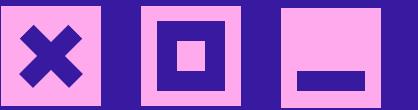


GRUPO KATIE



MINICURSO BÁSICO DE PYTHON

By Katie



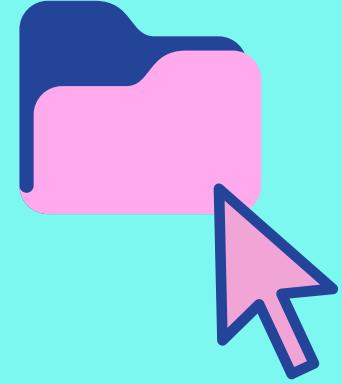
REALIZAÇÃO:



APOIO:



- @katie.ufal
- katie@ic.ufal.br
- github.com/GrupoKatie



OI GENTIE, TURU
POM??

NO VÍDEO DE HOJ... OPS
NA AULA DE HOJE VAMOS
APRENDER PYTHON :)



ROTEIRO



0. 0 QUE VOCÊ PRECISA SABER



2. CONFIGURANDO O AMBIENTE



4. OPERADORES



6. FUNÇÕES



8. REPETIÇÕES



1. 0 QUE É PYTHON

3. VARIÁVEIS E TIPOS DE DADOS

5. CONDICIONAIS

7. LISTA

9. DICIONÁRIOS





O QUE É PROGRAMAÇÃO?

É o ato de dizer ao computador o que queremos que ele faça através de uma sequência de passos que chamamos de algoritmo.



O QUE É LINGUAGEM DE PROGRAMAÇÃO?

É a linguagem que faz o intermédio entre a máquina e o programador.



O QUE VAMOS FAZER COM NOSSOS CÓDIGOS?

No inicio nossos códigos parecem até ser inúteis, mas servem para darmos vida aos programas, e através deles que aprendemos a usar a linguagens de programação. Assim, podemos criar programas grandes, com uma equipe colaborando com o código.

O QUE É PYTHON?

É uma linguagem de programação multiuso e muito intuitiva.

POR QUE USAR PYTHON?

É uma linguagem de fácil e rápido aprendizado. Essa linguagem é mais próxima da nossa, e por isso facilita o entendimento.

ONDE PODEMOS USAR PYTHON?

- Desenvolvimento de sistemas web
- Ciência de dados
- Inteligência Artificial



0. INTRODUÇÃO

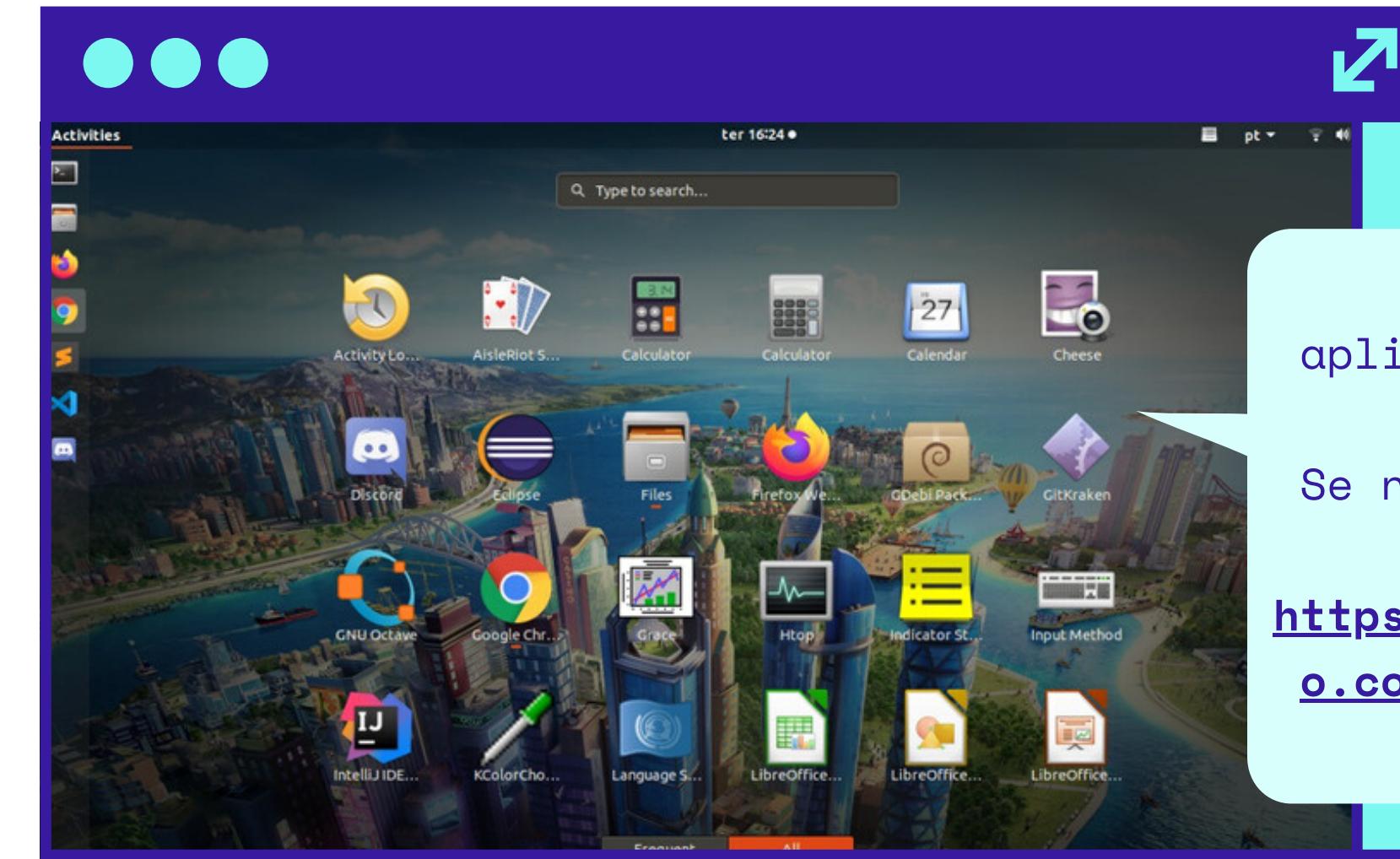
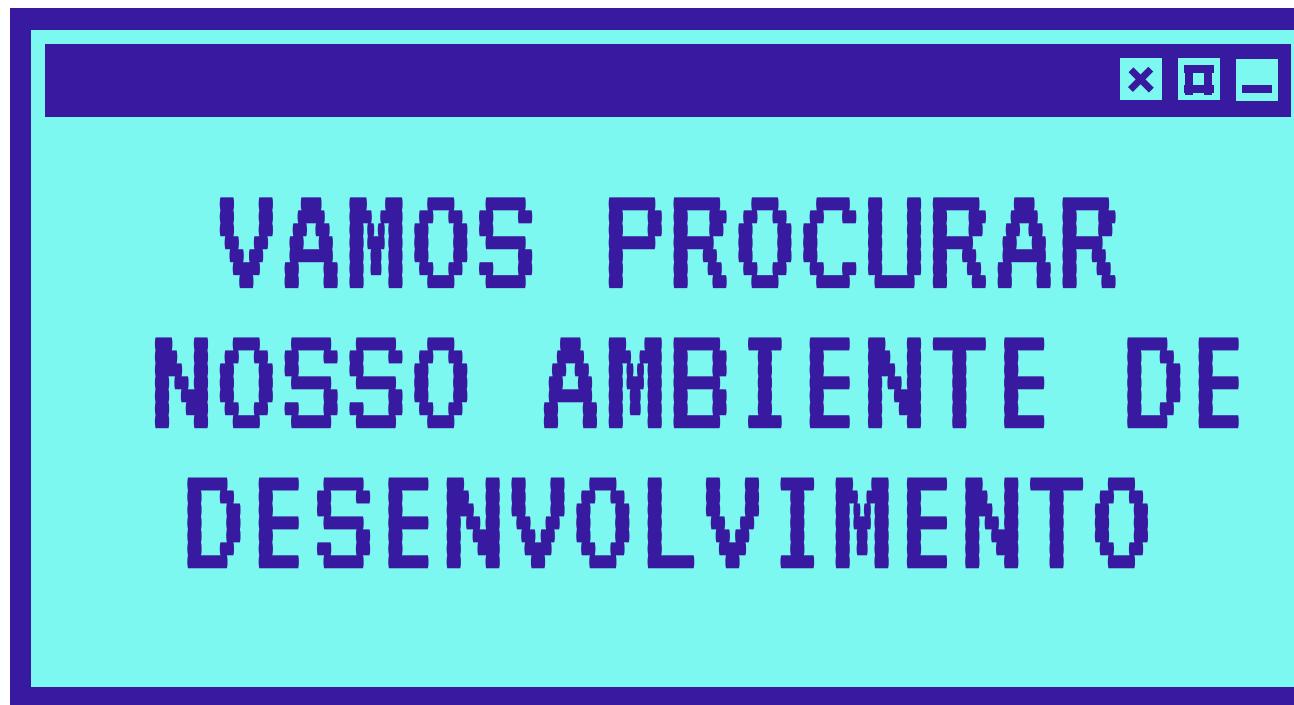
GRUPO KATIE

