

ALGORITMIA E ESTRUTURAS DE DADOS

EXCEPTIONS HANDLING

LICENCIATURA EM
TECNOLOGIAS E SISTEMAS DE INFORMAÇÃO PARA A WEB
#ESMAD #P.PORTO

Exceptions handling

- ❑ Exceptions
- ❑ try – except
- ❑ Múltiplas exceções try-exception
- ❑ A keyword else
- ❑ A keyword finally
- ❑ Definir exceções: raise



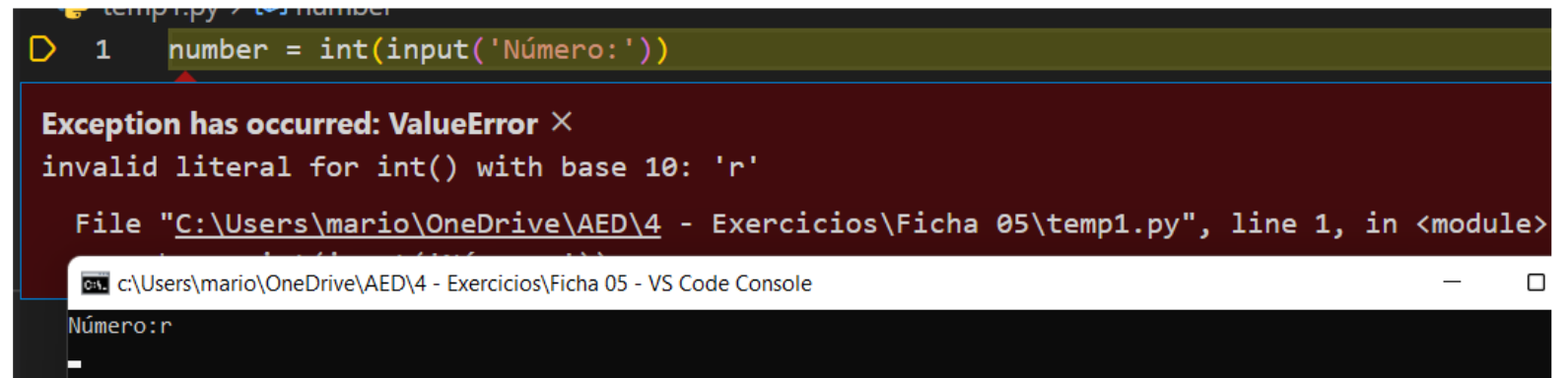
❖ Exceptions

❑ Ao executar o código, podem ocorrer diferentes erros:

❑ Erros de codificação feitos pelo programador (sintaxe ou lógica)

```
1 while True
2     print('Hello world')
```

❑ Erros devido a uma entrada errada de dados



The screenshot shows a code editor with the line `number = int(input('Número:'))`. Below it, a red error message box states: "Exception has occurred: ValueError ×", "invalid literal for int() with base 10: 'r'", and "File 'C:\Users\mario\OneDrive\AED\4 - Exercicios\Ficha 05\temp1.py', line 1, in <module>". At the bottom, the console shows the prompt "Número:r" followed by a carriage return character.

❑ Erros imprevisíveis devido a fatores externos

❑ p.e. um ficheiro removido

❑ P.e. Pasta não existente

❖ Exceptions

- ❑ Expressões ou instruções sintaticamente corretas podem causar um erro
 - ❑ p.e. uma divisão por uma variável sem conteúdo ou com valor de 0
- ❑ Quando ocorre um erro, ou exceção, o Python normalmente para a execução e gera uma mensagem de erro.
- ❑ Essas exceções podem ser tratadas usando a estruturas **try – except**
- ❑ O objetivo é o de capturar os erros de execução, trata-los, e não permitir que o programa crache

❖ try - except

- ❑ Python executa o bloco incluso definido em **try**
- ❑ Quando ocorre um erro durante essa execução, as instruções incluídas em **try** são canceladas, sendo executado o código incluído na keyword **except**



