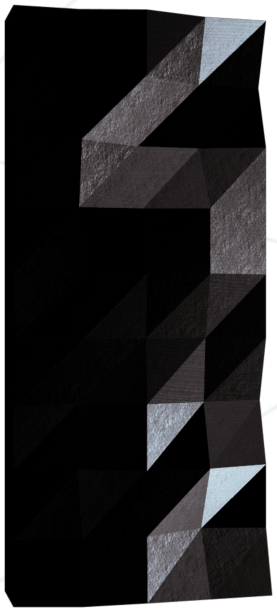

Missão

Módulo 8 - Python

Preâmbulo: Neste módulo, veremos como usar métodos e escopos.

Versão: 2.0

Sumário

Capítulo I

Uma palavra sobre esta Missão

Bem-vindo(a)!
Você começará o primeiro Módulo desta Missão de programação de computadores. Nosso objetivo é apresentar a você o código por trás do software que você usa diariamente e mergulhar no aprendizado entre pares, o modelo educacional da 42.
Programação é sobre lógica, não matemática. Ela fornece blocos de construção básicos que você pode montar de inúmeras maneiras. Não existe uma única solução "correta" para um problema — sua solução será única, assim como as soluções de seus colegas.
Rápido ou lento, elegante ou bagunçado, contanto que funcione, isso é o que importa! Esses blocos de construção formarão uma sequência de instruções (para cálculos, exibições, etc.) que o computador executará na ordem que você projetar.
Em vez de fornecer um curso onde cada problema tem apenas uma solução, nós o(a) colocamos em um ambiente de aprendizado entre pares. Você procurará elementos que possam ajudá-lo(a) a enfrentar seu desafio, refiná-los através de testes e experimentação e, finalmente, criar seu próprio programa. Discuta com os outros, compartilhe suas perspectivas, apresente novas ideias juntos, e teste tudo você mesmo(a) para garantir que funcione.
A avaliação por pares é uma oportunidade fundamental para descobrir abordagens alternativas e identificar possíveis problemas em seu programa que você possa ter perdido (considere o quão frustrante uma falha de programa pode ser). Cada revisor abordará seu trabalho de forma diferente — como clientes com diferentes expectativas — dando a você novas perspectivas. Você pode até formar conexões para futuras colaborações.
Ao final desta Missão, sua jornada será única. Você terá enfrentado desafios diferentes, validado projetos diferentes e escolhido caminhos diferentes dos outros — e isso é perfeitamente normal! Esta é uma experiência coletiva e individual, e todos ganharão algo com ela.
Boa sorte a todos; esperamos que você aproveite esta jornada de descoberta.

Capítulo II

Introdução

O que este módulo mostrará a você:

- Você aprenderá como manipular arrays e suas funções associadas.

Capítulo III

Instruções gerais

A menos que especificado de outra forma, as seguintes regras se aplicam todos os dias desta Piscina.

- Este documento é a única fonte confiável. Não confie em boatos.
- Este documento pode ser atualizado até uma hora antes do prazo de entrega.
- As tarefas devem ser concluídas na ordem especificada. Tarefas posteriores não serão pontuadas, a menos que todas as anteriores sejam concluídas corretamente.
- Preste muita atenção aos direitos de acesso de seus arquivos e pastas.
- Suas tarefas serão avaliadas por seus colegas da Missão.
- Todas as tarefas de shell devem ser executadas usando `/bin/bash`.
- Você não deve deixar nenhum arquivo em seu espaço de trabalho de envio além daqueles explicitamente solicitados pelas tarefas.
- Tem uma pergunta? Pergunte ao seu vizinho à sua esquerda. Caso contrário, tente seu vizinho à sua direita.
- Todas as respostas técnicas de que você precisa podem ser encontradas nas páginas de `man` ou online.
- Lembre-se de usar o servidor do Discord!
- Leia os exemplos completamente, pois eles podem revelar requisitos que não são imediatamente óbvios na atribuição descrição.
- Por Thor, por Odin! Use seu cérebro!!!

Capítulo IV

Exercício 00 Descobrindo Métodos!

	Exercício : 00
Descobrindo Métodos!	
Pasta de entrega : $ex00/$	
Arquivos para entregar : `hello_all.py`	
Funções ou bibliotecas autorizadas : `Todos`	

- Crie um programa chamado `hello_all.py`.
- Certifique-se de que este programa seja executável.
- Este programa deve:
  - Definir um método chamado `hello` que imprime "Hello, everyone!".
  - Chamar este método para exibir a mensagem. Consulte o exemplo abaixo, exceto que a definição do método foi ocultada.

```
?> cat hello_all.py
# Sua definição de método

hello()
?> ./hello_all.py
Hello, everyone!
?>
```

`Procure por "definição de método em Python".`

Capítulo V

Exercício 01: upcase_it

	Exercício : 01
O Retorno de upcase!	
Pasta de entrega : $ex01/$	
Arquivos para entregar : `upcase_it.py`	
Funções ou bibliotecas autorizadas : `Todos`	