MINICURSO PYTHON

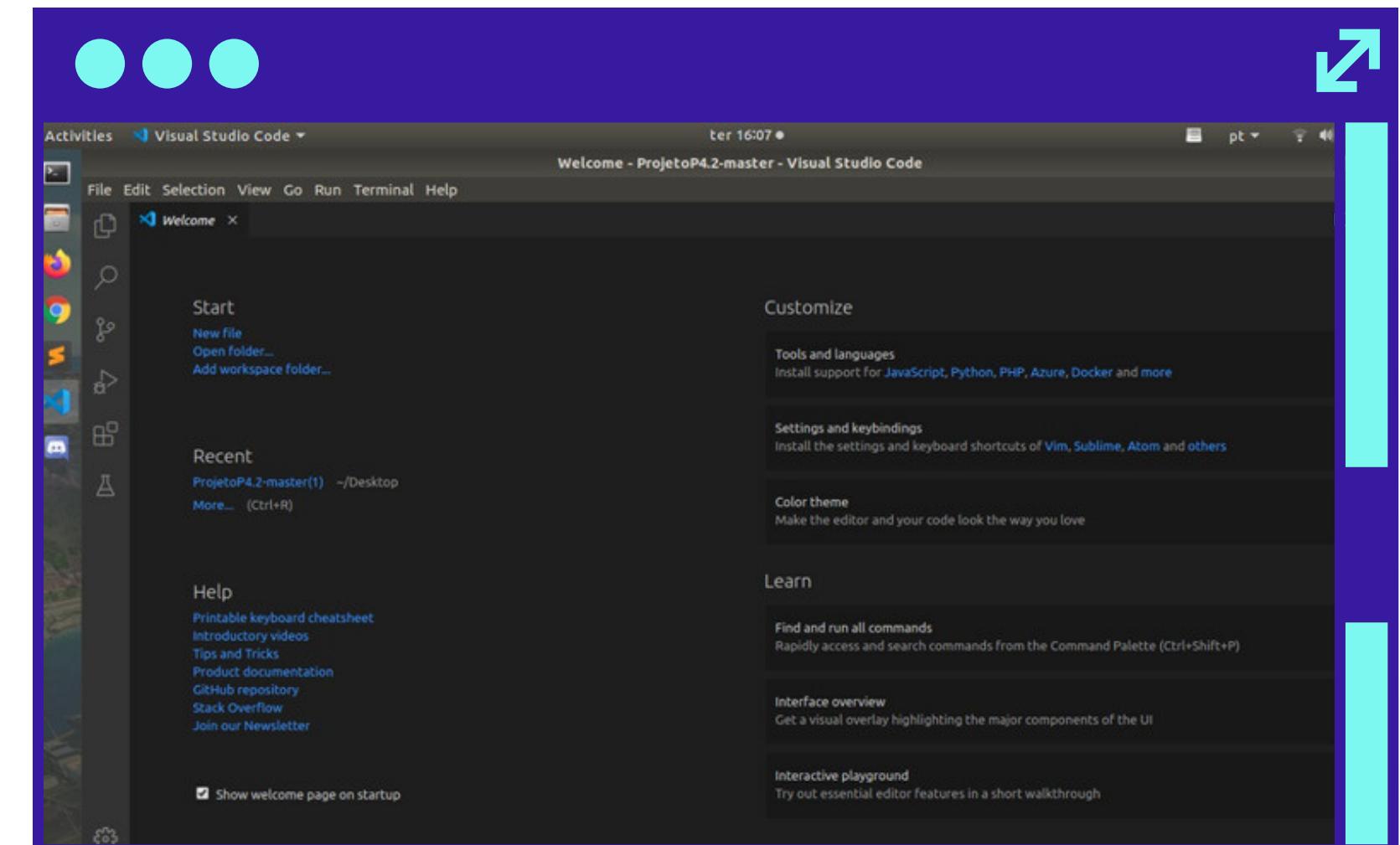
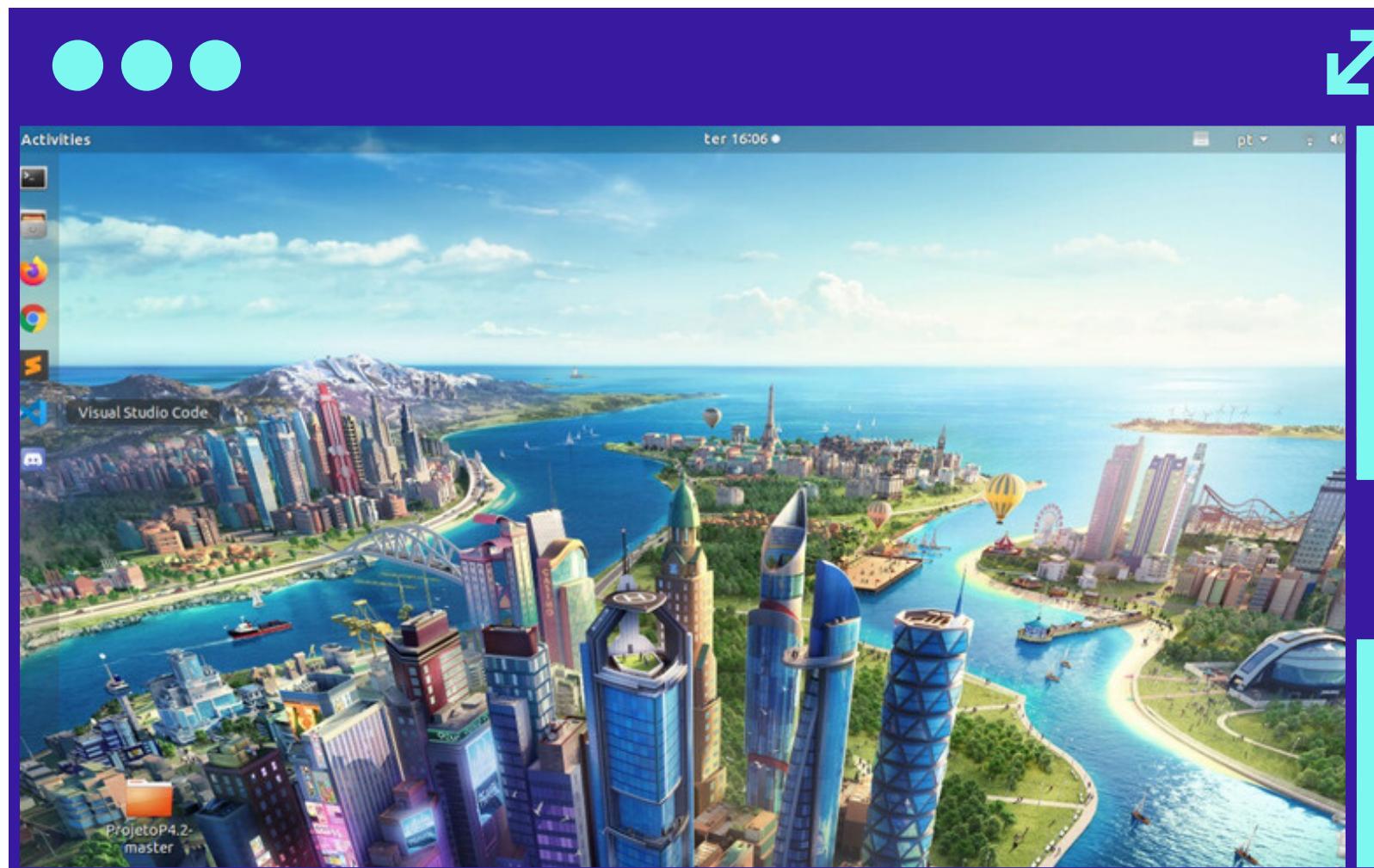
2º PASSO

CONFIGURAR O AMBIENTE

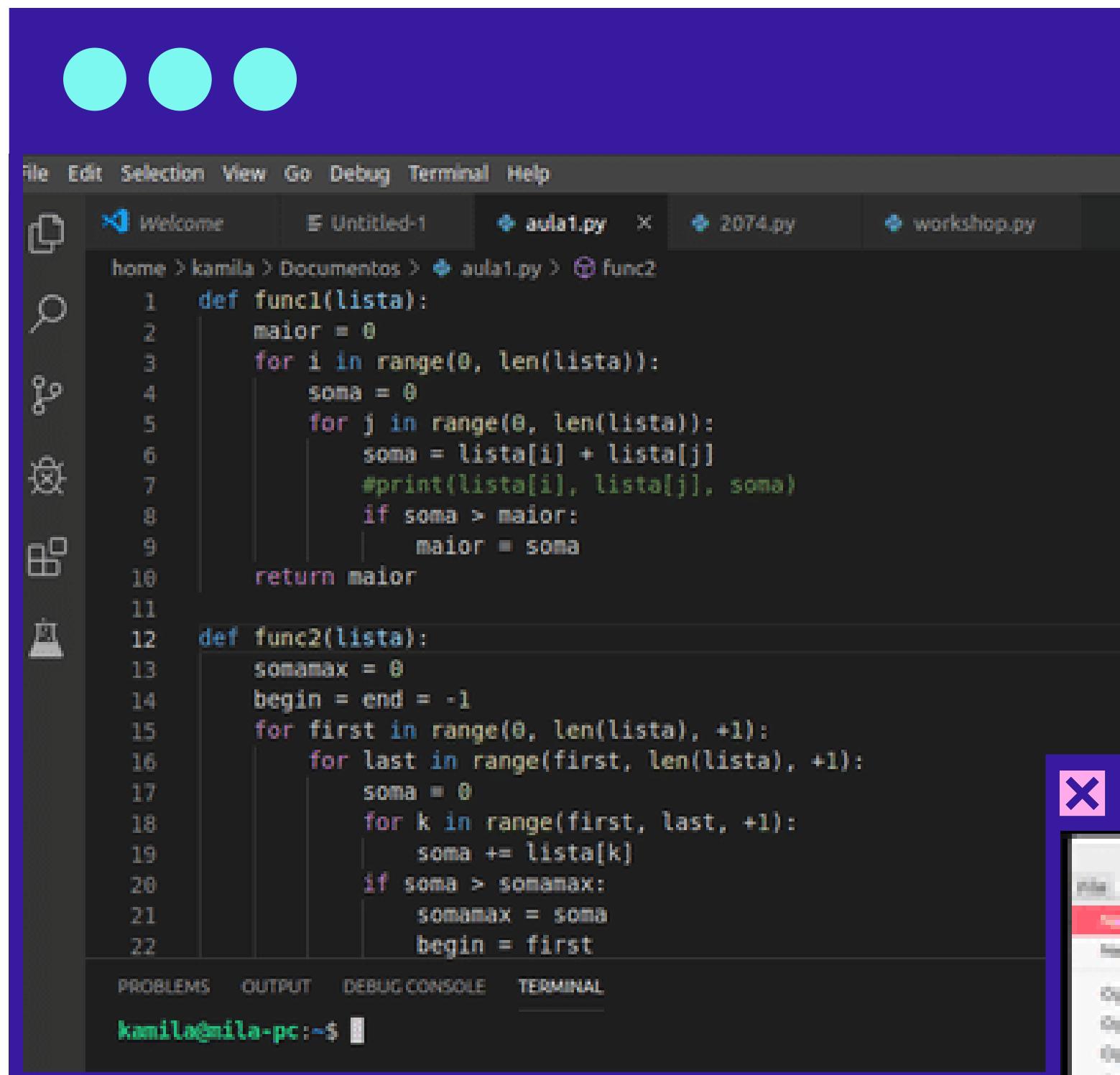
2. CONFIGURANDO O AMBIENTE



Vá no menu de aplicativos e pesquise por "vscode". Se não tiver instalado baixe no site <https://code.visualstudio.com> e instale na sua máquina.

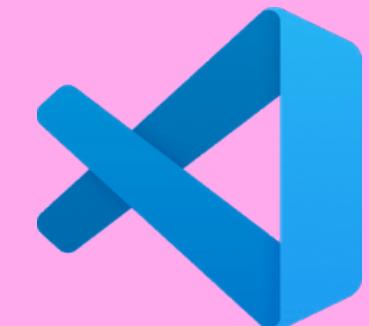


2. CONFIGURANDO O AMBIENTE

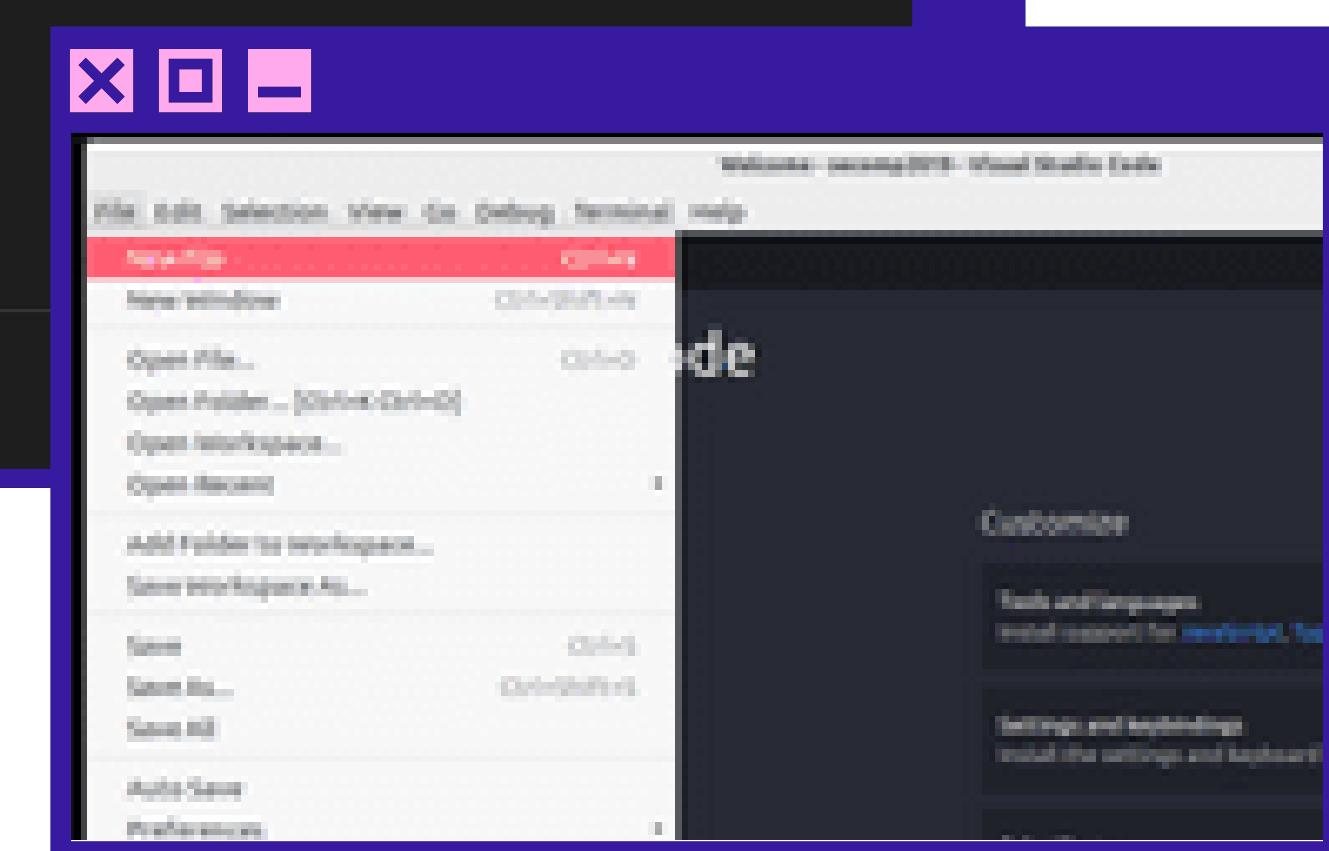


O editor de texto *Visual Studio Code* (VSCode) é o ambiente onde iremos escrever nossos incríveis códigos:

