

An illustration of a city at night, viewed from a rooftop. In the foreground, a man is sitting at a desk, working on a laptop. To his right, a cat is visible. In the background, a city skyline is visible with various buildings and a water tower. The scene is lit with a warm, orange glow, suggesting sunset or sunrise. The overall style is a flat, modern illustration.

# 1ª Rodada de Workshop dos Quarenteners

PYTHON (DIA 1)

# Referências

---

- Using Python for Research. Prof. JP Omnela, *Associate Professor of Biostatistics* (Harvard University). Disponível em [Link](#).
- Python for Data Science. Prof. Ilkay Altintas, *Chief Data Science Officer at the San Diego Supercomputer Center* (University of San Diego). Disponível em [Link](#).
- CodeAcademy: Course "Learn Python 2". Disponível em [Link](#).
- CoCalc: Collaborative Calculation and Data Science. Disponível em [Link](#).



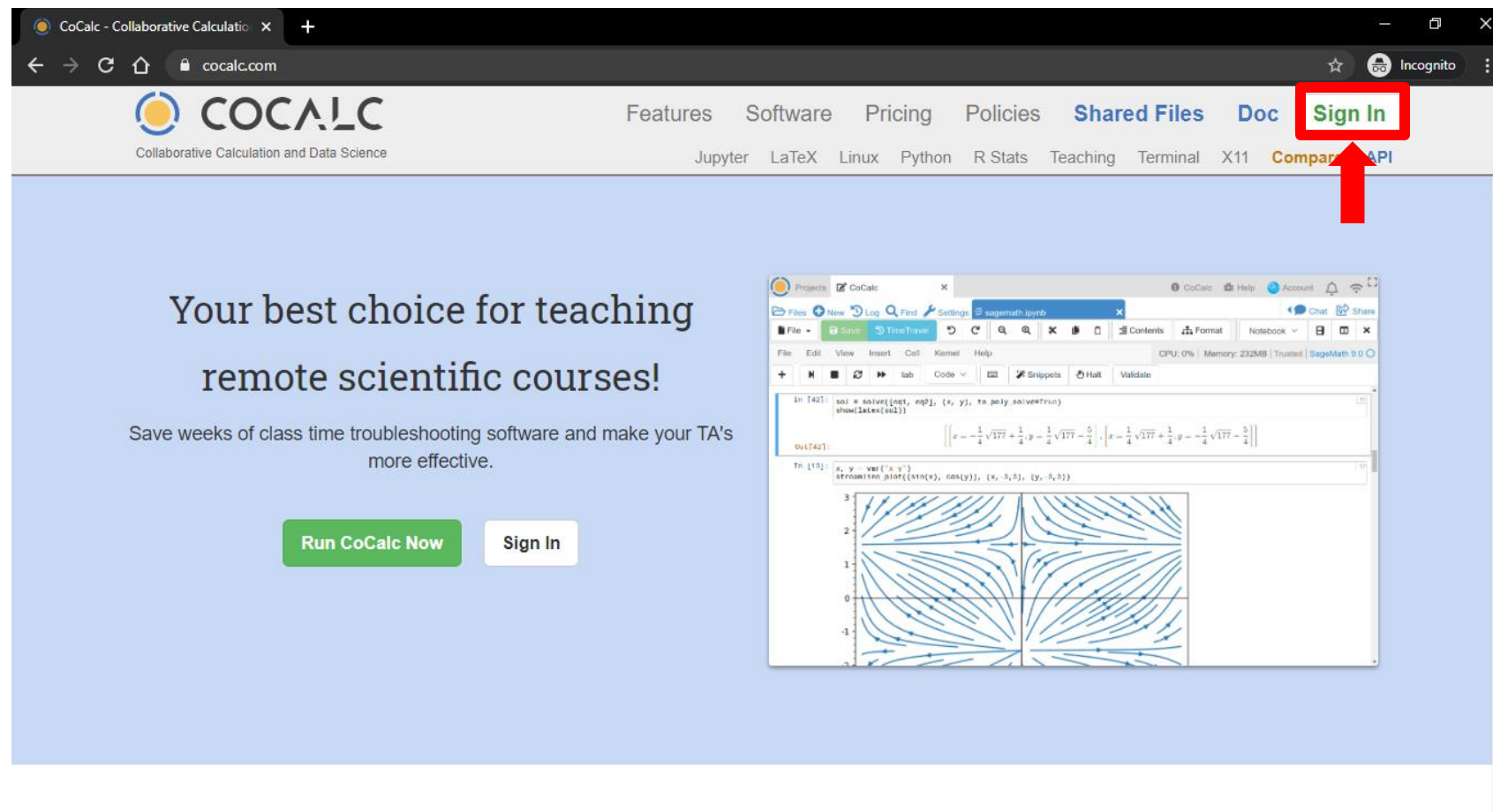
# Sumário

---

1. Ferramentas
2. Estruturas de Condição e Operadores Booleanos
3. Sequências
4. Estruturas de repetição

# Ferramenta

## 1. Clicar em Sign In



# Ferramenta

1. Aceitar os termos de condição
2. Conectar utilizando uma das opções
3. Clicar em Sign Up!

CoCalc

cocalc.com/app

Incognito (2)

Sign in CoCalc Help

Collaborative Calculation and Data Science

Create an Account

☒ I agree to the [Terms of Service](#). I also agree to receive occasional emails from CoCalc related to support.

Connect with

[f](#) [GitHub](#) [G](#) [Twitter](#)

Or sign up via email

First name

Last name

Email address

Choose a password

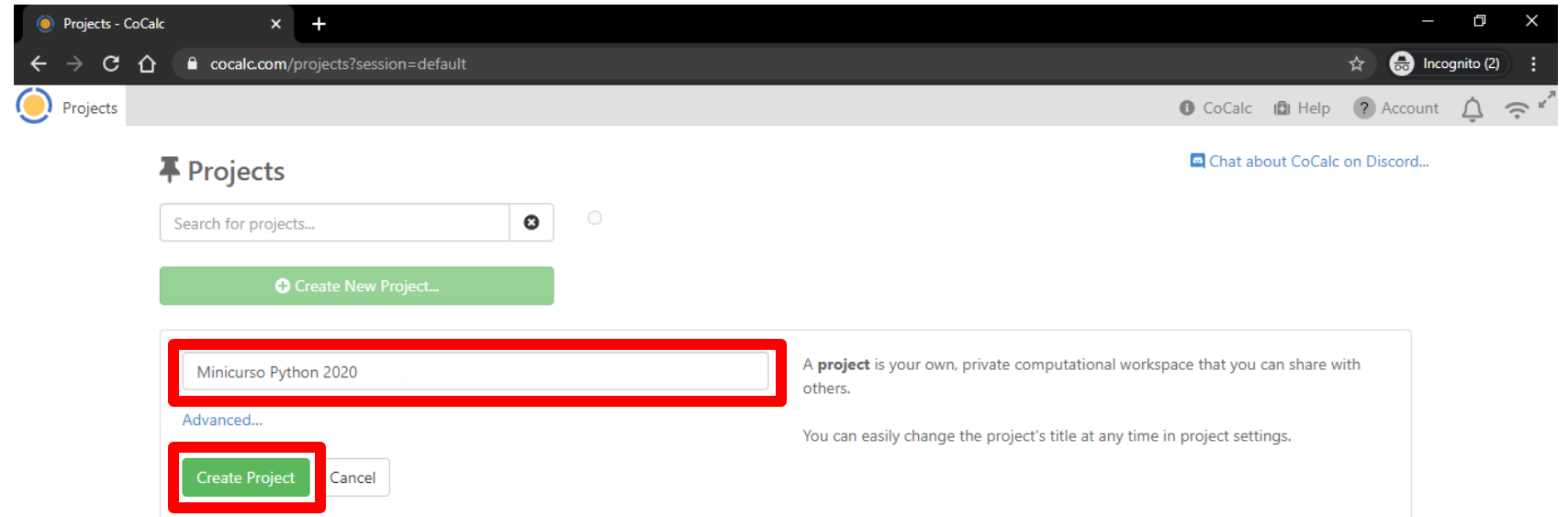
Sign Up!

Create a new account to the left or sign in with an existing account above.  
Questions? Create a [support ticket](#).

