



Instituto Politécnico da Guarda
Escola Superior de Tecnologia e Gestão

Caderno de Exercícios
Algoritmos e Programação em Python

19 a 24 de outubro

Python: Introdução, instalação e configuração

Problem 1 - Calculate the area of an circle

Proble 2 - Student assesement

Área e perímetros

Proposta de exercícios

Desafio: área de um poligono

Curso: Engenharia Informática

Unidade Curricular:

Algoritmos e Estruturas de Dados

Algorithms and Data Structures

Ano Letivo: 2020/2021

Docente: Paulo Jorge Costa Nunes

Coordenador da área disciplinar: José Carlos Coelho Martins da Fonseca

Conteúdo

1	Introdução	5
2	Instalação e configuração do Python	7
3	Problem 1	11
3.1	Algorithm: Calculate the area of a circle	11
3.2	Python Programs and tests	11
4	Student assessment	13
4.1	Problem 2: Student assessment	13
4.2	Python Programs and tests	13
4.3	Desafio / Challenge	14
5	Perímetro de um retângulo	15
5.0.1	Problema	15
5.1	Desenvolvimento do algoritmo	15
5.1.1	Modelo	15
5.1.2	Esboço	16
5.1.3	Algoritmo	16
5.2	Programa	17
5.2.1	Caso 1	17
6	Volume paralelepípedo	19
6.1	Enunciado do problema	19
6.1.1	Desenvolvimento do algoritmo	19
6.1.1.1	Modelo	19
6.1.1.2	Esboço	20
6.1.1.3	Algoritmo	20
6.1.1.4	Programa	20
6.1.1.5	Caso 1	21
6.2	Validação de dados - Python	21
7	Proposta de exercícios	23
7.0.1	Perímetro de uma rotunda	23
7.0.2	Área de uma rotunda	23
7.0.3	Volume de um cilindro	23
7.0.4	Área, perímetro e centro de massa de polígonos	23

Capítulo 1

Introdução

Com este caderno de exercícios pretende-se contribuir para o ensino da linguagem de programação Python na unidade curricular de Algoritmos e Estrutura de Dados do curso de Engenharia Informática.

Existem muitos recursos na Internet para desenvolvimento em Python, de seguida são apresentados alguns:

1. **Python** — Site oficial to Python <https://www.python.org>.
2. **python-course.eu** — <https://www.python-course.eu/index.php> — Os itens da linguagem Python são apresentados com muitas figuras. São também, apresentadas figuras com exemplos de aplicações reais relacionadas com os items.¹
3. **EduMaven / Python Programming** (879 pages) — <https://edumaven.com/python-programming>
4. **Python 3 Tutorial in PDF - Tutorialspoint** — http://www.tutorialspoint.com/python3/python3_tutorial.pdf.
5. **Python 3.5.2 documentation** — <https://docs.python.org/3.5/>.
6. **The Python Package Index (PyPI)** — O PyPI é um repositório de software para a linguagem de programação Python. Atualmente (2016-09-12) com 88441 módulos desenvolvidos por terceiros <https://pypi.python.org/pypi>.
7. **Python** — <https://www.python.org>.
8. **Python** — <https://www.tutorialspoint.com/python/>.
9. **Book** — <http://www.openbookproject.net/thinkcs/python/english2e/>.
10. **How to Think Like a Computer Scientist: Learning with Python 2nd Edition documentation** — <http://www.openbookproject.net/thinkcs/python/english2e/#>.
11. **Book** — Python Programming: An Introduction to Computer Science by John M. Zelle, Ph.D <http://mcsp.wartburg.edu/zelle/python/>.

- Programas Simples — Diretórios com os programas <http://mcsp.wartburg.edu/zelle/python/ppics3/code/>.

¹Novo em 2018/2019

- Código fonte dos programas para download — <http://mcsp.wartburg.edu/zelle/python/ppics3/code.zip>.
 - **Slides** — Slides Powerpoint para as aulas <http://mcsp.wartburg.edu/zelle/python/ppics3/slides/>.
12. **Book** - Practical Programming (2nd edition) An Introduction to Computer Science Using Python 3 by Paul Gries, Jennifer Campbell, Jason Montojo — <https://pragprog.com/book/gwpy2/practical-programming>
- Programas Simples — Diretórios com os programas <http://mcsp.wartburg.edu/zelle/python/ppics3/code/>.
 - Código fonte dos programas para download — <http://mcsp.wartburg.edu/zelle/python/ppics3/code.zip>.
 - **Slides** — Slides Powerpoint para as aulas <http://mcsp.wartburg.edu/zelle/python/ppics3/slides/>.
13. **Book** — Invent Your Own Computer Games with Python, 2nd Edition 2nd Edition by Al Sweigart (Author) <http://inventwithpython.com/downloads/>.