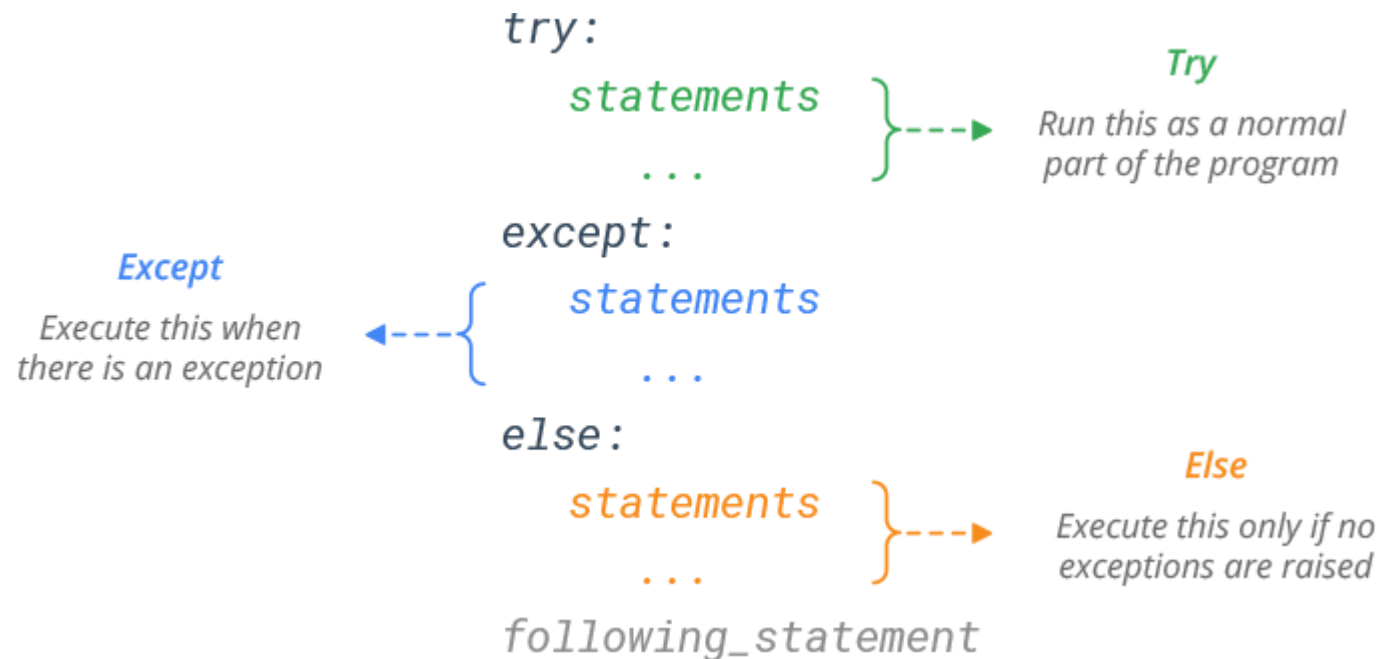
❖ try – except – except ...

- ❑ Podemos definir diversos blocos de exceções para capturar e lidar com erros / exceções específicas, dando uma informação mais objetiva ao utilizador

```
1
2  numero=10
3  try:
4      divisor = int(input('Número:'))
5      divisao= numero/divisor
6  except ZeroDivisionError:
7      print("Divisão por zero não é possível")
8  except:
9      print("Não é possível efetuar a divisão")
10
```

