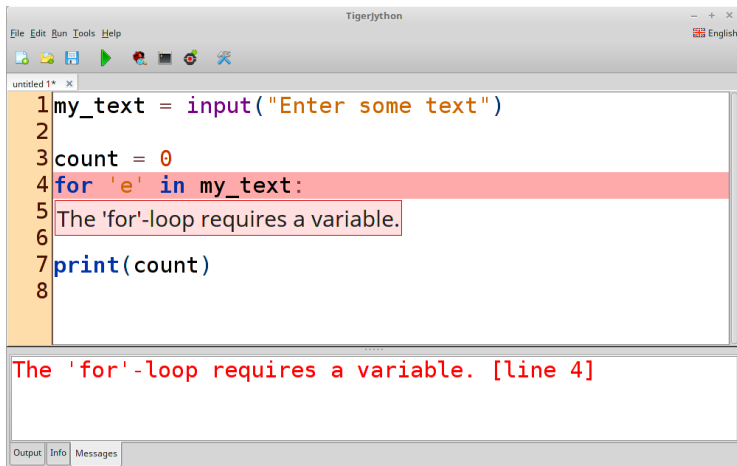
The screenshot shows the Tigerlython IDE interface. The main editor window displays the following code:

```
1 x = foo (  
2  
3 spam(x + y)
```

A red box highlights the error message: "There is a closing bracket or parenthesis missing: ')'. [line 1]".

The output window at the bottom shows the error message: "There is a closing bracket or parenthesis missing: ')'. [line 1]".

# Better error messages

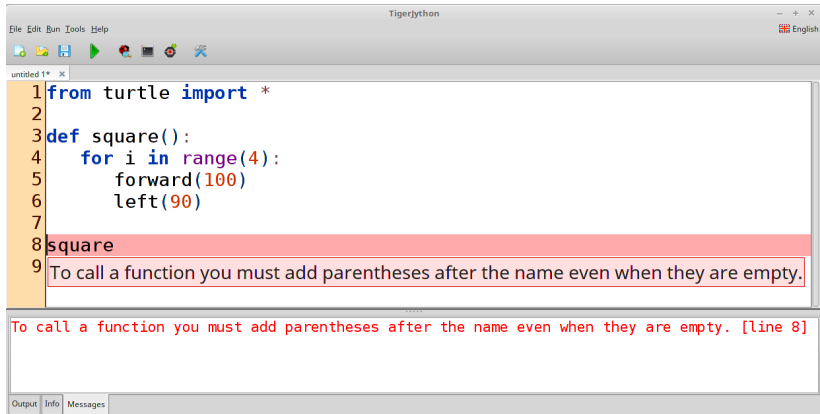


The screenshot shows the TigerJython IDE interface. The main editor window contains the following Python code:

```
1 my_text = input("Enter some text")
2
3 count = 0
4 for 'e' in my_text:
5     The 'for'-loop requires a variable.
6
7 print(count)
8
```

The line `4 for 'e' in my_text:` is highlighted in red. A tooltip-like box points to this line with the text: "The 'for'-loop requires a variable." Below the editor, a separate window displays the error message in red text: "The 'for'-loop requires a variable. [line 4]". At the bottom of the IDE, there are tabs for "Output", "Info", and "Messages".

# Better error messages



The screenshot shows a Python IDE window titled "TigerJython". The code editor contains the following Python code:

```
1 from turtle import *
2
3 def square():
4     for i in range(4):
5         forward(100)
6         left(90)
7
8 square
9
```

Line 8 is highlighted in red, and a tooltip message is displayed over it: "To call a function you must add parentheses after the name even when they are empty." Below the code editor, a larger error message box displays the same text in red: "To call a function you must add parentheses after the name even when they are empty. [line 8]". The IDE interface includes a menu bar (File, Edit, Run, Tools, Help), a toolbar with icons for file operations and execution, and a status bar at the bottom with tabs for Output, Info, and Messages.

**Does it work? – Yes, but...**

# Students' understanding is brittle

# Students' understanding is brittle

## **NameError: name 'S' is not defined**

```
def hexagon(s):  
    for i in range(6):  
        forward(S)  
        right(60)  
  
hexagon(100)
```

# Students' understanding is brittle

## NameError: name 'S' is not defined

```
def hexagon(s):  
    for i in range(6):  
        forward(S)  
        right(60)
```

```
hexagon(100)
```

```
def hexagon(s):  
    for i in range(6):  
        forward(100)  
        right(60)
```

```
hexagon(100)
```



## Students' understanding is brittle

**IndentationError: expected an indented block**

```
for i in range(4):  
forward(100)  
right(90)
```

## Students' understanding is brittle

### IndentationError: expected an indented block

```
for i in range(4):  
forward(100)  
right(90)
```

```
forward(100)  
right(90)  
forward(100)  
right(90)  
forward(100)  
right(90)  
forward(100)  
right(90)
```

# Students' understanding is brittle

**You need parentheses to call a function**

```
def square():  
    for i in range(4):  
        forward(100)  
        left(90)
```

square

# Students' understanding is brittle

## You need parentheses to call a function

```
def square():  
    for i in range(4):  
        forward(100)  
        left(90)
```

square

```
def square():  
    for i in range(4):  
        forward(100)  
        left(90)
```

(square)

# Addressing misconceptions

## Variables and Assignment

Test whether  $x$  is positive before computing the square root of  $x$ .

## Variables and Assignment

Test whether  $x$  is positive before computing the square root of  $x$ .

```
x = input("Enter a number:")
y = sqrt(x)
if x >= 0:
    print(y)
else:
    print("No result exists")
```

# Variables and Assignment

*What image does the turtle draw?*

```
s = 1
t = 3 * s + 1
for i in range(4):
    forward( t )
    left( 90 )
    s += 2
```



# Variables and Assignment

- Students use *mathematical* reasoning:  
 $y = \text{sqrt}(x)$  establishes a relationship between  $x$  and  $y$ .
- *Lazy* evaluation:  
 $y$  is (re)computed from  $x$  when  $y$  is *used/required*.
- Even variables and assignment can be difficult!

# Variables and Assignment

- Students use *mathematical* reasoning:  
 $y = \text{sqrt}(x)$  establishes a relationship between  $x$  and  $y$ .
- *Lazy* evaluation:  
 $y$  is (re)computed from  $x$  when  $y$  is *used/required*.
- Even variables and assignment can be difficult!
- **Make it explicit and discuss it in teaching!**

# Wrapping up...

## Wrapping up. . .

- **What is a “syntax error”?**
- **Where do syntax errors come from?**
- **What can we do about syntax errors?**

# Thank You