Capítulo 2

Instalação e configuração do Python

No endereço <http://www.python.org/download/> estão disponíveis diversas versões do Python para diversos sistemas operativos/servidores web. Assim como as respectivas instruções de instalação e configuração do Python.

Os passos para instalar o Python no *Windows* são:

1. Visitar o endereço <http://www.python.org/download/> e descarregar (fazer o download) da versão mais atual do Python 3.
2. Double-click no ficheiro *.msi* descarregado e seguir as instruções. Todas as opções por omissão são adequadas para aprendizagem da linguagem por novos programadores.
3. Testar a instalação — Iniciar o *Python Shell* através do menu *Start* do Windows. Deverá aparecer uma janela semelhante à apresentada na figura 2.1 que tem o nome de *Python Shell*. O pronto (*prompt*) `> > >`, aguarda a escrita, a partir teclado, de operações do utilizador. As linhas `> > > 2 + 4`, `6` e `> > >` representam o resultado da operação de somar os números 2 e 4.

A figura 2.2 mostra a janela do editor IDLE (*Integrated DeveLopment Environment*) do Python semelhante à janela *Shell*. Esta janela apresenta uma barra de menus com diversas funcionalidades para editar, executar e fazer *debug* dos programas.

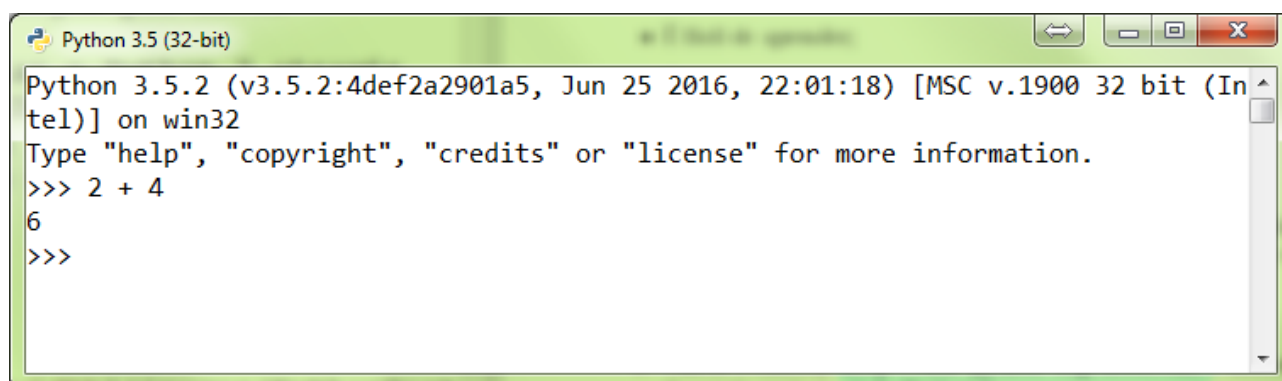


Figura 2.1: Teste de instalação, janela shell interativa do Python.

A listagem 2.1 mostra o código fonte de um programa escrito em Python. A linha 1 iniciada pelo símbolo `#` é um comentário em Python. As linhas de código 2 e 3 escrevem no ecrã e em linhas diferentes o texto contido entre os símbolos `'` (delimitadores de strings). A linha 4 lê a sequência de caracteres (nome) digitados pelo utilizador e armazena-os na variável `myName`. A