❖ try – except - else

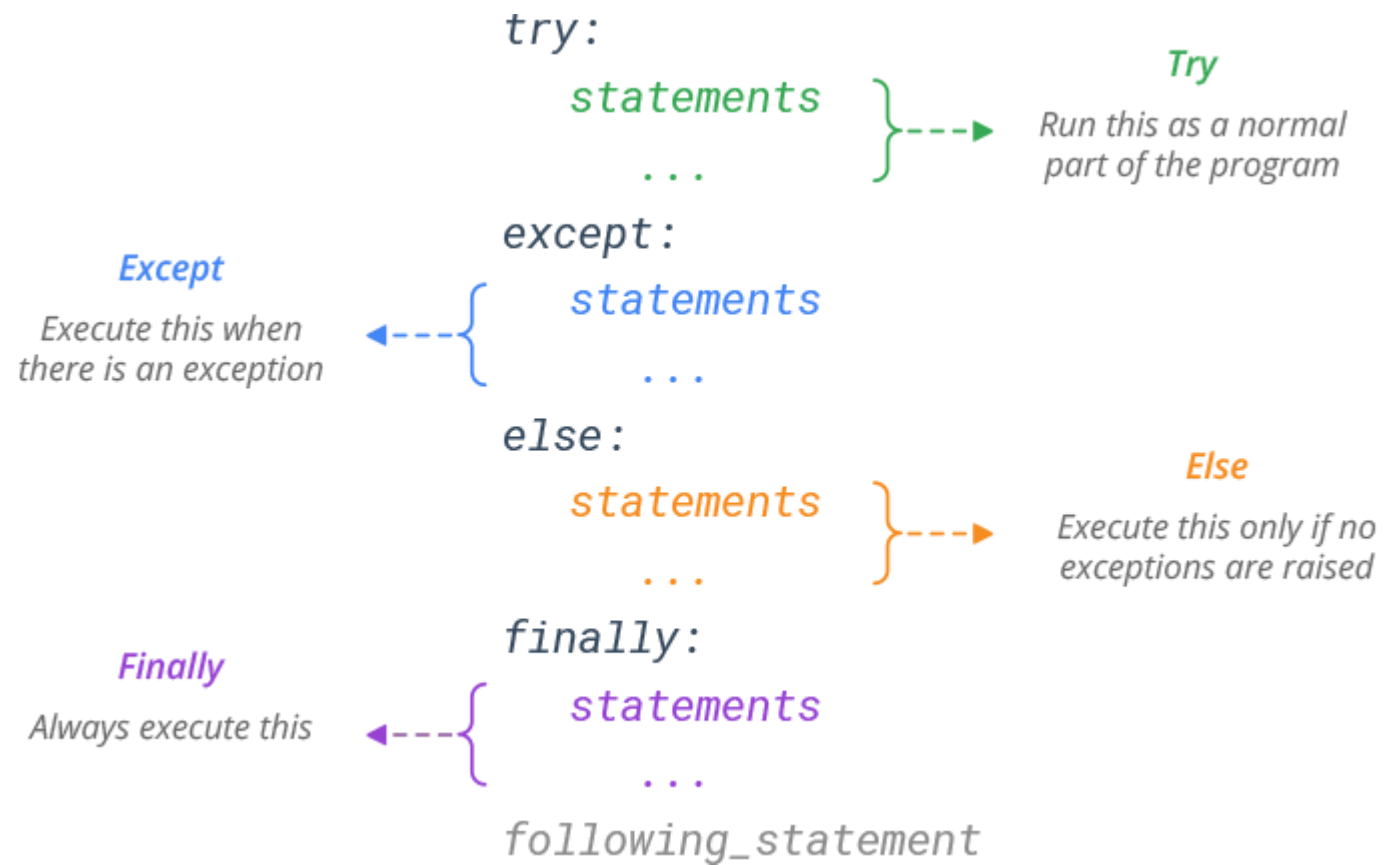
- ❑ A cláusula **else** é opcional
- ❑ A cláusula **else** é executada apenas quando **não ocorre** nenhuma exceção (erro)



❖ try – except – else - finally

❑ A cláusula **finally** também é opcional

❑ Uma cláusula **finally** é **sempre** executada (desde que expressa no código), tenha ocorrido uma exceção ou não