1. abra o programa VSCode
 2. Clique em FILE
 3. New File

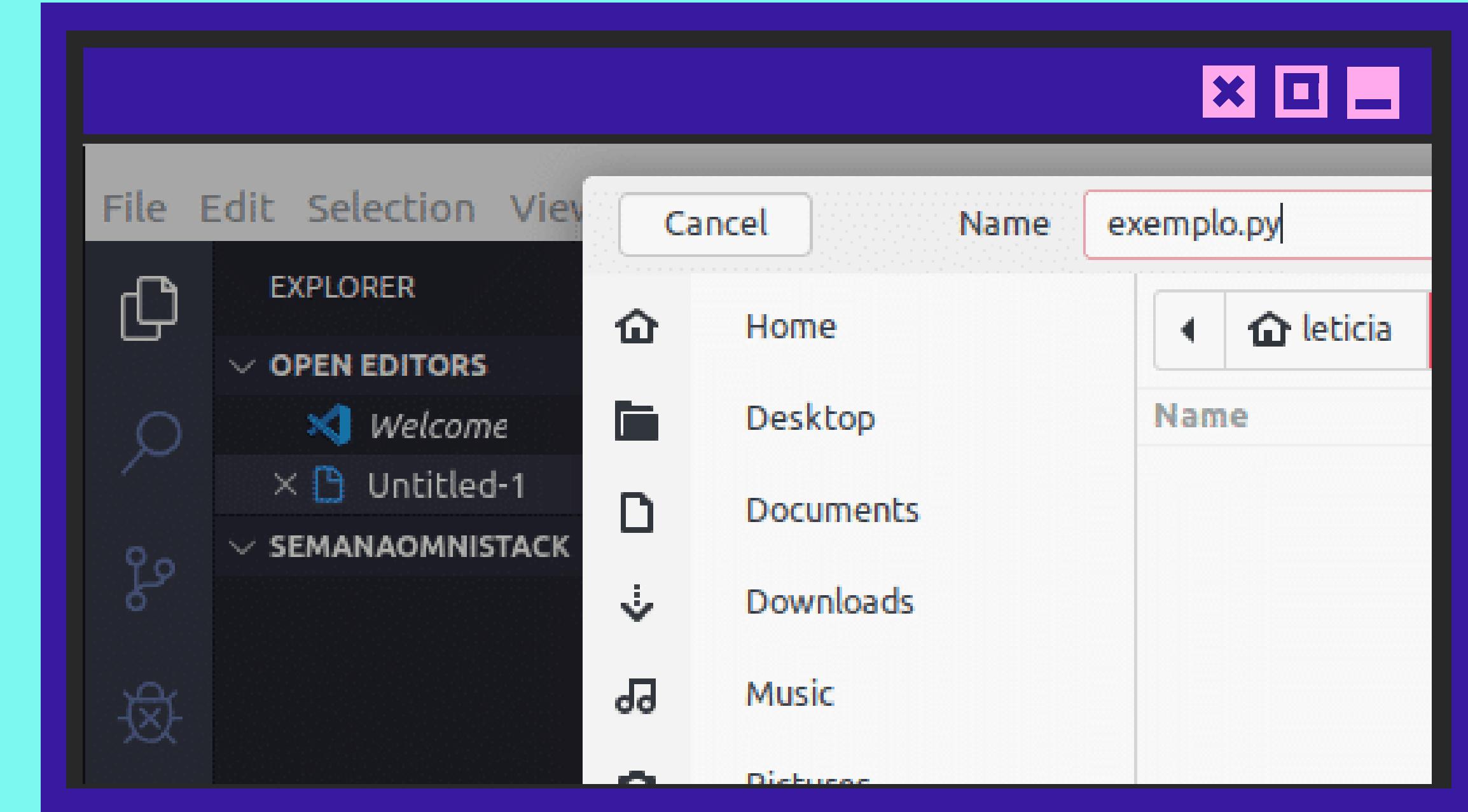
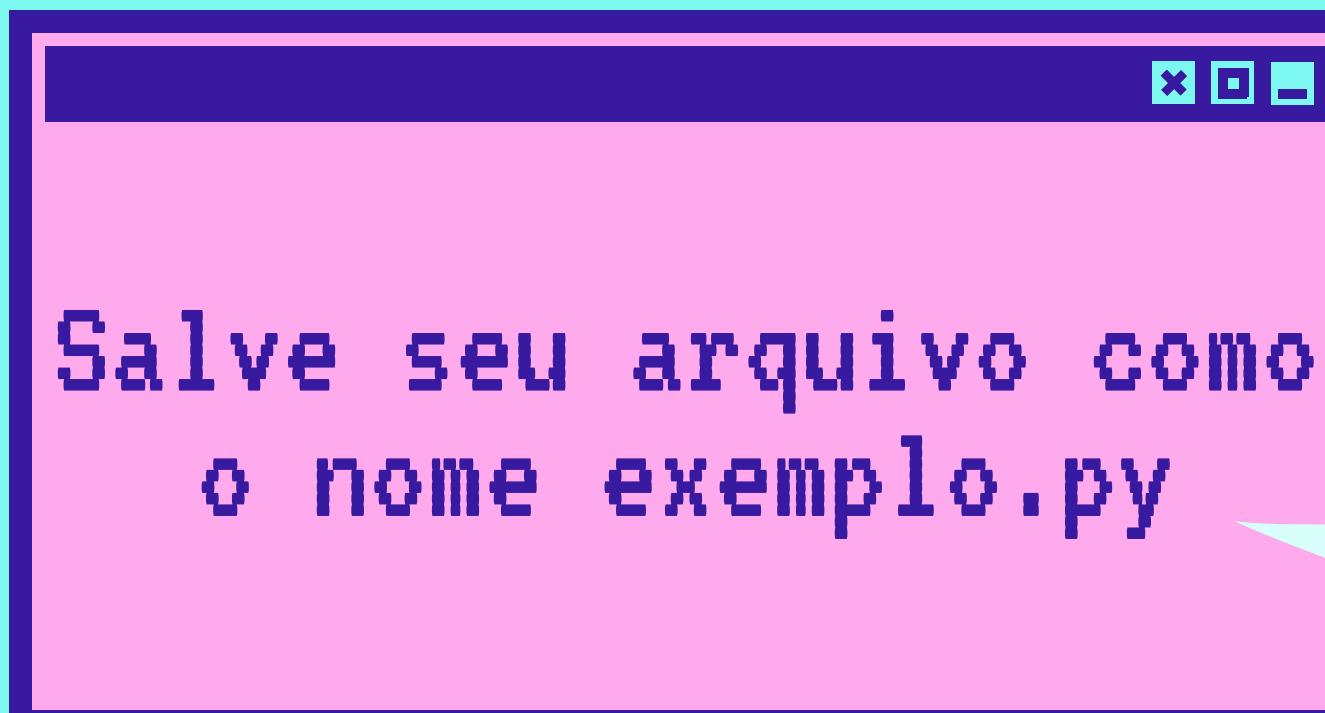


Visual Studio Code



Clique na opção "**File**" e depois em "**New file**" para criar um novo arquivo em branco

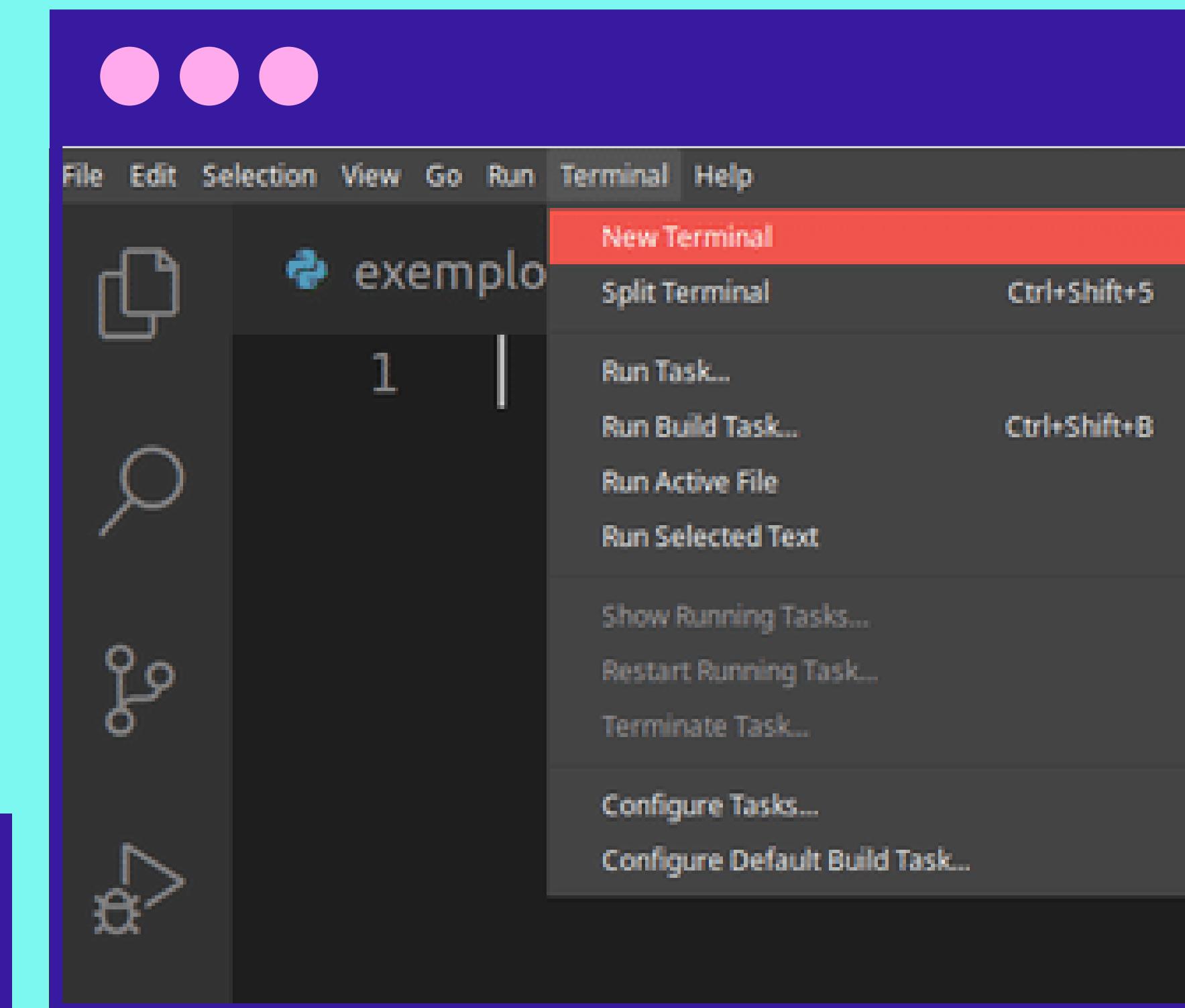
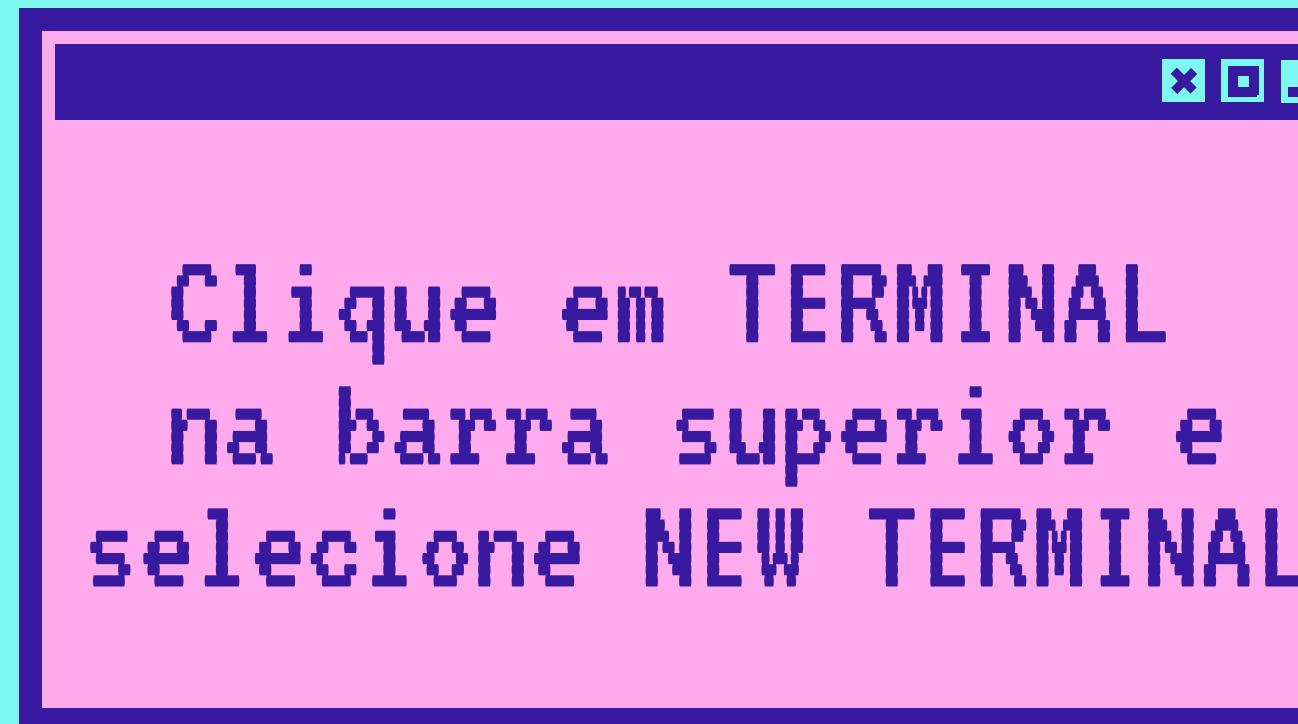
2. CONFIGURANDO O AMBIENTE