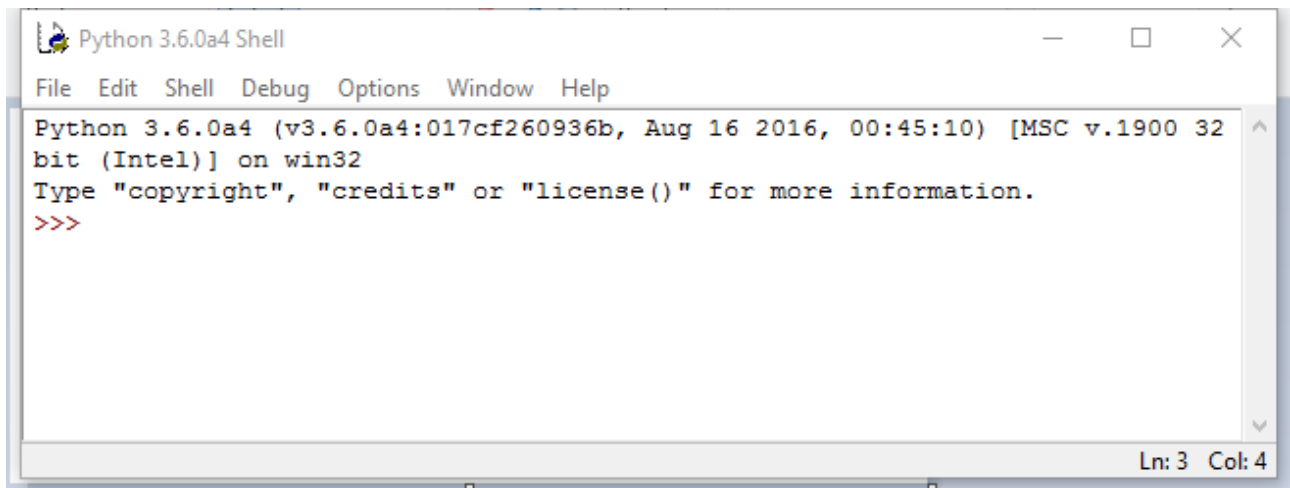


Figura 2.2: Teste de instalação, janela shell interativa do Python.

linha final escreve numa nova linha do ecrã o texto 'It is good to meet you, ', concatenado com o valor da referida variável (Ou seja o nome do utilizador).

As figuras 2.3 e 2.4 ilustram o programa `hello.py` no editor do Python e o output da sua execução com o Ana, respetivamente. A execução de programas pode ser efetuada através da opção *Run Module* do menu *Run*, como ilustra a 2.3.

```

1 # This program says hello and asks for my name.
2 print('Hello world!')
3 print('What is your name?')
4 myName = input()
5 print('It is good to meet you, ' + myName)