- Crie um programa chamado `upcase_it.py` (de novo!)
- Certifique-se de que este programa seja executável.
- Defina um método no programa chamado `upcase_it`.
- O método `upcase_it` deve receber uma string como argumento e retornar a string em maiúsculas.
- Teste o método chamando-o em seu programa. No exemplo abaixo, testamos com "hello":

```
?> cat upcase_it.py
# Sua definição de método

print(upcase_it("hello"))
?> ./upcase_it.py
HELLO
?>
```

Capítulo VI

Exercício 02: downcase_all

	Exercício : 02
Vamos passear no Array	
Pasta de entrega : $ex02/$	
Arquivos para entregar : `downcase_all.py`	
Funções ou bibliotecas autorizadas : `Todos`	

- Crie um programa chamado `downcase_all.py`.
- Certifique-se de que este programa seja executável.
- Defina um método neste programa chamado `downcase_it`.
- O método `downcase_it` deve receber uma string como argumento e retornar a string em minúsculas.
- Aplique este método a cada parâmetro do programa e exiba o valor de retorno para cada um.
- Se não houver parâmetros, exiba "none"seguido por uma quebra de linha.

```
?> ./downcase_all.py
none
?> ./downcase_all.py "HELLO WORLD" "I understood Arrays well!"
hello world
i understood arrays well!
?>
```

Capítulo VII

Exercício 03: greetings_for_all

	Exercício : 03
Vamos dizer olá a todos!	
Pasta de entrega : $ex03/$	
Arquivos para entregar : `greetings_for_all.py`	
Funções ou bibliotecas autorizadas : `Todos`	

- Crie um programa chamado `greetings_for_all.py` que não receba nenhum parâmetro.
- Certifique-se de que este programa seja executável.
- Crie um método chamado `greetings` que receba um nome como parâmetro e exiba uma mensagem de boas-vindas com esse nome.
- Se o método for chamado sem um argumento, seu parâmetro padrão deve ser "noble stranger".
- Se o método for chamado com um argumento que não seja uma string, um erro a mensagem deve ser exibida em vez da mensagem de boas-vindas.

```
?> cat greetings_for_all.py | cat -e
# sua definição de método aqui

greetings('Alexandra')
greetings('Wil')
greetings()
greetings(42)
```

produzirá a seguinte saída:

```
?> ./greetings_for_all.py | cat -e
Hello, Alexandra.$
Hello, Wil.$
Hello, noble stranger.$
Error! It was not a name.$
?>
```

`Parâmetro padrão do Google, método isinstance.`

Capítulo VIII

Exercício 04: methods_everywhere

	Exercício : 04
Métodos em todos os lugares!	
Pasta de entrega : $ex04/$	
Arquivos para entregar : `methods_everywhere.py`	
Funções ou bibliotecas autorizadas : `Todos`	

- Crie um programa chamado`methods_everywhere.py` que recebe parâmetros.
- Certifique-se de que este programa seja executável.
- Você precisa criar dois métodos diferentes neste programa:
  - O método `shrink`: Deve receber uma string como parâmetro e exibir os primeiros oito caracteres dessa string.
  - O método `enlarge`: Deve receber uma string como parâmetro e anexar caracteres 'Z' até que a string tenha um total de oito caracteres. Então, ele exibe a string resultante.
- Para cada argumento passado para o programa:
  - Se o argumento tiver mais de oito caracteres, chame o método `shrink` nele.
  - Se o argumento tiver menos de oito caracteres, chame o método `enlarge` nele.
  - Se o argumento tiver exatamente oito caracteres, exiba-o diretamente seguido por uma nova linha.

```
?> ./methods_everywhere.py | cat -e
none$
?> ./methods_everywhere.py 'lol' 'physically' 'backpack' | cat -e
lolZZZZZ$
physical$
backpack$
?>
```

Método shrink: Use slices.

Método enlarge: Semelhante aos arrays, você pode adicionar caracteres a uma string usando o operador de concatenação

Capítulo IX

Exercício 05: scope_that

	Exercício : 05
Não pode tocar nisso!	
Pasta de entrega : $ex05/$	
Arquivos para entregar : `scope_that.py`	
Funções ou bibliotecas autorizadas : `Todos`	

- Crie um programa chamado `scope_that.py` que não receba nenhum parâmetro.
- Certifique-se de que este programa seja executável.
- Dentro do programa, crie um método chamado `add_one` que receba um parâmetro e adicione 1 ao parâmetro.
- Inicialize uma variável no corpo do programa, exiba-a e, em seguida, chame o método que adiciona 1.
- Exiba sua variável novamente no corpo do programa.
- O que você observa?

Capítulo X

Entrega e avaliação por pares

- Você deve ter a pasta `missao` na raiz do seu diretório inicial.
- Dentro da pasta `missao`, você deve ter uma pasta chamada `module8`.
- Dentro da pasta `module8`, você deve ter uma pasta para cada exercício.
- O exercício 00 deve estar na pasta `ex00`, o exercício 01 na pasta `ex01`, etc.
- Cada pasta de exercício deve conter os arquivos solicitados na tarefa.

`Observe que, durante sua defesa, qualquer coisa que não esteja presente na pasta do módulo não será verificado.`