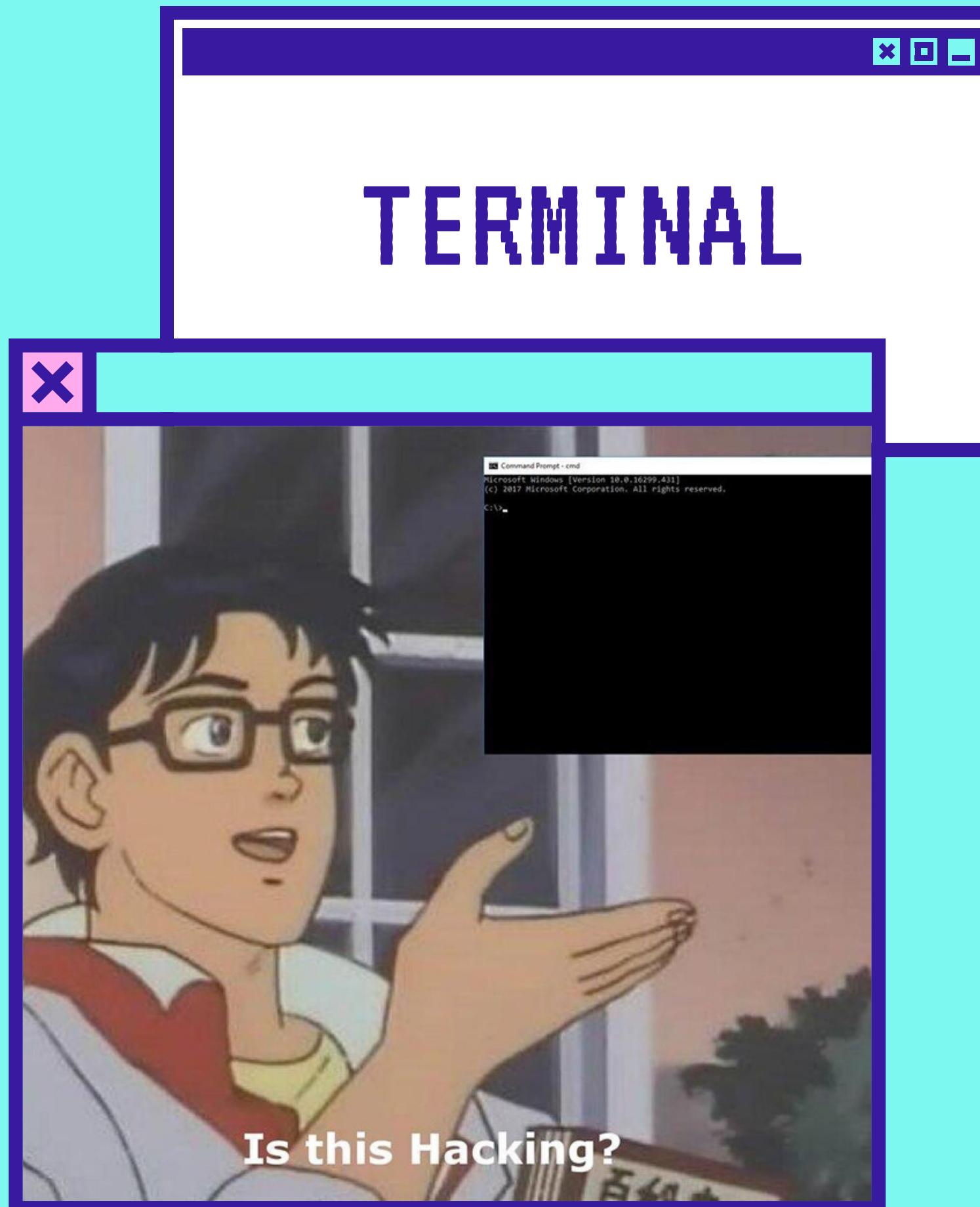
.py é a extensão que usamos para que o computador entenda que esse arquivo é um programa em python

Se você usa alguma distribuição Linux, o python 3 virá junto com o sistema. Se você usa Windows vai precisar instalar o python na sua máquina, use esse tutorial para baixá-lo e instalá-lo:
<https://python.org.br/installacao-windows/>

2. CONFIGURANDO O AMBIENTE



Mas o que é Terminal?

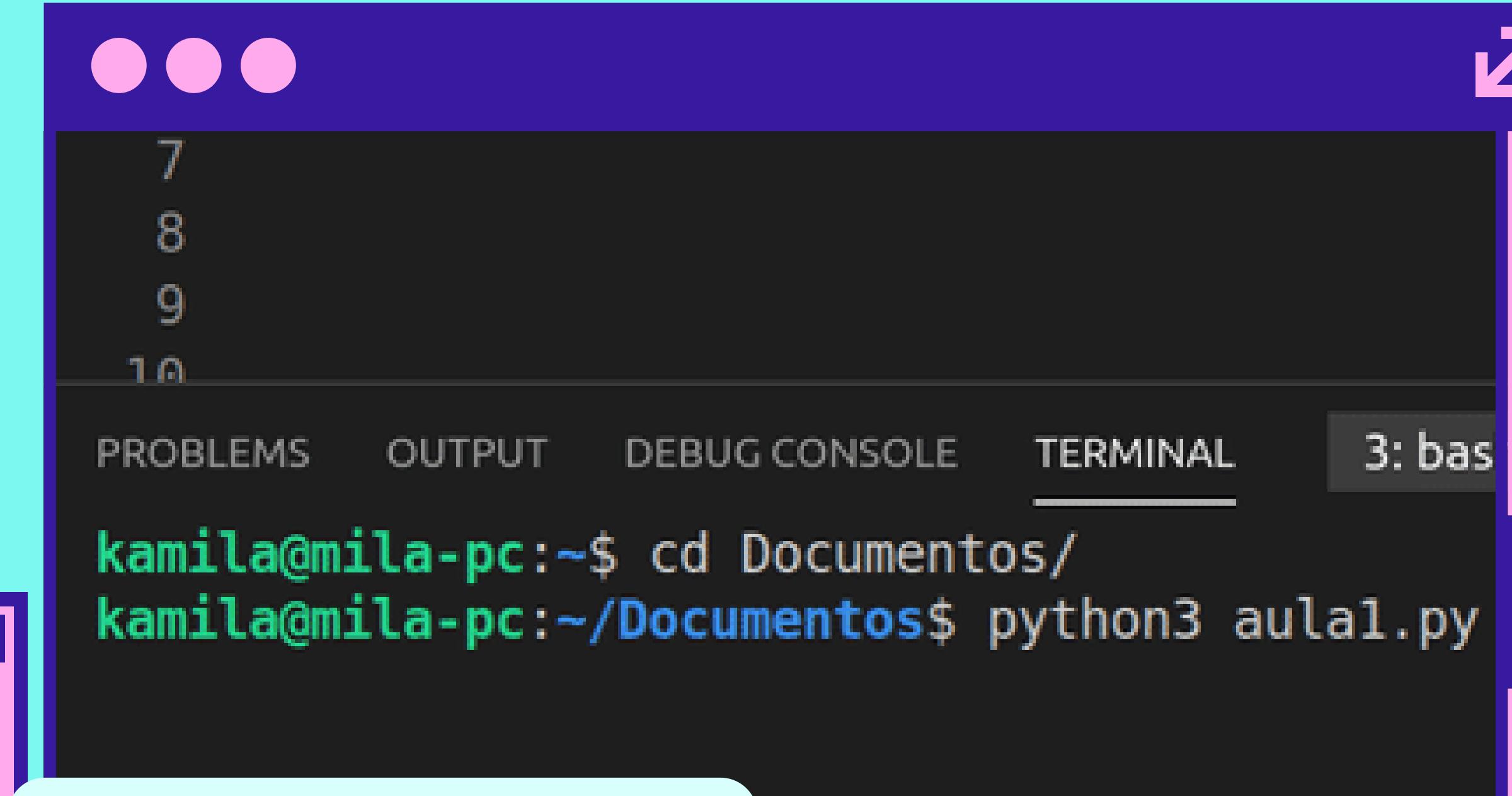


FAMOSA TELA PRETA USADA EM FILMES DE HACKERS

Basicamente, é um programa que você digita comandos para falar diretamente com um computador, dar instruções e visualizar informações dele. Ou seja, ele serve para você executar tarefas em sua máquina sem utilizar a interface gráfica (pastas, ícones ou mouse). Todos os comandos são executados através de pura digitação de texto.

EXECUTANDO UM CÓDIGO

Para que o terminal localize seu código, entre na pasta onde você salvou o arquivo com o comando:
cd Documentos para executar o código use:
python3 aula1.py



A screenshot of a terminal window titled "Terminal". The window has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL, with TERMINAL being the active tab. The terminal shows the following command-line session:

```
kamila@mila-pc:~$ cd Documentos/  
kamila@mila-pc:~/Documentos$ python3 aula1.py
```

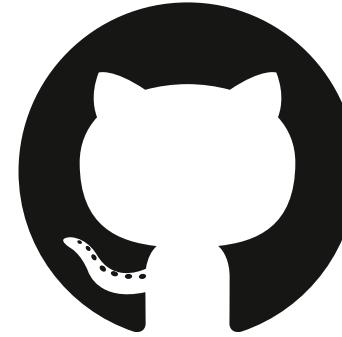
The status bar at the bottom right indicates "Ln 14". A callout bubble from the text block on the left points to the "cd Documentos" command in the terminal.

