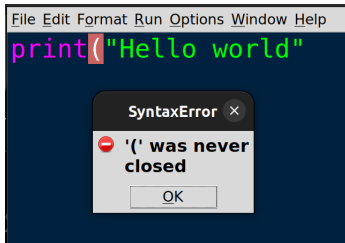


Programmation en Python

Anne Garcia-Sanchez

M2i cyber dev - CFA CCI Avignon
16 octobre 2023

SyntaxError: erreurs de syntaxe



- erreurs détectées par l'analyseur syntaxique
- doivent être corrigées par le programmeur

Exceptions

- erreurs détectées durant l'exécution
- fournissent de l'information sur l'erreur qui se produit
- peuvent être utilisées dans un programme

<https://docs.python.org/fr/3/tutorial/errors.html>

Exception: ValueError

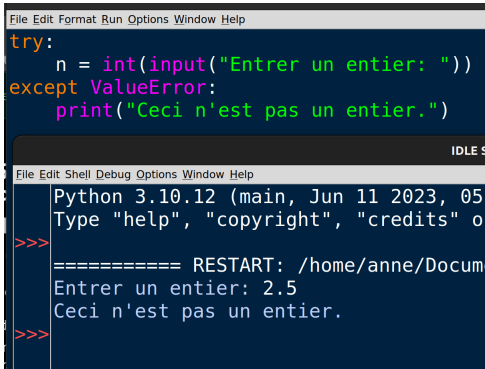
```
getinteger.py - /home/anne/Documents/PYTHON/DEMO/getinteger.py (3.10.12)
File Edit Format Run Options Window Help

n = int(input("Entrer un entier: "))
print(f"valeur de n: {n}")

IDLE Shell 3.10.12
File Edit Shell Debug Options Window Help

Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/anne/Documents/PYTHON/DEMO/getinteger.py =====
Entrer un entier: 2.5
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py", line 578, in runcode
    exec(code, self.locals)
  File "/home/anne/Documents/PYTHON/DEMO/getinteger.py", line 1, in <module>
    n = int(input("Entrer un entier: "))
ValueError: invalid literal for int() with base 10: '2.5'
>>>
```

Exception: ValueError



The screenshot shows a Python IDLE shell window. The top menu bar includes 'File', 'Edit', 'Format', 'Run', 'Options', 'Window', and 'Help'. The code being executed is a try-except block that attempts to convert user input to an integer. The user has entered '2.5', which causes a 'ValueError' exception. The shell displays the exception message 'ValueError: invalid literal for int() with base 10: '2.5'' and the corresponding error traceback. The prompt '>>>' is shown at the end of the output.

```
File Edit Format Run Options Window Help
try:
    n = int(input("Entrer un entier: "))
except ValueError:
    print("Ceci n'est pas un entier.")

IDLE Shell
File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023, 05:10:11) [AMD64] on win32
Type "help()", "copyright()", "credits()" or "quit()" for more
>>>
===== RESTART: /home/anne/Documents/Python/Exceptions/
Entrer un entier: 2.5
ValueError: invalid literal for int() with base 10: '2.5'
>>>
```

Exception: NameError

File Edit Format Run Options Window Help

```
try:
    n = int(input("Entrer un entier: "))
except ValueError:
    print("Ceci n'est pas un entier.")
print(f"valeur de n: {n}")
```

IDLE Shell 3.10.12

File Edit Shell Debug Options Window Help

Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.

>>>

```
===== RESTART: /home/anne/Documents/PYTHON/DEMO/getinteger2.py =====
Entrer un entier: 2.5
Ceci n'est pas un entier.
```

>>>

```
===== RESTART: /home/anne/Documents/PYTHON/DEMO/getinteger2.py =====
Entrer un entier: 2.5
Ceci n'est pas un entier.
```

Traceback (most recent call last):

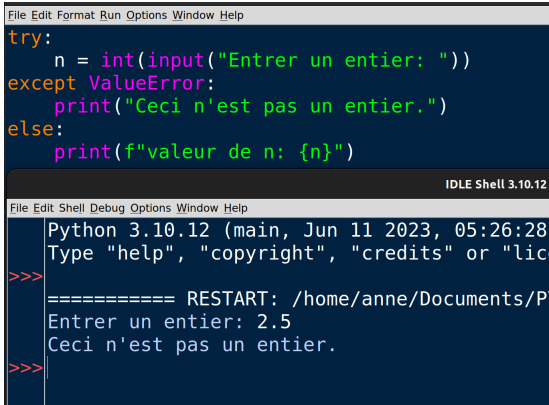
File "/usr/lib/python3.10/idlelib/run.py", line 578, in runcode
exec(code, self.locals)