```

Listing 2.1: Programa `hello.py`

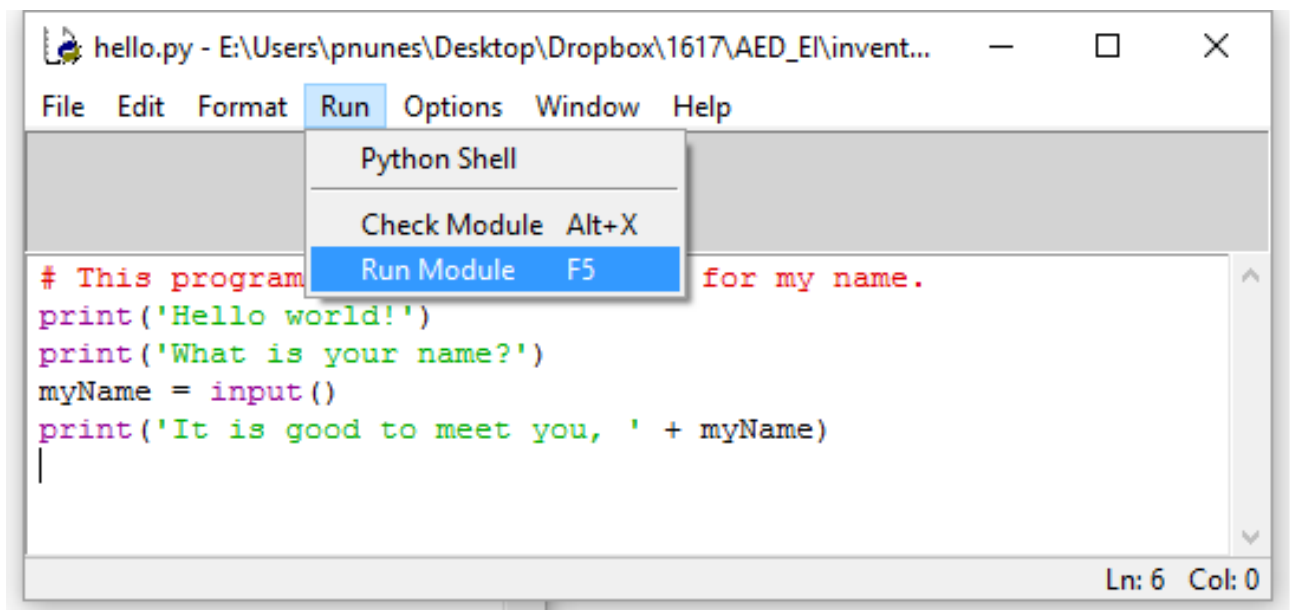
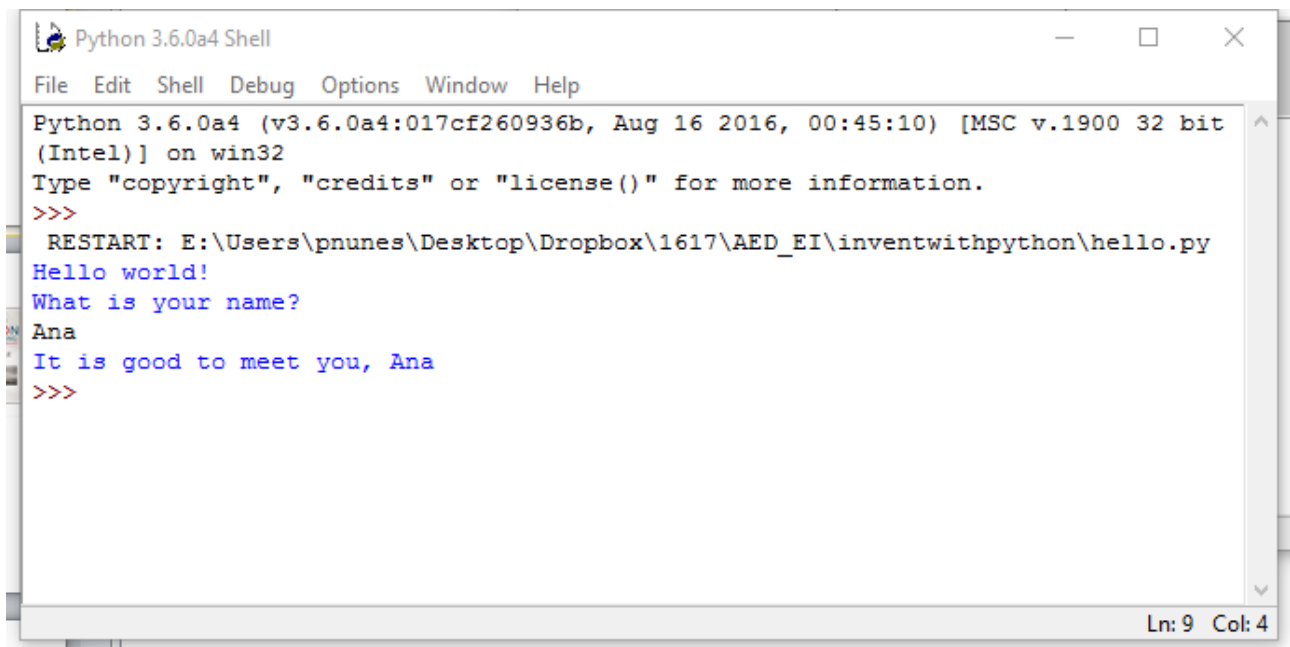


Figura 2.3: Programa `hello.py` no editor Python.



The image shows a screenshot of a Python 3.6.0a4 Shell window. The window has a title bar with the text "Python 3.6.0a4 Shell" and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with the following items: File, Edit, Shell, Debug, Options, Window, and Help. The main area of the window contains the following text:

```
Python 3.6.0a4 (v3.6.0a4:017cf260936b, Aug 16 2016, 00:45:10) [MSC v.1900 32 bit  
(Intel)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>>  
RESTART: E:\Users\pnunes\Desktop\Dropbox\1617\AED_EI\inventwithpython\hello.py  
Hello world!  
What is your name?  
Ana  
It is good to meet you, Ana  
>>>
```

At the bottom right of the window, the status bar shows "Ln: 9 Col: 4".

Figura 2.4: Resultado do programa hello.py para o nome Ana.