Vamos entender esse comando? "**cd**" é o comando que serve para mudar de pastas, "**Documentos**" é o nome da pasta. Então lembre de se certificar que está entrando na pasta correta.

Outra coisa, executando esses comandos não vamos ter resposta porque não escrevemos nenhum código, ok?!

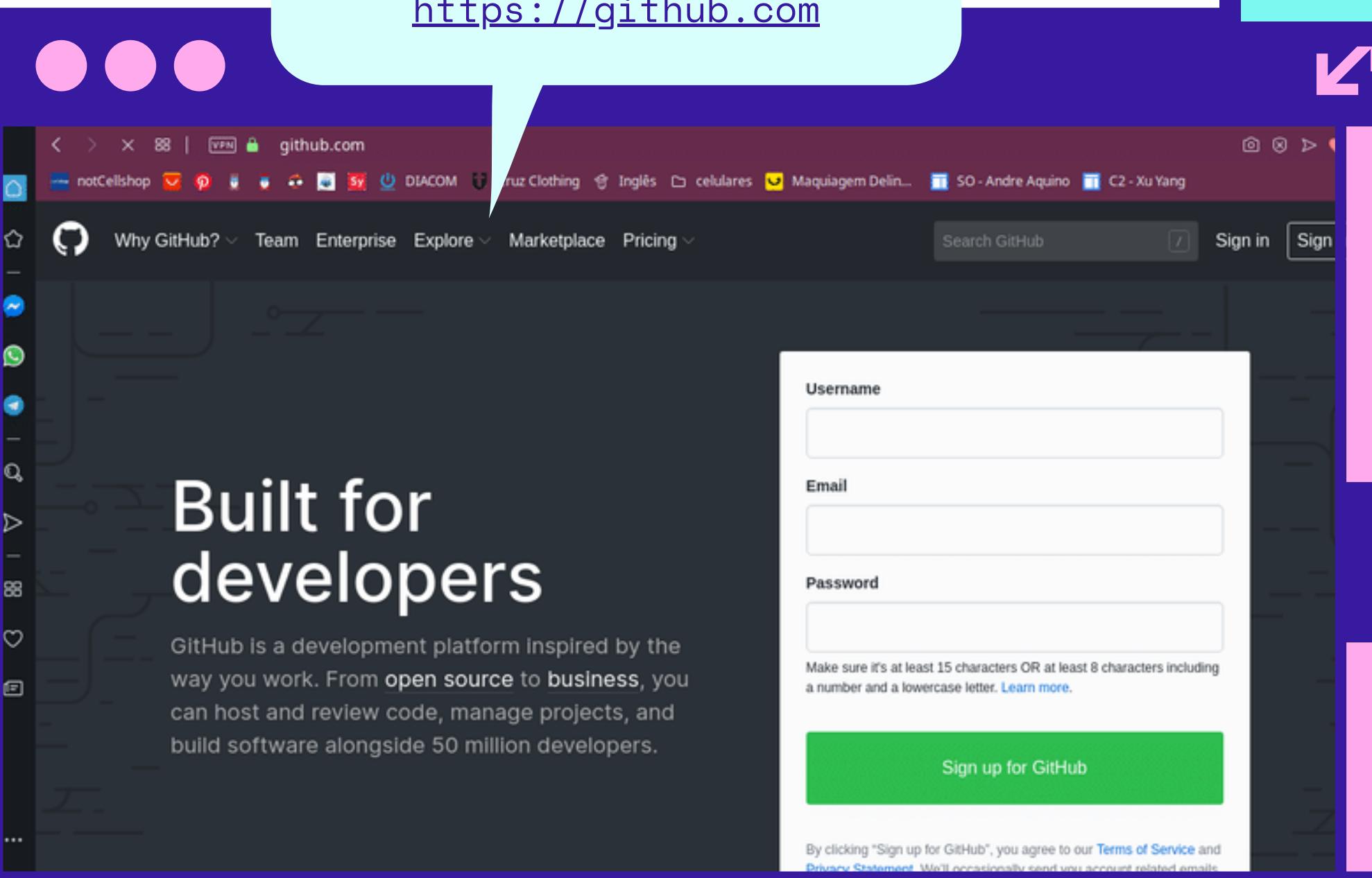
Para sair de uma pasta use o comando "**cd ..**" sem as aspas

BÔNUS :)



Vamos falar sobre GIT e GITHUB?!

Crie uma conta no Github:
<https://github.com>



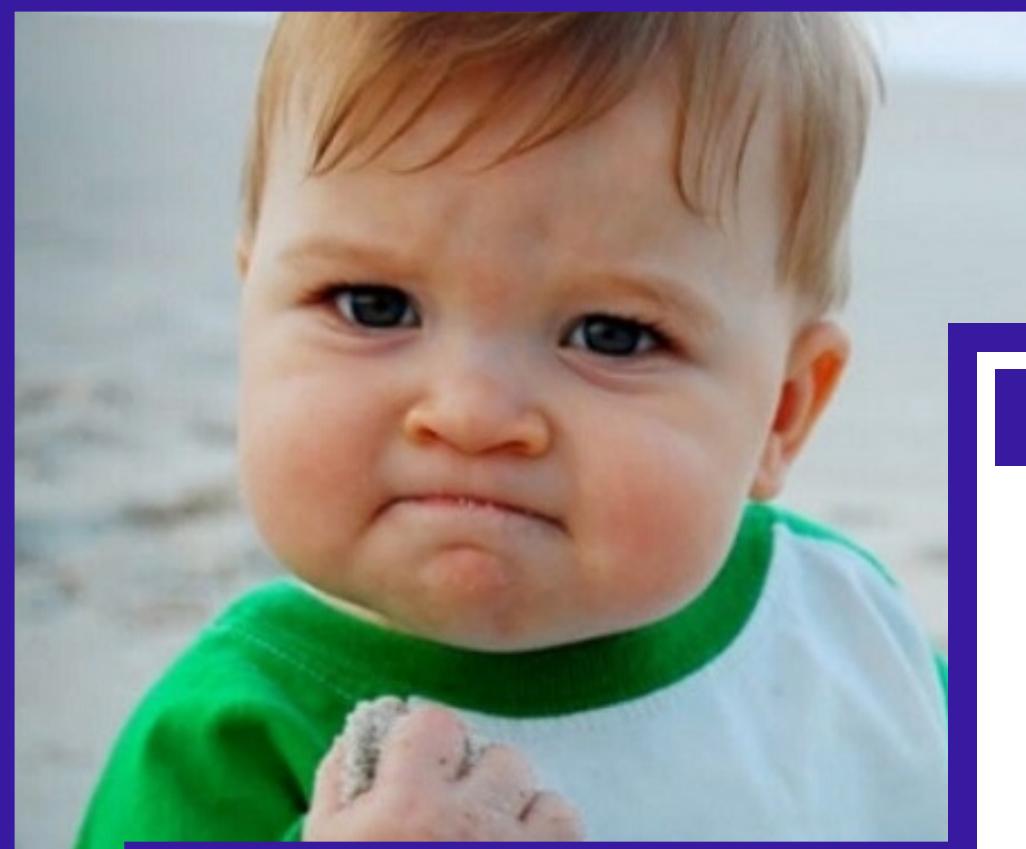
A screenshot of a web browser showing the GitHub sign-up page. The URL 'github.com' is visible in the address bar. The page features a dark header with the GitHub logo and navigation links like 'Why GitHub?', 'Team', 'Enterprise', 'Explore', 'Marketplace', and 'Pricing'. Below the header, there's a large 'Sign up for GitHub' button. The main form has fields for 'Username', 'Email', and 'Password', with a note below stating 'Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter.' A green 'Sign up for GitHub' button is at the bottom. The background of the slide shows a faint image of the GitHub homepage with the slogan 'Built for developers'.

Inicialmente, escrevemos códigos unicamente em nossa máquina, mas quando criamos grandes aplicações, geralmente precisamos da colaboração de outras pessoas. Nesse caso, precisaremos trabalhar em equipe e escrever códigos juntos. Então, como podemos resolver isso? Isso aí, usando Git e Github!

Git é um sistema de controle de versão de arquivos. Ou seja, é um facilitador para compartilharmos nossos códigos entre outros programadores sem perder nada e com a possibilidade de restaurar versões anteriores de um arquivo.

Github é uma plataforma de hospedagem de código-fonte com controle de versão usando o Git. Além disso, é uma rede social de programadores! Você pode seguir e ser seguido por outros programadores, e como seus códigos podem ficar armazenados lá, existe a possibilidade de deixar ficar visíveis ou não para a comunidade.

X □ -



...



VAMOS COMEÇAAAR!

○ ○ ○ =



^

▼

□

VARIÁVEIS E TIPOS DE DADOS

GRUPO KATIE

MINICURSO PYTHON

3º PASSO

VARIÁVEIS E TIPOS DE DADOS

TIPOS DE DADOS



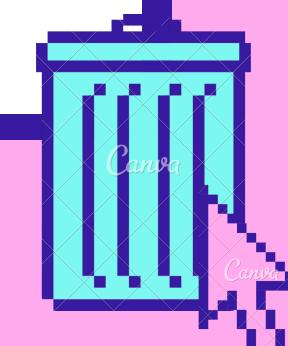
VARIÁVEIS



Uma **variável** é um **espaço na memória** do computador destinado a um **dado** que é alterado (ou não) durante a execução do algoritmo.

Até o momento, conhecemos, basicamente, três tipos de dados:

1. **inteiros** (`int`)
2. **ponto flutuante** (`float`)
3. **string** (`str`).



INTEIRO

Representa um número inteiro.

BOOLEAN

Representa um dos dois valores,
TRUE ou FALSE.



STRING

Representa uma lista de
caracteres.*

FLOAT

Representa um número decimal.

- Para fins didáticos, vamos considerar em nosso tutorial que String é apenas uma lista.



REFACE O EXEMPLO COM SEUS DADOS

Execute seu código no terminal usando o comando:
python3 aula1.py

```
1 nome = 'Katie'  
2 idade = 30  
3 altura = 1.59  
4 casada = True  
5  
6 print(nome)  
7 print(idade)  
8 print(altura)  
9 print(casada)
```

Se você se esqueceu como executar um programa, volte ao tópico "*2. Configurando o ambiente*". As linhas onde aparece **print** vão ser explicadas mais adiante, ok?!

OPERADORES

GUPO KATIE

MINICURSO PYTHON

4º PASSO

OPERADORES

The diagram illustrates various arithmetic operators within a window frame. On the left side, five operations are listed: Soma, Subtração, Divisão, Multiplicação, and Igualdade. To the right of these, their corresponding symbols are displayed: +, -, /, *, and ==.