File "/home/anne/Documents/PYTHON/DEMO/getinteger2.py", line 5, in <module>
print(f"valeur de n: {n}")

NameError: name 'n' is not defined

>>>

Exception: try except else



```
File Edit Format Run Options Window Help
try:
    n = int(input("Entrer un entier: "))
except ValueError:
    print("Ceci n'est pas un entier.")
else:
    print(f"valeur de n: {n}")

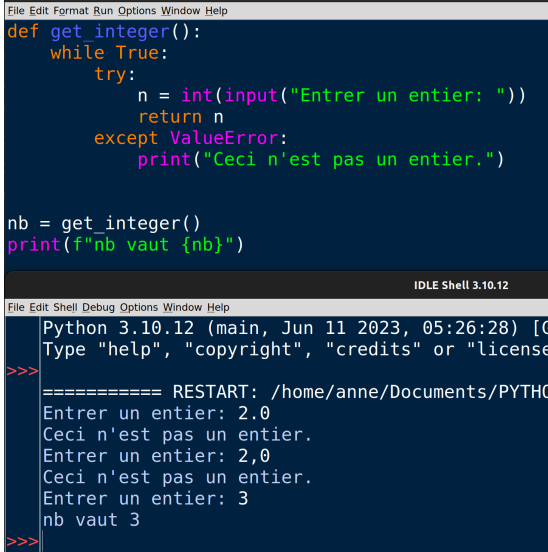
IDLE Shell 3.10.12
File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023, 05:26:28)
Type "help", "copyright", "credits" or "license()"
>>>
===== RESTART: /home/anne/Documents/Python
Entrer un entier: 2.5
Ceci n'est pas un entier.
>>>
```

Exception: contrôler la saisie (avec while True)

```
getinteger5.py - /home/anne/Documents/PYTHON/
File Edit Format Run Options Window Help
while True:
    try:
        n = int(input("Entrer un entier: "))
    except ValueError:
        print("Ceci n'est pas un entier.")
    else:
        print(f"valeur de n: {n}")
        break

IDLE Shell 3.10.1
File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023, 05:26:2
Type "help", "copyright", "credits" or "li
>>>
===== RESTART: /home/anne/Documents/
Entrer un entier: 2.0
Ceci n'est pas un entier.
Entrer un entier: 2,0
Ceci n'est pas un entier.
Entrer un entier: integer(2)
Ceci n'est pas un entier.
Entrer un entier: 5
valeur de n: 5
>>>
```


Exception: contrôler la saisie (avec une fonction)



The image shows a screenshot of a Python IDLE Shell window. The top part of the window displays the source code for a function named `get_integer()`. This function uses a `while True:` loop and a `try:` block to attempt to convert user input to an integer. If a `ValueError` is raised, it prints a message and loops back. Below the function definition, the function is called and its result is printed. The bottom part of the window shows the interactive shell where the program is executed. It shows the prompt `>>>`, the program output, and the user's input. The first two attempts fail with the message "Ceci n'est pas un entier." because of a comma in the input "2,0". The third attempt succeeds with the input "3".

```
File Edit Format Run Options Window Help
def get_integer():
    while True:
        try:
            n = int(input("Entrer un entier: "))
            return n
        except ValueError:
            print("Ceci n'est pas un entier.")

nb = get_integer()
print(f"nb vaut {nb}")

IDLE Shell 3.10.12

File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [C
Type "help", "copyright", "credits" or "license
>>>
===== RESTART: /home/anne/Documents/PYTHO
Entrer un entier: 2.0
Ceci n'est pas un entier.
Entrer un entier: 2,0
Ceci n'est pas un entier.
Entrer un entier: 3
nb vaut 3
>>>
```

Contrôler la saisie sans rien afficher en cas de problème

```
File Edit Format Run Options Window Help
def get_integer():
    while True:
        try:
            return int(input("Entrer un entier: "))
        except ValueError:
            pass

nb = get_integer()
print(f"nb vaut {nb}")

IDLE Shell 3.10.12
File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC]
Type "help", "copyright", "credits" or "license()"
>>>
===== RESTART: /home/anne/Documents/PYTHON
Entrer un entier: 2.0
Entrer un entier: hello
Entrer un entier: 5
nb vaut 5
>>>
```

Contrôler la saisie de façon générale (façon input)

```
def get_integer(question):  
    while True:  
        try:  
            return int(input(question))  
        except ValueError:  
            pass  
  
nb = get_integer("Entrer un entier: ")  
print(f"nb vaut {nb}")
```