❖ try – except – else - finally

- ❑ Usamos a cláusula **finally** para definir ações que devem ser executadas em todas as circunstâncias, como p.e. fechar um ficheiro

```
# finally clause is always executed
try:
    x = 1/0
except:
    print('Something went wrong')
finally:
    print('Always execute this')
# Prints Something went wrong
# Prints Always execute this
```

```
# Exception handling during file manipulation
f = open('myfile.txt')
try:
    print(f.read())
except:
    print("Something went wrong")
finally:
    f.close()
```

❖ try – except – else - finally

❑ Lista de exceções standard em Python:

https://www.tutorialspoint.com/python/standard_exceptions.htm

<https://docs.python.org/3/library/exceptions.html#concrete-exceptions>

Exceções	Descrição
ValueError	Dados com conteúdo inválido para o tipo de dados
ZeroDivisionError	Tentativa de divisão por zero
TypeError	Operação inválida para o tipo de dados
IOError	Tentativa de abrir um ficheiro que não existe
IndexError	Quando um índice especificado não é válido
ImportError	Quando não é possível importar um determinado módulo
ArithmeticError	Quando a operação aritmética especificada não é possível de concretizar

❖ Exemplos

```
Exceptions.py > ...
1
2
3 try:
4     numero = int(input("Número: "))
5 except ValueError:
6     print("O valor é incorreto")
7 except:
8     print("Ocorreu um erro na inserção de dados")
9
10 print("código a seguir ao tratamento de erro!")
11
12
13 input()
```

C:\WINDOWS\py.exe

Número: 123
código a seguir ao tratamento de erro!

❖ Exemplos

```
try:  
    numero = int(input("Número: "))  
except ValueError:  
    print("O valor é incorreto")  
except:  
    print("Ocorreu um erro na inserção")
```

```
print("código a seguir ao tratamento d
```

```
input()
```

C:\WINDOWS\py.exe

Número: a123
O valor é incorreto
código a seguir ao tratamento de erro!

❖ Exemplos

```
try:
    numero = int(input("Número: "))
    divisor = int(input("Divisor: "))
    quociente = numero / divisor
    print(quociente)
except ValueError:
    print("O número inserido está incorreto")
except ZeroDivisionError:
    print("Não é possível dividir por 0!")
except:
    print("Erro no cálculo!")

print("código a seguir ao try-except")
input()
```

C:\WINDOWS\py.exe

Número: 10
Divisor: 2
5.0
código a seguir ao try-except

❖ Exemplos

Exveptions1.py > ...

```
1  try:
2      numero = int(input("Número: "))
3      divisor = int(input("Divisor: "))
4      quociente = numero / divisor
5      print(quociente)
6  except ValueError:
7      print("O número inserido está incorreto")
8  except ZeroDivisionError:
9      print("Não é possível dividir por 0!")
10 except:
11     print("Erro no cálculo!")
12
13 print("código a seguir ao try-except")
14 input()
15
```

C:\WINDOWS\py.exe

Número: 120
Divisor: 0
Não é possível dividir por 0!
código a seguir ao try-except

❖ Exemplos

```
Exceptions.py > ...
1  try:
2      numero = int(input("Número: "))
3      divisor = int(input("Divisor: "))
4      quociente = numero / divisor
5      print(quociente)
6  except ValueError:
7      print("O número inserido está incorreto")
8  except ZeroDivisionError:
9      print("Não é possível dividir por 0!")
0  except:
1      print("Erro no cálculo!")
2  finally:
3      print("código executado!")
4
5  print("código a seguir ao try-except")
6  input()
7
```

C:\WINDOWS\py.exe

Número: 10
Divisor: 3
3.3333333333333335
código executado!
código a seguir ao try-except

❖ Exemplos

```
try:
    f = open("teste.txt")
    f.write("teste de escrita em ficheiro")
except:
    print("Erro na abertura do ficheiro!")
finally:
    if f.closed == False:
        f.close()
```