OPERADORES ARITMÉTICOS

Soma +

Subtração -

Divisão /

Multiplicação *

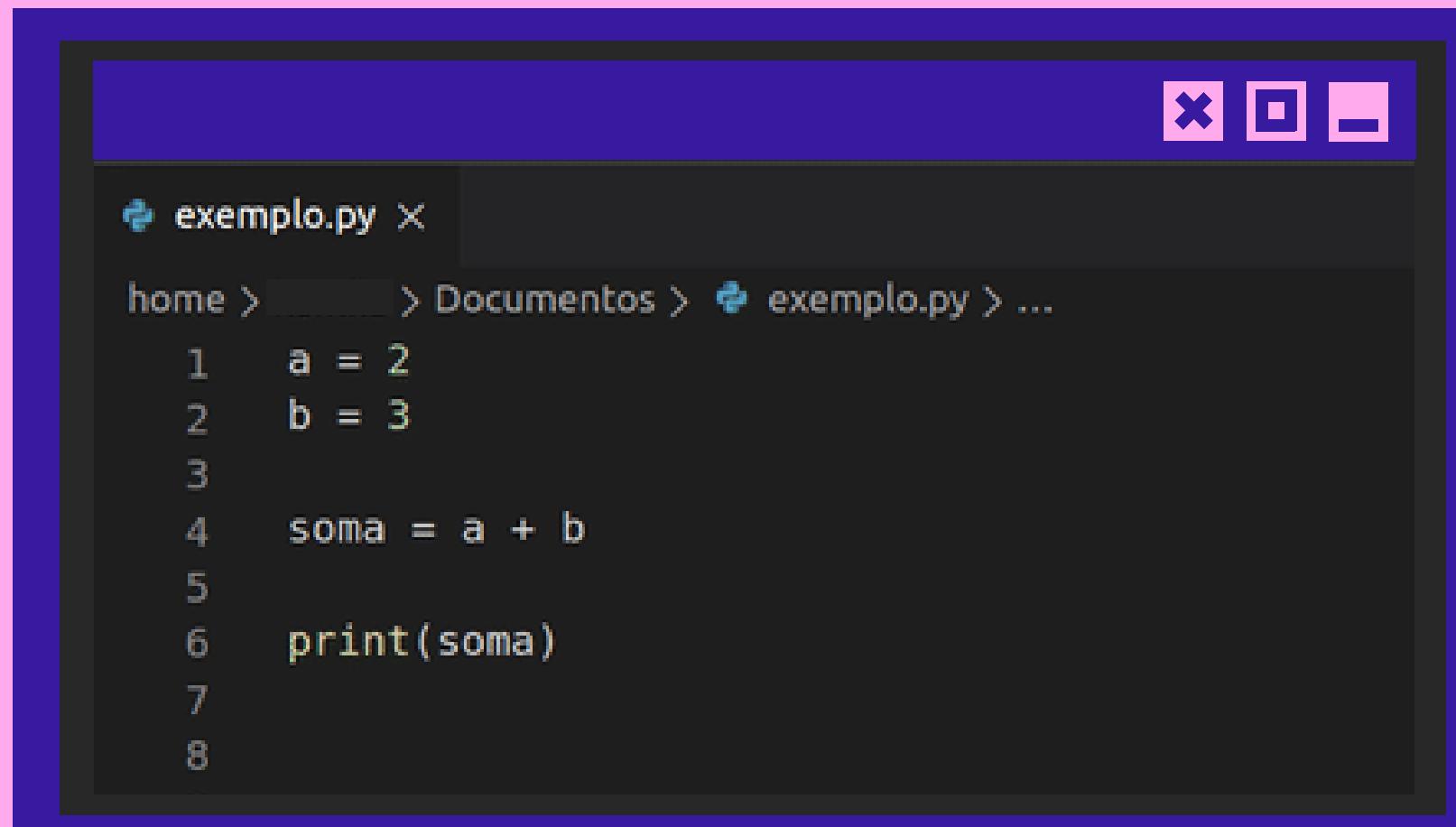
Igualdade ==

4. OPERADORES

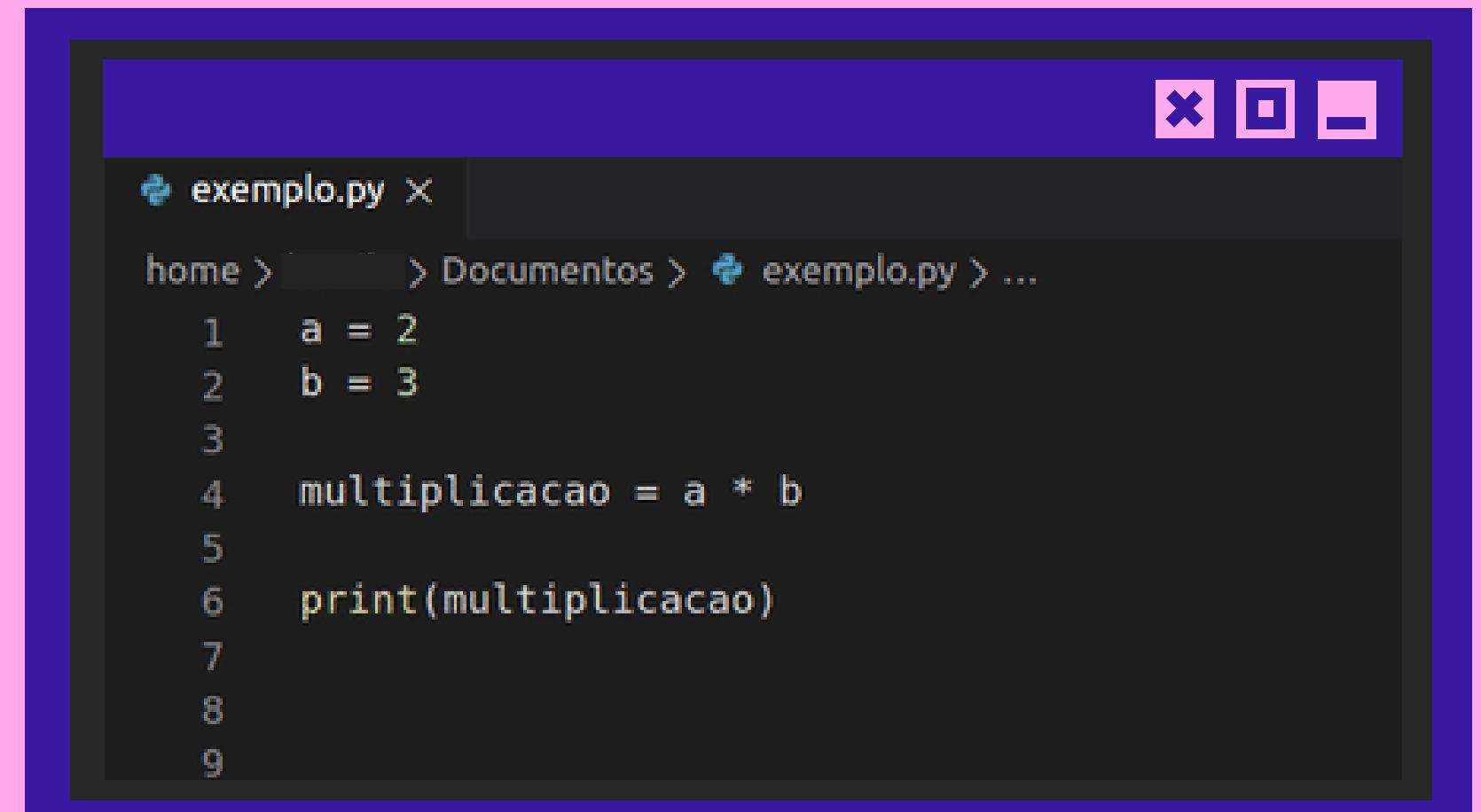


```
exemplo.py - Untitled (Workspace) - Visual Studio Code
File Edit Selection View Go Run Terminal Help
+ exemplo.py ×
home > > Documentos > exemplo.py > ...
1 a = 2
2 b = 3
3
4 divisao = a / b
5
6 print(divisao)
7
8
```

Usando o operador aritméticos de divisão "/" você terá como resultado um número decimal, do tipo **float**. Se você quiser uma divisão exata use o operador "://" e terá com resultado um número inteiro, do tipo **int**. Se quiser o resto de uma divisão, use "%".



```
+ exemplo.py ×
home > > Documentos > exemplo.py > ...
1 a = 2
2 b = 3
3
4 soma = a + b
5
6 print(soma)
7
8
```



```
+ exemplo.py ×
home > > Documentos > exemplo.py > ...
1 a = 2
2 b = 3
3
4 multiplicacao = a * b
5
6 print(multiplicacao)
7
8
9
```

PRINT

Função usada para "imprimir" ou simplesmente mostrar no terminal o que queremos que o nosso código execute.

```
PRINT('HELLO WORLD')
```

INPUT

Função usada para receber qualquer dado que o usuário digite em seu teclado.

```
NOME = INPUT()
```

o que é isso?
A função INPUT retorna o valor digitado pelo usuário, e esse valor deve ser guardado em alguma variável, certo?!
Nesse exemplo, nós guardamos na variável **NOME**.



CONDICIONAIS

GRUPO KATIE

MINICURSO PYTHON

5º PASSO

CONDICIONAIS

Tipos

Operadores

`==`

Igualdade

`!=`

Desigualdade

`>`

Comparação

`<`

Comparação

`>=`

Comparação

`<=`

Comparação

Valor

Verifica a igualdade entre dois valores

Verifica a diferença entre dois valores

Verificar se o valor de A é maior que o valor de B

Verificar se o valor de A é menor que o valor de B

Verificar se o valor de A é maior ou igual o valor de B

Verificar se o valor de A é menor ou igual o valor de B

PODEMOS COMPARAR COISAS DA SEGUINTE FORMA:

```
1 a = 2
2 b = 4
3
4 if a > b:
5     print(a, 'é maior que ', b)
6
7 else:
8     print(b, 'é maior que ', a)
9
10
11
```

```
1 nome1 = 'Elisa'
2 nome2 = 'Eliana'
3
4 if nome1 == nome2:
5     print('Iguais')
6
7 else:
8     print('Diferente')
9
10
```

Escreva o código na sua máquina e execute no terminal.

Cast

```
1 a = int(input())
2 b = int(input())
3
4 soma = a + b
5 print(soma)
```

Toda informação que recebemos do usuário é armazenada em forma de string. Então, para realizarmos as operações com os tipos de dados que aprendemos, utilizaremos um **cast**, como no exemplo ao lado.

Você pode fazer cast em qualquer tipo de variável, se quiser transformar **float** em **int**, pode ser da seguinte forma:

Ex.: numero = 2.34

a = int(numero)

agora teremos a = 2

Cast é um tipo de conversão de dados. No exemplo ao lado, temos uma string e queremos que ela seja do tipo int.

APLICAÇÃO

GRUPO KATIE

MINICURSO PYTHON

APLICAÇÃO 1

FAÇA UM PROGRAMA QUE RECEBA AS
CARACTERÍSTICAS DE UMA PESSOA E RETORNE
SEU IMC - ÍNDICE DE MASSA CORPORAL

ENTRADA: PESO (EM KG) E ALTURA (EM M);