Exceptions: ZeroDivisionError

```
>>> def division(a, b) :  
    return a/b  
  
>>> division(3, 2)  
1.5  
>>> division(3, 0)  
Traceback (most recent call last):  
  File "/usr/lib/python3.8/idlelib/run.py", line 55  
    exec(code, self.locals)  
  File "<pyshell#55>", line 1, in <module>  
  File "<pyshell#53>", line 2, in division  
ZeroDivisionError: division by zero  
>>> |
```

Exceptions: ZeroDivisionError

File Edit Format Run Options Window Help

```
def division(a, b):  
    try:  
        return(a / b)  
    except ZeroDivisionError:  
        print("Division par 0")
```

File Edit Shell Debug Options Window Help

```
Python 3.10.12 (main, Jun 11 2023  
Type "help", "copyright", "credit  
>>>==== RESTART: /home/anne  
>>>division(3,2)  
1.5  
>>>division(3,0)  
Division par 0  
>>>
```

Exceptions: TypeError

```
>>> division(3, '2')
Traceback (most recent call last):
  File "/usr/lib/python3.8/idlelib/run.py", line 559, in runcode
    exec(code, self.locals)
  File "<pyshell#59>", line 1, in <module>
  File "<pyshell#56>", line 3, in division
TypeError: unsupported operand type(s) for /: 'int' and 'str'
>>> |
```

```
>>> division(3.5, '2')
Traceback (most recent call last):
  File "/usr/lib/python3.8/idlelib/run.py", line 559, in runcode
    exec(code, self.locals)
  File "<pyshell#60>", line 1, in <module>
  File "<pyshell#56>", line 3, in division
TypeError: unsupported operand type(s) for /: 'float' and 'str'
>>> |
```

Capter plusieurs exceptions

File Edit Format Run Options Window Help

```
def division(a, b):  
    try:  
        return(a / b)  
    except ZeroDivisionError:  
        print("Division par 0")  
    except TypeError:  
        print("les valeurs doivent être de type entier ou flottant")
```

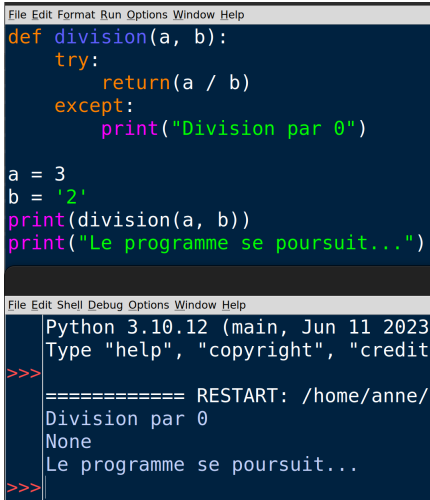
IDLE Shell 3.10.12

File Edit Shell Debug Options Window Help

```
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux  
Type "help", "copyright", "credits" or "license()" for more informa  
>>>  
===== RESTART: /home/anne/Documents/PYTHON/DEMO/division2.py  
>>> division(3, 2)  
1.5  
>>> division(3, '2')  
les valeurs doivent être de type entier ou flottant  
>>> division(3.5, 2)  
1.75  
>>> division(3.5, 2.5)  
1.4  
>>> division(3.5, '2')  
les valeurs doivent être de type entier ou flottant  
>>>
```

Exceptions: mauvaise pratique

On peut ne pas spécifier l'exception: très très fortement déconseillé!
Peut masquer un autre problème...



```
File Edit Format Run Options Window Help
def division(a, b):
    try:
        return(a / b)
    except:
        print("Division par 0")

a = 3
b = '2'
print(division(a, b))
print("Le programme se poursuit...")

File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023)
Type "help", "copyright", "credit"
>>>
===== RESTART: /home/anne/
Division par 0
None
Le programme se poursuit...
>>>
```


try...else...finally

- else: exécuté si aucune exception levée
- finally: exécuté quoi qu'il arrive

try...else...finally

```
while True:
    try:
        n = int(input("Entrer un entier: "))
        print(f"n vaut {n} ")
    except ValueError:
        pass
    else:
        print("On passe ici s'il n'y a pas d'exception ValueError")
        break
    finally:
        print("On passe là dans tous les cas")
```

IDLE Shell 3.10.12

File Edit Shell Debug Options Window Help

Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on lin
Type "help", "copyright", "credits" or "license()" for more info

>>>

===== RESTART: /home/anne/Documents/PYTHON/DEMO/getinteger

Entrer un entier: 2.5

On passe là dans tous les cas

Entrer un entier: hello

On passe là dans tous les cas

Entrer un entier: 6

n vaut 6

On passe ici s'il n'y a pas d'exception ValueError

On passe là dans tous les cas

>>>

déclencher une exception / définir une exception

`raise`