Capítulo 3

Problem 1

3.1 Algorithm: Calculate the area of a circle

What is wanted is to calculate the area of a circle, which can be calculated with the following [1].

```
1 Begin
2   Pi = 3.14159;
3   Read Radius;
4   Area = PI x Radius ^ 2;
5   Write Area;
6 End
```

Listing 3.1: Area of a circle

3.2 Python Programs and tests

Note: $Radius^2 = Radius \times Radius$

```
1 Pi = 3.14159
2 Radius = eval(input())
3 Area = Pi * Radius * Radius
4 print (Area)
5
6 >>> python AreaOfCircle.py
7 3
8 28.274309999999996
```

Listing 3.2: Python program: Area of a circle

Mathematical function $pow(x, y)$

This method returns the value of x^y .

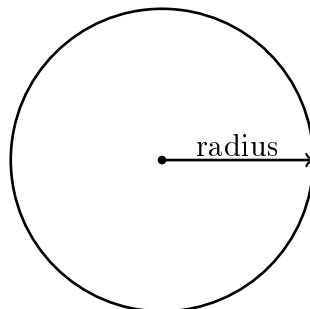


Figura 3.1: Circle

```
1 import math
2 Radius = eval(input())
3 Area = math.pi * pow(Radius, 2)
4 print (Area)
5
6 >>> python AreaOfCircle2.py
7 3
8 28.274309999999996
```

Listing 3.3: Python program: Area of a circle with Python math.pi constant

```
1 import math
2 Radius = eval(input())
3 Area = math.pi * pow(Radius, 2)
4 print ("%0.4f" % Area)
5
6 >>> python AreaOfCircle2.py
7 3
8 28.2743
```

Listing 3.4: Python program: Area of a circle with Python math.pi constant and formatted output

See mathematical functions at: https://www.tutorialspoint.com/python3/python_quick_guide.htm

Capítulo 4

Student assessment

4.1 Problem 2: Student assessment

Check that a student has passed or failed based on the arithmetic mean of the grades of 5 subject. If the mean of the 5 grades is greater than or equal to 10 the student passed, if not the student failed [1].

The arithmetic mean can be calculated by the following equation:

$$Mean = \frac{\sum_{i=1}^5 Grade_i}{5} \quad (4.1)$$

```
1 Begin
2   Read Grade1;
3   Read Grade2;
4   Read Grade3;
5   Read Grade4;
6   Read Grade5;
7   Mean = (Grade1+Grade2+Grade3+Grade4+Grade5)/5;
8   If Mean >= 10 Then
9     Write "The student passed";
10  Else
11    Write "The student failed";
12  End If
13 End
```

Listing 4.1: Algorithm: Student assessment

4.2 Python Programs and tests

Note: $Radius^2 = Radius \times Radius$

```
1 Grade1 = eval(input())
2 Grade2 = eval(input())
3 Grade3 = eval(input())
4 Grade4 = eval(input())
5 Grade5 = eval(input())
6 Mean = (Grade1+Grade2+Grade3+Grade4+Grade5)/5;
7 if Mean >= 10:
8     print("The student passed");
9 else:
10    print("The student failed")
11
12 >>> python StudenAssesement.py
13 10
14 12
15 15
16 16
17 9
18 The student passed
```

Listing 4.2: Python program: Student assessment

4.3 Desafio / Challenge

Como é que resolveria o problema se o estudante tivesse 40 disciplinas?
How would you solve the problem if the student had 40 grades?

Capítulo 5

Perímetro de um retângulo

5.0.1 Problema

Elabore um algoritmo/programa que permita calcular o perímetro de um retângulo.

5.1 Desenvolvimento do algoritmo

5.1.1 Modelo

A figura 5.1 ilustra o modelo para calcular o perímetro. A formula seguinte permite calcular o seu valor:

$$P = 2 \times L + 2 \times C$$

onde:

- **L** - Largura do retângulo ($L \in \mathbb{R}^+$)
- **C** - Comprimento do retângulo ($C \in \mathbb{R}^+$)
- **P** - Perímetro do retângulo ($P \in \mathbb{R}^+$)

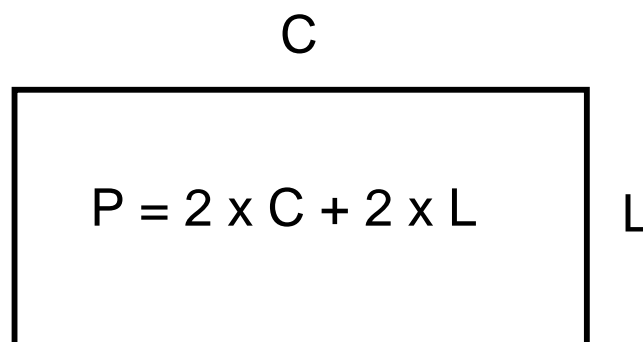


Figura 5.1: Modelo para calcular o perímetro de um retângulo.