EX: 77.27

1.60

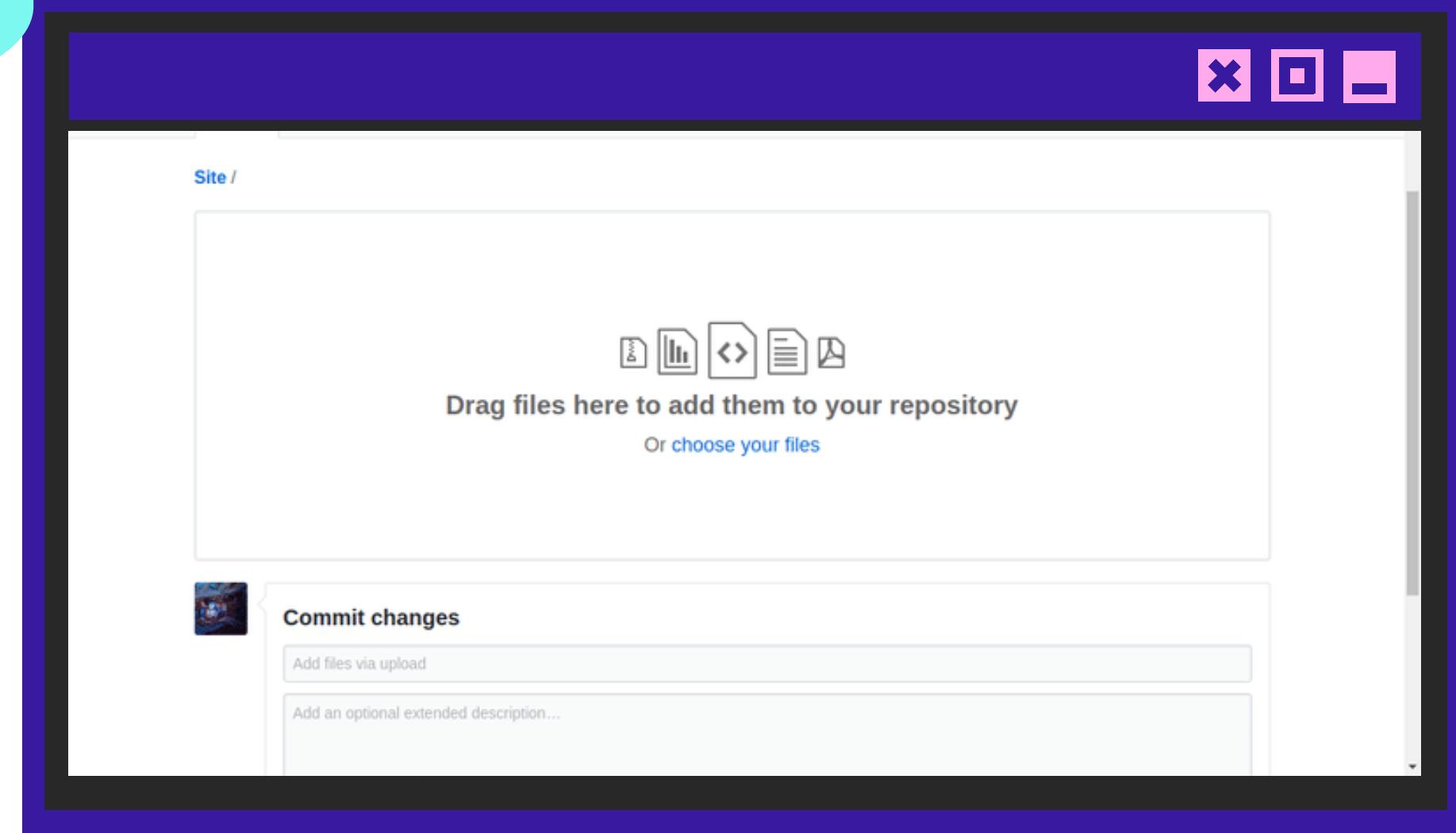
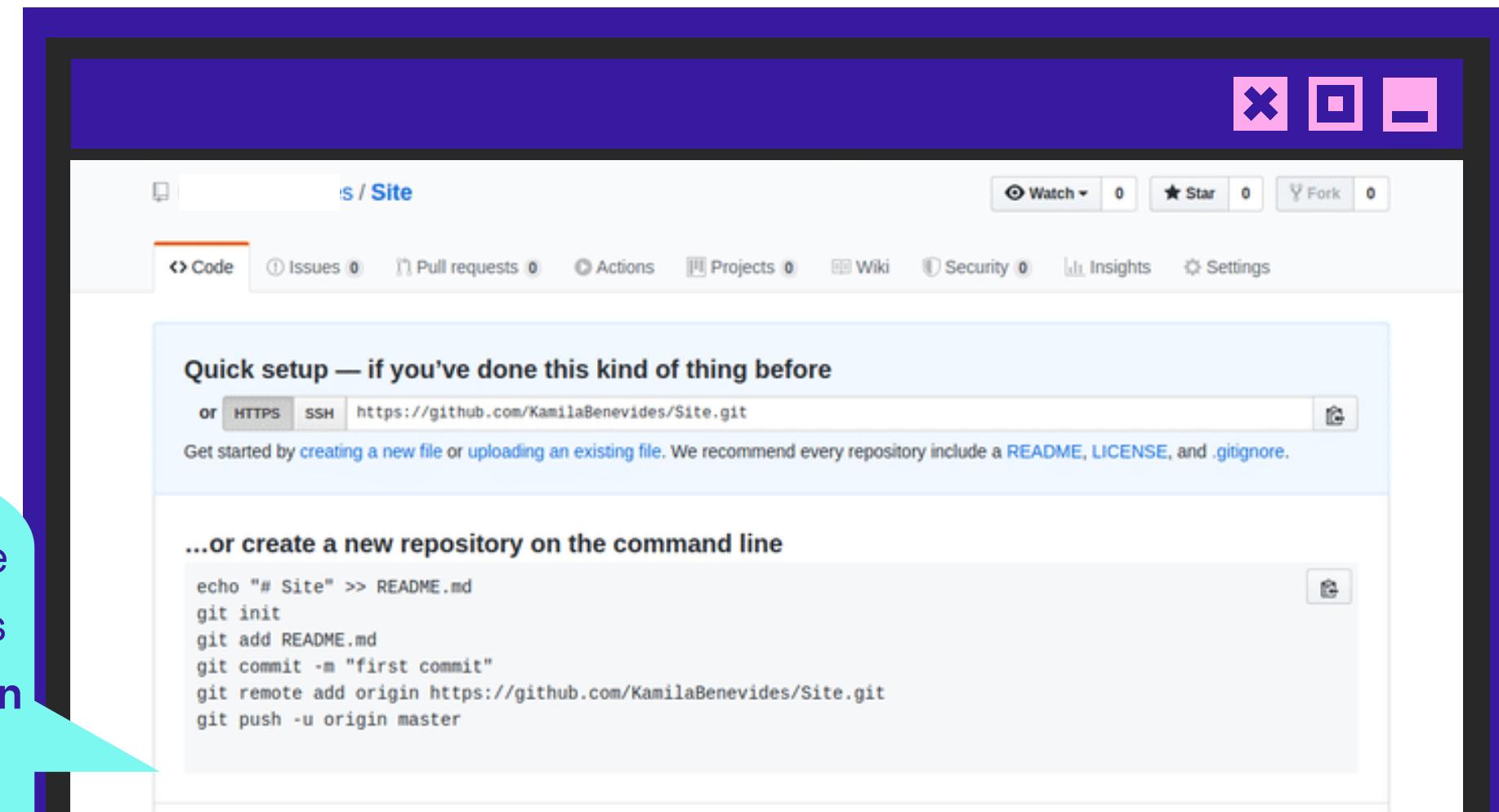
SAÍDA: O IMC;

EX: 30.1835

VAMOS GUARDAR NOSSO CÓDIGO NO GITHUB!

Crie um **novo repositório**, logo aparecerá essa tela, mas calma! nesse momento não vamos precisar usar todos esses comandos, clique em **uploading an existing file** e aparecerá uma tela igual a tela abaixo, agora você pode fazer o uploading do seu código

Você pode enviar códigos um por um, manualmente, da forma que estamos ensinando. Para adicionar com linhas de comando pelo terminal use esse tutorial:
<http://guides.railsgirls.com/guides-ptbr/github>



APLICAÇÃO

GRUPO KATIE

MINICURSO PYTHON

APLICAÇÃO 2

FAÇA UM PROGRAMA QUE CONVERTA METROS PARA
CENTÍMETROS.

ENTRADA: VALOR EM METROS;
EX: 50

SAÍDA: VALOR EM CENTÍMETROS;
EX: 5000

APLICAÇÃO

GRUPO KATIE

MINICURSO PYTHON

APLICAÇÃO 3

FAÇA UM PROGRAMA QUE CONVERTA UM NÚMERO DE
SEGUNDOS NO NÚMERO EQUIVALENTE DE HORAS,
MINUTOS E SEGUNDOS.

ENTRADA: UM NÚMERO DE SEGUNDOS;
EX: 7201

SAÍDA: NÚMERO DE HORAS,
MINUTOS E SEGUNDOS;
EX: 2h 0m 1s

APLICAÇÃO

GRUPO KATIE

MINICURSO PYTHON

APLICAÇÃO 4

A PARTIR DE 3 NOTAS FORNECIDAS DE UM ALUNO, INFORME SE ELE FOI APROVADO, FICOU DE RECUPERAÇÃO OU FOI REPROVADO.

MÉDIA DE APROVAÇÃO: ≥ 7.0 ;

MÉDIA DE RECUPERAÇÃO: ≥ 5.0 E < 7.0 ;

MÉDIA DE REPROVAÇÃO: < 5.0 ;

ENTRADA: TRÊS NÚMEROS INTEIROS

EX: 5.5

7

4.3

SAÍDA: A SITUAÇÃO DO ALUNO;

EX: "Ficou de recuperação"

Boa! Você chegou na
metade do curso. Vamos
dar continuidade ao
tutorial?



FUNÇÕES

GRUPO KATIE

MINICURSO PYTHON

6º PASSO

FUNÇÕES



Função:

Uma função é uma sequência de comandos que executa alguma tarefa, usamos geralmente para quebrar o código em blocos ou executar determinada tarefa mais de uma vez

FUNÇÕES

No exemplo ao lado, temos uma função chamada **soma**. Seu código começa a execução na linha 5, onde temos as duas variáveis **a=2** e **b=3** e uma variável **resultado**. A variável resultado recebe o retorno da função. O que essa função faz? Ela recebe os dois parâmetros **numero1** e **numero2** e faz a operação de soma, logo na linha 3 ela retorna a variável **soma** para a função principal, e em seguida mostra na tela o resultado, na linha 8.

```
1 def soma(numero1, numero2):  
2     soma = numero1 + numero2  
3     return soma  
4  
5 a = 2  
6 b = 3  
7 resultado = soma(a, b)  
8 print(resultado)
```

Calma que vamos explicar o que é parâmetro e retorno =)

6. FUNÇÕES

```
x □ -  
exemplo.py x  
home >     > Documentos > exemplo.py > ...  
1 def idade(anos):  
2     print('Ela tem', anos, 'anos.')  
3  
4 a = 25  
5  
6 idade(a)  
7
```

Veja que essa função não tem **return**, Você pode fazer uma função sem retorno, desde que não esqueça de citar ela no código, ou "chamar" ela, como visto na linha 6.

Fique atento à sintaxe da função. Sempre inicie com a palavra **def** e depois de fechar) use :

Então o que é **parâmetro**? São os argumentos, valores das variáveis, que serão usados nas funções. Um parâmetro como o exemplo ao lado, "anos" é apenas representado pela variável na função, ou seja, o seu valor não muda pois é apenas uma referência.

Return é o que a função vai devolver como resultado.

```
x □ -  
1 def somaVarios(num1, num2):  
2     soma = num1 + num2  
3     print(num1, '+', num2, '=', soma)  
4  
5 a = 25  
6 b = 9  
7 c = 3  
8 d = 10  
9  
10 somaVarios(a, b)  
11 somaVarios(a, c)  
12 somaVarios(a, d)
```

Nesse código, nós queremos que seja feita uma soma várias vezes com variáveis diferentes, mas para não ficarmos 'copiando e colando' o mesmo pedaço de códigos várias vezes, usamos uma função para otimizar nosso código.

FUNÇÕES ÚTEIS

**len(Lista)****sum(lista)****max(lista)****min(lista)****split()****append(elemento)**

As funções **len**, **sum**, **max**, **min**, recebem, respectivamente, uma lista como parâmetro e retornam: a quantidade de elementos, a soma dos valores, o maior valor, menor valor.

A função **split***, por padrão, tem como parâmetro o caracter espaço ' ', mas você pode colocar outros caracteres como parâmetro também. O que ela faz? ela "quebra" a string de entrada toda vez que encontrar o parâmetro que você colocou.

EX: `entrada = input().split()`
a entrada será uma lista, cada espaço lido terá uma quebra.

A função **append*** recebe como parâmetro um elemento, pode ser um inteiro, string, lista, e esse elemento é colocado no fim de uma outra lista.

EX: `numeros.append(2)`
o número 2 será colocado no final da lista **numeros**

próxima página, teremos exemplos do uso de **split** e **append**.

- Para fins didáticos, vamos considerar que Split e Append são funções.

A estrutura lista será ensinada no próximo passo.

6. FUNÇÕES

exemplo.py ×

```
home > |       > Documentos > exemplo.py > ...
1  entrada = input().split('.')
2
3  for i in range(len(entrada)):
4      print(entrada[i])
5
6
7
```

Teste a entrada com:
123.456.789 e veja a
saída. Troque o parâmetro
da função **split** com
outros caracteres.

Aqui a string '**Atalaia**'
vai ser adicionada no
final da lista **cidades**.

exemplo.py ×

```
home >       > Documentos > exemplo.py > ...
1  cidades = ['Maceio', 'Arapiraca', 'Rio Largo']
2
3  cidades.append('Atalaia')
4
5  print(cidades)
6
7
8
9
```

LISTAS

GUPO KATIE

MINICURSO PYTHON

7^a PASSO

LISTAS



```
1 nomes = ['Betina', 'Neymar', 'Nazare']  
2  
3 print(nomes)  
4
```

Temos no exemplo uma lista de nomes.
Teste usando nomes da sua preferência

LISTAS



ASSIM COMO ESCREVEMOS UM NÚMERO OU CARACTERE, AGORA PODEMOS ARMAZENAR ESSES DADOS AGRUPADOS EM FORMA DE LISTAS.



```
1 impares = [1, 3, 5, 7, 9]  
2 pares = [2, 4]  
3  
4 seq = impares + pares  
5 print(seq)  
6
```

Podemos brincar com operações utilizando listas. No exemplo à esquerda, temos a concatenação (um agrupamento) das duas listas, impares e pares, execute esse código e veja o resultado

REPETIÇÕES

GRUPO KATIE

MINICURSO PYTHON

8º PASSO

REPETIÇÕES

Repetições

Executar uma determinada ação repetidas vezes, Para isso, utilizaremos as estruturas '**for**' e '**while**'.