❖ Criar Exceções

- ❑ É possível definirmos as nossas próprias exceções, definindo condições para a sua ocorrência
- ❑ Para lançar (definir) uma exceção usamos a keyword **raise**

```
Exp3.py > ...
1
2 try:
3     numero = int(input("indique um Número positivo:"))
4     if numero < 0:
5         raise ValueError()
6 except:
7     print("valor incorreto")
8
9
10 input()
11
```

C:\WINDOWS\py.exe

indique um Número positivo:-1
valor incorreto

❖ Criar Exceções

```
try:  
    numero = int(input("indique um Número entre [0 e 20]: "))  
    if numero < 0 or numero > 20:  
        raise ValueError()  
except:  
    print("valor não está dentro dos limites definidos")
```

input()

C:\WINDOWS\py.exe

```
indique um Número entre [0 e 20]: 24  
valor não está dentro dos limites definidos  
_
```

❖ Criar Exceções

```
try:
    numero = int(input("\n indique um Número entre [0 e 20]: "))
    if numero < 0 or numero > 20:
        raise ValueError()
except ValueError:
    print("O número inserido está incorreto!")
except:
    print("Ocorreu um erro!")
```

input()

C:\WINDOWS\py.exe

indique um Número entre [0 e 20]: a12
O número inserido está incorreto!

❖ Criar Exceções

```
valido=False
while not valido:
    try:
        numero = int(input("\n indique um Número entre [0 e 20]: "))
        if numero < 0 or numero > 20:
            raise ValueError()
    except ValueError:
        print("O número inserido está incorreto!")
    except:
        print("Ocorreu um erro!")
    else:
        valido = True
```

input()

C:\WINDOWS\py.exe

indique um Número entre [0 e 20]: a12
O número inserido está incorreto!

indique um Número entre [0 e 20]: 21
O número inserido está incorreto!

indique um Número entre [0 e 20]: 15

```
1  """
2  Simulador de Peso ideal, considerando o sexo e a altura
3  Peso ideal = (h-100) - (h-150)/k.  k -2 => feminino, k=4 => masculino
4  """
5  valido=False          # Input e validação da variável sexo
6  while not valido:
7      try:
8          sexo = input("\nIndique o sexo (M/F): ")
9          if sexo.lower() != "m" and sexo.lower() != "f":
10             raise ValueError()
11     except ValueError:
12         print("Não inseriu um M ou um F. Por favor volte a tentar!")
13     except:
14         print("Ocorreu um erro. Por favor volte a tentar")
15     else:
16         valido=True
17
18 while True:            # Input e validação da variável altura
19     try:
20         altura = int(input("\nIndique a altura (cm): "))
21     except ValueError:
22         print("Não inseriu um valor inteiro. Por favor volte a tentar!")
23     except:
24         print("Ocorreu um erro. Por favor volte a tentar")
25     else:
26         break
27
28 if sexo.lower() == "m":
29     k = 4
30 else: k = 2
31 pesoIdeal = (altura - 100) - (altura - 150) / k
32 print("O Peso Ideal é {:.2f} Kg" .format(pesoIdeal))
33
```

```
C:\WINDOWS\System32\cmd.  ×  +  v

Indique o sexo (M/F): a
Não inseriu um M ou um F. Por favor volte a tentar!

Indique o sexo (M/F): f

Indique a altura (cm): 1.78
Não inseriu um valor inteiro. Por favor volte a tentar!

Indique a altura (cm): 178
O Peso Ideal é 64.00 Kg
Press any key to continue . . . |
```

❖ Exceptions

Avalia o teu conhecimento

- ☐ Recupere o exercício 9 da Ficha 03 (função **printCharLine**)
- ☐ O nº de caracteres a imprimir em cada linha deve estar embutido numa estrutura **try-exception**.
- ☐ Um valor válido deve variar entre [5-12]
- ☐ Caso não seja um número válido, deve ser pedido ao utilizador a inserção de novo valor, até que se obtenha um valor inteiro entre [5-12]