5.1.2 Esboço

```

1 LER C
2 LER L
3 P = 2 x C + 2 x L
4 ESCREVER P

```

Listing 5.1: Esboço do algoritmo perímetro de um retângulo.

5.1.3 Algoritmo

```

1 Algoritmo: algoritmoPerimetroRetangulo
2 Objetivo: Permite calcular o perímetro de um retângulo
3 Variáveis
4 Entrada:
5   L (Inteiro T2) - Largura (> 0, <= 99)
6   C (Inteiro T2) - Comprimento (> 0, <= 99)
7 Saída:
8   P (Inteiro T5) - Perímetro (> 0, <= 99999)
9 Data: 2016-9-26 13:46:11
10 Autor: Paulo Nunes
11 Versão: 1.0
12 Obs:
13 Início:
14   /* Entrada de dados (INPUT) */
15   ESCREVER "Largura?"
16   ESCREVER "Comprimento?"
17   /* Processamento (PROCESSING) */
18   P = 2 x L + e x C
19   /* Saída de resultados (OUTPUT) */
20   ESCREVER "Perímetro: ", P
21 Fim.

```

Listing 5.2: Algoritmo perímetro de um retângulo sem validação de dados de entrada.

```

1 Algoritmo: algoritmoPerimetroRetangulo
2 Objetivo: Permite calcular o perímetro de um retângulo
3 Variáveis
4 Entrada:
5   L (Inteiro T2) - Largura (> 0, <= 99)
6   C (Inteiro T2) - Comprimento (> 0, <= 99)
7 Saída:
8   P (Inteiro T5) - Perímetro (> 0, <= 99999)
9 Data: 2016-9-26 13:46:11
10 Autor: Paulo Nunes
11 Versão: 1.0
12 Obs:
13 Início:
14   /* Entrada de dados (INPUT) */
15   FAZER
16     ESCREVER "Largura?"
17     LER L
18   ATÉ ( (L > 0) E (L <= 99) )
19   FAZER
20     ESCREVER "Comprimento?"
21     LER C
22   ATÉ ( (C > 0) E (C <= 99) )
23   /* Processamento (PROCESSING) */
24   P = 2 x L + e x C
25   /* Saída de resultados (OUTPUT) */
26   ESCREVER "Perímetro: ", P
27 Fim.

```

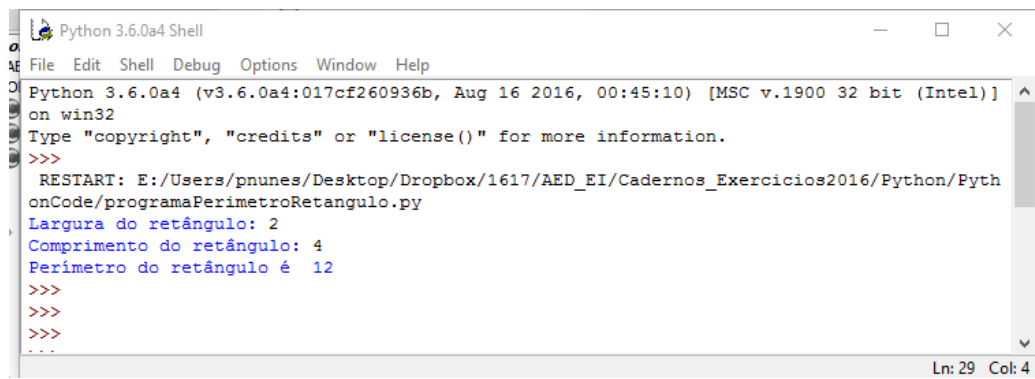
Listing 5.3: Algoritmo perímetro de um retângulo.