A estrutura que vai ser executada repetidas vezes até que uma condição seja atingida.



8. REPETIÇÕES

```
x □ -  
exemplo.py ×  
home > kamila > Documentos > exemplo.py > ...  
1 for i in range(0, 5, 1):  
2     print(i)  
3  
4  
5
```

Aqui imprimimos na tela os números de 0 a 4.

A função **range** controla a repetição. A variável **i** começa em 0, o **for** é executado repetidas vezes até a variável chegar no valor 5, a cada rodada do **for**, a variável **i** é incrementada uma vez

Para ficar claro, como parâmetro da função **range**, temos (inicio da repetição, fim da repetição, de quanto vai ser incrementado a variável de centro)

```
x □ -  
exemplo.py ×  
home > kamila > Documentos > exemplo.py > ...  
1 lista = ['Paris', 'Dubai', 'Veneza']  
2 i = 0  
3  
4 while i < len(lista):  
5     print(lista[i])  
6     i = i + 1  
7  
8
```

Aqui temos uma lista com strings.

Nosso **while** inicializa com a variável **i** (com valor 0, linha 2) para o controle do loop, então temos que a repetição começa em 0 e vai até o tamanho da lista.

Dentro da repetição, escrevemos uma string de cada vez. E não se esqueça de somar +1 na variável **i**, caso contrário entrará em loop infinito.

Lembrando que a função **len()** retorna a quantidade de itens da lista

BÔNUS :)



Podemos incrementar nossa variável de controle com outras sintaxes:

i = i + 1

ou

i += 1

Também podemos usar outros operadores.

Podemos trabalhar com String usando lista:

```
exemplo.py
home > Documentos > exemplo.py ...
1 palavra = 'Paralelepipedo'
2
3 for i in range(len(palavra)):
4     print(palavra[i])
5
```

Execute na sua máquina e veja a saída do programa

Observe no exemplo ao acima; na função **range**, temos apenas um parâmetro, o de limite da repetição, podemos escrever apenas ela, pois o python já deixa, por padrão, o inicio como 0 e a incrementação ser +1.

DICIONÁRIO

GUPO KATIE

MINICURSO PYTHON

9º PASSO

DICIONÁRIO

DICIONARIOS

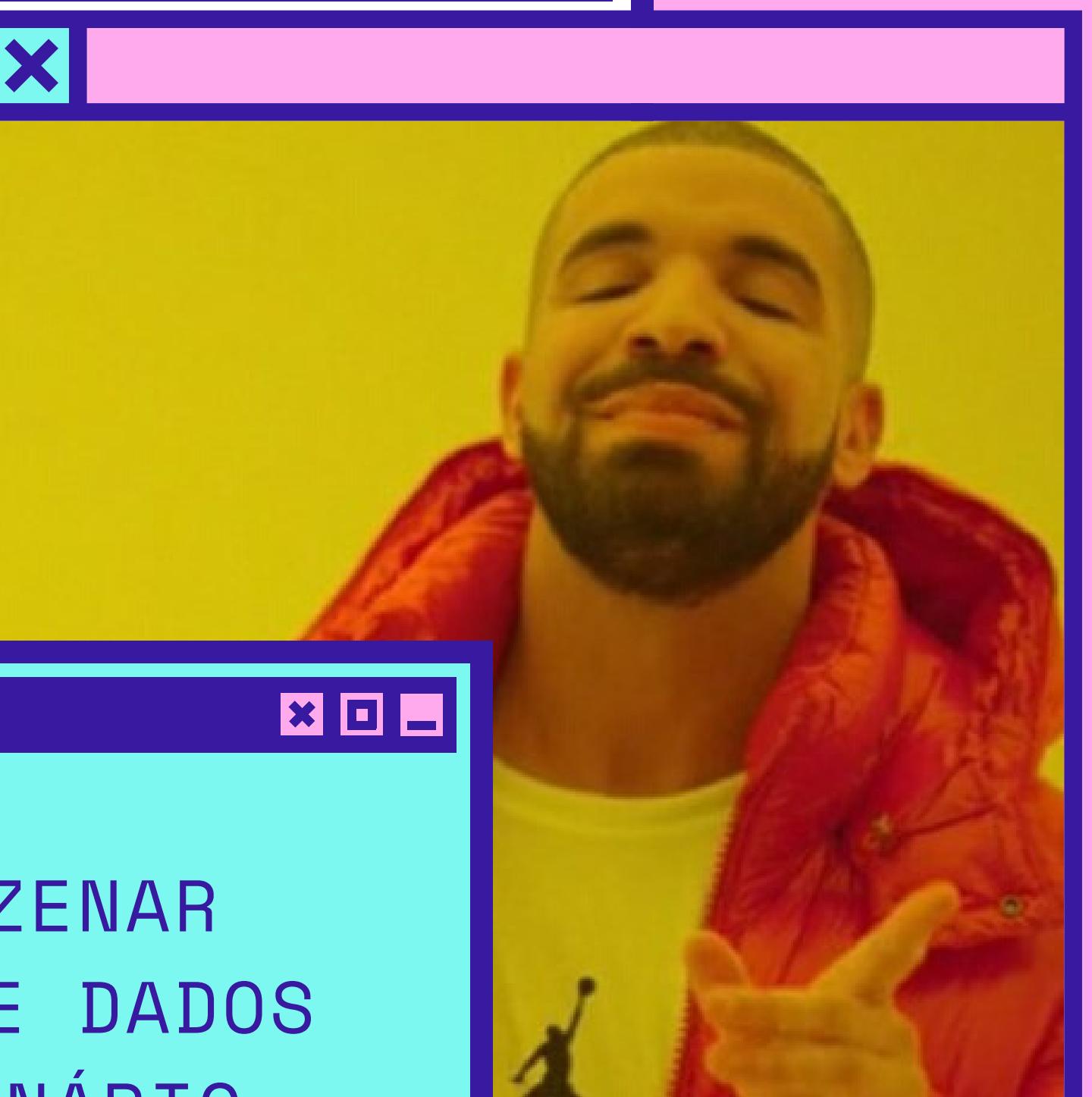
Agora podemos agrupar vários dados em blocos usando dicionário.

É semelhante a uma lista, mas agora temos chave e valor para identificação.

Chave é a identificação do elemento armazenado no dicionário.

Valor é o elemento armazenado no dicionário..

VC PODE ARMAZENAR VÁRIOS TIPOS DE DADOS NO MESMO DICIONÁRIO, ISSO É SUPER LEGAL NÉ?!



9. DICIONÁRIO



```
1 dicionario = {'nomeGaroto': 'Miguel', 'cidade': 'Paris', 'statusSocial': 'Rico', 'qtFilhos': [2,4,8]}\n2 print(dicionario['nomeGaroto'])\n3
```

chave 'nomeGaroto'
aponta para a string
'miguel'

chave 'cidade'
aponta para a string
'Paris'

chave 'statusSocial'
aponta para a string 'Rico'

chave 'qtFilhos'
aponta para a lista de
numeros '[2,4,8]'

Você pode alterar os
dados do seu dicionário.
Para acessar um elemento,
você utiliza a chave como
índice

Você pode imprimir o todas as
informações do dicionário:
print(dicionario)

Índice: Posição da
variável em uma
estrutura



```
3 dicionario['nomeGaroto'] = 'Gustavo'\n4 print(dicionario['nomeGaroto'])\n5
```

APLICAÇÃO

GRUPO KATIE

MINICURSO PYTHON

APLICAÇÃO 5

USANDO FUNÇÕES, FAÇA UM PROGRAMA QUE CALCULE A VELOCIDADE MÉDIA E A ACELERAÇÃO DE UM VEÍCULO. CONSIDERE O TEMPO DA ENTRADA PARA O CÁLCULO DA VELOCIDADE MÉDIA E DA ACELERAÇÃO. DA MESMA FORMA, CONSIDERE QUE A VELOCIDADE DA ENTRADA É A VARIAÇÃO DE VELOCIDADE.

Entrada: o espaço (metros), tempo (segundos);

EX: 1

20

Saída: a velocidade média (em m/s) e a aceleração (em m/s^2);

EX: 0.05

0.0025

APLICAÇÃO

GRUPO KATIE

MINICURSO PYTHON

APLICAÇÃO 6

FAÇA UM PROGRAMAS, QUE USE A ESTRUTURAS DE REPETIÇÃO WHILE PARA IMPRIMIR OS NÚMEROS PARES ENTRE 0 200 COM A ESTRUTURA DE REPETIÇÃO FOR IMPRIMA A QUANTIDADE DE NÚMEROS PARES ENTRE 0 E 200.

ENTRADA: NÃO HÁ ENTRADA;

SAÍDA: TODOS OS NÚMEROS PARES ENTRE 0 E 200 E QUANTIDADE DE NÚMEROS PARES = 100.

APLICAÇÃO

GRUPO KATIE

MINICURSO PYTHON

APLICAÇÃO 7

RECEBA UMA LISTA DE NÚMEROS E MOSTRE QUEM É O
MAIOR NÚMERO, O MENOR NÚMERO, E EM SEGUIDA
SOME TODOS ELES.

ENTRADA: UMA SEQUENCIA DE NÚMEROS
SAÍDA: O MAIOR, O MENOR E A SOMA DE TODOS.

EX.: 3 74 9 2 7 4 10

EX.:

APLICAÇÃO

GRUPO KATIE

MINICURSO PYTHON

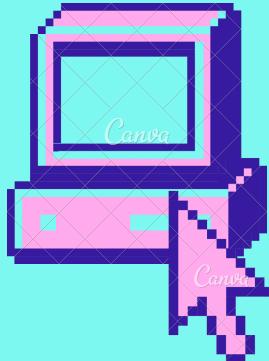
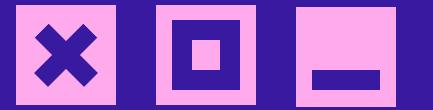
APLICAÇÃO 8

CRIE UM DICIONÁRIO COM SUAS INFORMAÇÕES (NOME, IDADE, CIDADE, ESTADO, UNIVERSIDADE, CURSO E PERÍODO) E IMPRIMA TRÊS INFORMAÇÕES NELE CONTIDAS.

ENTRADA: NÃO HÁ ENTRADA;

SAÍDA: TRÊS INFORMAÇÕES DE SUA ESCOLHA.

GRUPO KATIE

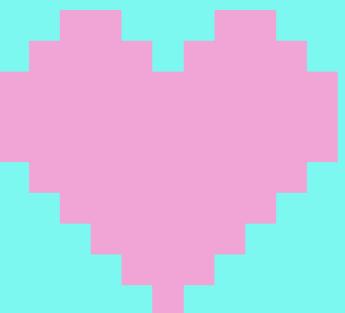


OBRIGADA GALERA,
VOÇES SÃO INCRÍVEIS!

@KATIE.UFAL

KATIE@IC.UFAL.BR

GITHUB.COM/GRUPOKATIE



POR ONDE CONTINUAR?

<https://python.org.br/introducao/>

Nesse link você encontra várias coisas úteis sobre *Python* e tutorias de diferentes níveis.

ONDE PRATICAR?

<http://www.thehuxley.com>

Programação é prática. The Huxley é uma plataforma que disponibiliza um banco de questões e você pode resolver usando várias linguagens de programação. Crie uma conta e comece a praticar!

CONHEÇA A COMUNIDADE PYLADIES BRASIL

<https://brasil.pyladies.com>

Grupo com foco em ajudar mais mulheres a tornarem-se participantes ativas e líderes da comunidade Python.