5.2 Programa

```
1 def main():
2     L = eval(input("Largura do retângulo: "))
3     C = eval(input("Comprimento do retângulo: "))
4     P = 2 * L + 2 * C
5     print("Perímetro do retângulo ", P)
6
7 main()
```

Listing 5.4: Programa Python perímetro de um retângulo

5.2.1 Caso 1



```
Python 3.6.0a4 Shell
File Edit Shell Debug Options Window Help
Python 3.6.0a4 (v3.6.0a4:017cf260936b, Aug 16 2016, 00:45:10) [MSC v.1900 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: E:/Users/pnunes/Desktop/Dropbox/1617/AED_EI/Cadernos_Exercicios2016/Python/Pyth
onCode/programaPerimetroRetangulo.py
Largura do retângulo: 2
Comprimento do retângulo: 4
Perímetro do retângulo é 12
>>>
>>>
>>>
...
Ln: 29 Col: 4
```

Figura 5.2: Caso 1: perímetro de um retângulo.

Capítulo 6

Volume paralelepípedo

6.1 Enunciado do problema

Elabore um algoritmo/programa que permita calcular o volume de um paralelepípedo, de acordo com a figura seguinte (6.1).

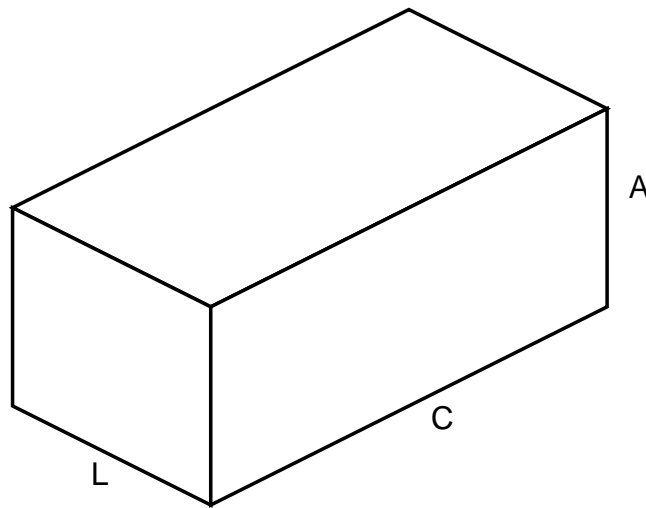


Figura 6.1: Modelo perímetro de um retângulo.

6.1.1 Desenvolvimento do algoritmo

6.1.1.1 Modelo

A figura 6.1 ilustra o modelo para calcular o volume de um paralelepípedo. A formula seguinte permite calcular o seu valor:

$$V = C \times L + \times C \times A$$

onde:

- L - Largura do paralelepípedo ($L \in \mathbb{R}^+$)
- C - Comprimento do paralelepípedo ($C \in \mathbb{R}^+$)
- A - Altura do paralelepípedo ($C \in \mathbb{R}^+$)
- V - Volume do paralelepípedo ($P \in \mathbb{R}^+$)

6.1.1.2 Esboço

```

1  LER C
2  LER L
3  LER A
4  V = C x L x A
5  ESCREVER V

```

Listing 6.1: Esboço do algoritmo o volume de um paralelepípedo.

6.1.1.3 Algoritmo

```

1  Algoritmo: programaVolumeParalelepipedo
2  Objetivo:
3      Permite calcular o volume de um paralelepípedo
4  Variáveis
5  Entrada:
6      L (Real T6.3) - Largura do paralelepípedo (m) (> 0, <= 999.999)
7      C (Real T6.3) - Comprimento do paralelepípedo (m) (> 0, <= 999.999)
8      A (Inteiro T6.3) - Altura do paralelepípedo (m) (> 0, <= 999.999)
9  Saída:
10     V (Real T12.3) - Volume do paralelepípedo (m3) (> 0.0, <= 999999999.999)
11  Data: 2016-9-26 19:4:15
12  Autor: Paulo Nunes
13  Versão: 1.0
14  Obs:
15  Início:
16      /* Entrada de dados (INPUT) */
17      FAZER
18          ESCREVER "Largura do paralelepípedo (m)?"
19          LER L
20      ATÉ ( (L > 0) E (L <= 999.999) )
21      FAZER
22          ESCREVER "Comprimento do paralelepípedo (m)?"
23          LER C
24      ATÉ ( (C > 0) E (C <= 999.999) )
25      FAZER
26          ESCREVER "Altura do paralelepípedo (m)?"
27          LER A
28      ATÉ ( (A > 0) E (A <= 999.999) )
29      /* Processamento (PROCESSING) */
30      V = C * L * A
31      /* Saída de resultados (OUTPUT) */
32      ESCREVER "Volume do paralelepípedo: ", V, " m3"
33  Fim.

```

Listing 6.2: Algoritmo Programa Python volume de um paralelepípedo

6.1.1.4 Programa

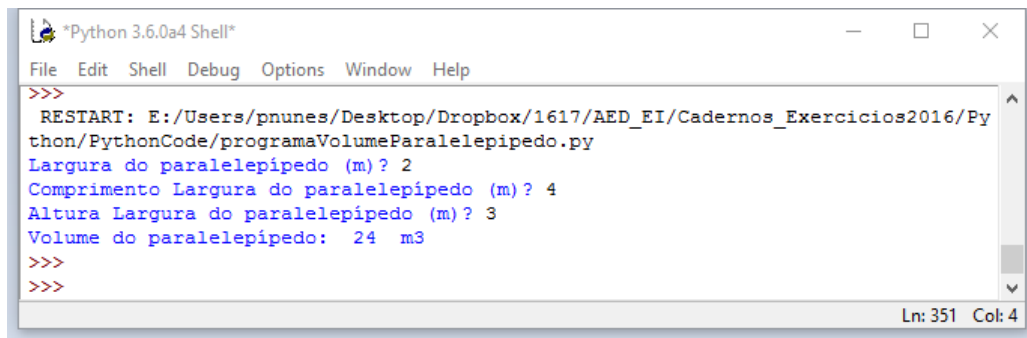
```

1  def programaVolumeParalelepipedo():
2      L = -1
3      while ((L <= 0.0) or (L >= 999.999)):
4          L = eval(input("Largura do paralelepípedo (m)? "))
5      C = -1
6      while ((C <= 0.0) or (C >= 999.999)):
7          C = eval(input("Comprimento Largura do paralelepípedo (m)? "))
8      A = -1
9      while ((A <= 0.0) or (A >= 999.999)):
10         A = eval(input("Altura Largura do paralelepípedo (m)? "))
11     V = C * L * A
12     print("Volume do paralelepípedo: ", V, " m3")
13
14 programaVolumeParalelepipedo()

```

Listing 6.3: Programa Python volume de um paralelepípedo

6.1.1.5 Caso 1



```
*Python 3.6.0a4 Shell*
File Edit Shell Debug Options Window Help
>>>
RESTART: E:/Users/pnunes/Desktop/Dropbox/1617/AED_EI/Cadernos_Exercicios2016/Python/PythonCode/programaVolumeParalelepipedo.py
Largura do paralelepipedo (m)? 2
Comprimento Largura do paralelepipedo (m)? 4
Altura Largura do paralelepipedo (m)? 3
Volume do paralelepipedo: 24 m3
>>>
>>>
```

Figura 6.2: Caso 1: Programa Python volume de um paralelepípedo.

6.2 Validação de dados - Python

```
1 while True:
2     try:
3         age = int(input("Please enter your age: "))
4     except ValueError:
5         print("Sorry, I didn't understand that.")
6         #better try again... Return to the start of the loop
7         continue
8     else:
9         #age was successfully parsed!
10        #we're ready to exit the loop.
11        break
12 if age >= 18:
13     print("You are able to vote in the United States!")
14 else:
15     print("You are not able to vote in the United States.")
```


Capítulo 7

Proposta de exercícios

7.0.1 Perímetro de uma rotunda

Elabore um algoritmo para calcular o perímetro de uma rotunda. Assim como o número de peças de cimento para circundar a rotunda.

7.0.2 Área de uma rotunda

Elabore um algoritmo para calcular a área de uma rotunda. Assim como o número de pés de flor para plantar na rotunda.

7.0.3 Volume de um cilindro

Elabore um algoritmo para calcular a área e o volume de um cilindro.

7.0.4 Área, perímetro e centro de massa de polígonos

Elabore um algoritmo e um programa em `Python` que permita calcular a área, perímetro e centro massa de um polígono 2D.

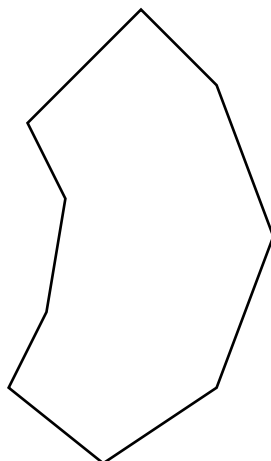


Figura 7.1: Polígono

Bibliografia

- [1] Carlos Carreto. Algoritmos e Estruturas de Dados, Resume of Class N^o1, Engenharia Informática, IPG. Sept 2017.