[Learn more about CoCalc...](#)

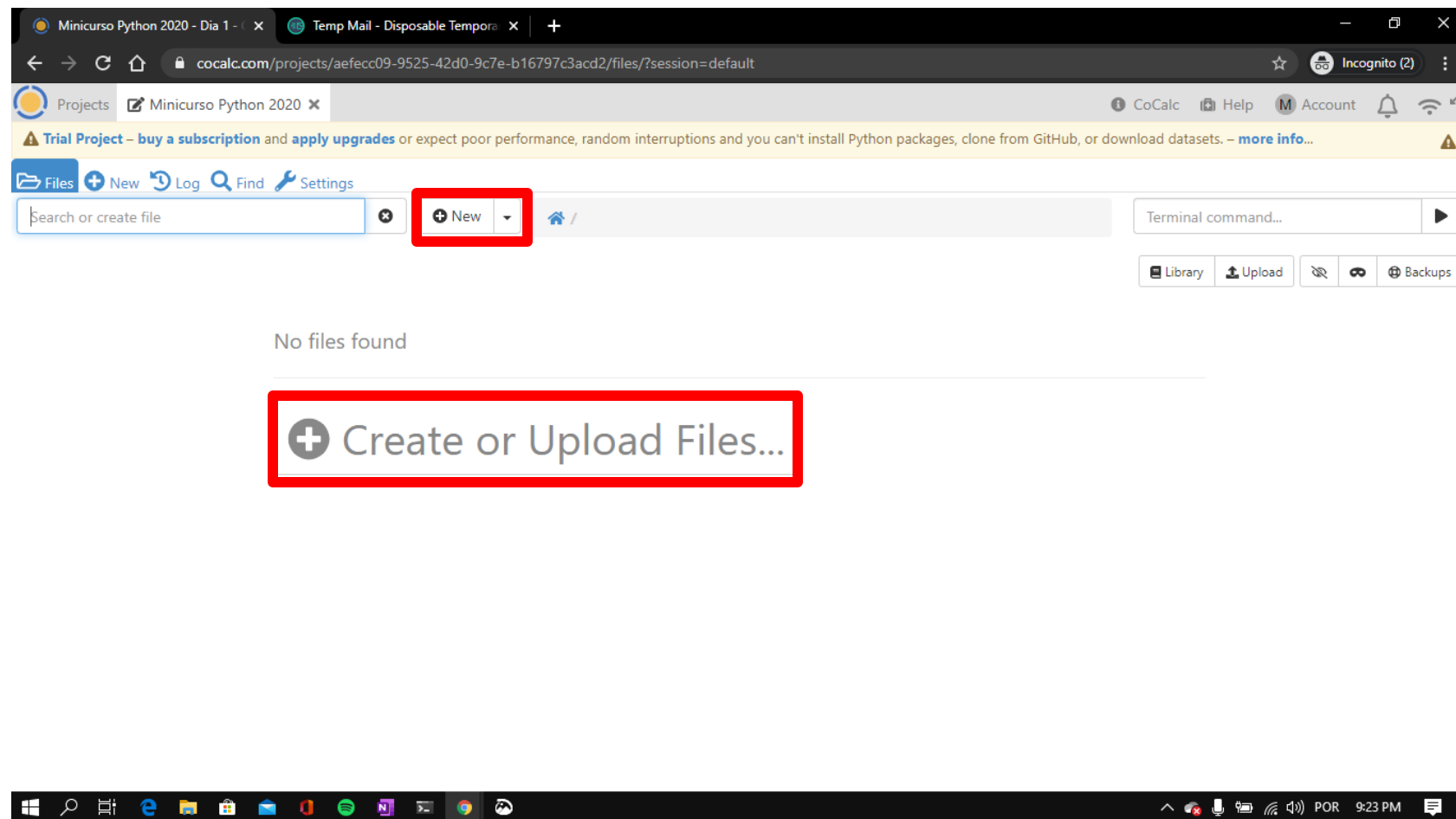
# Ferramenta

1. Inserir um nome para o projeto, que conterà os scripts do Dia 1 do minicurso.
2. Clicar em "Create Project"



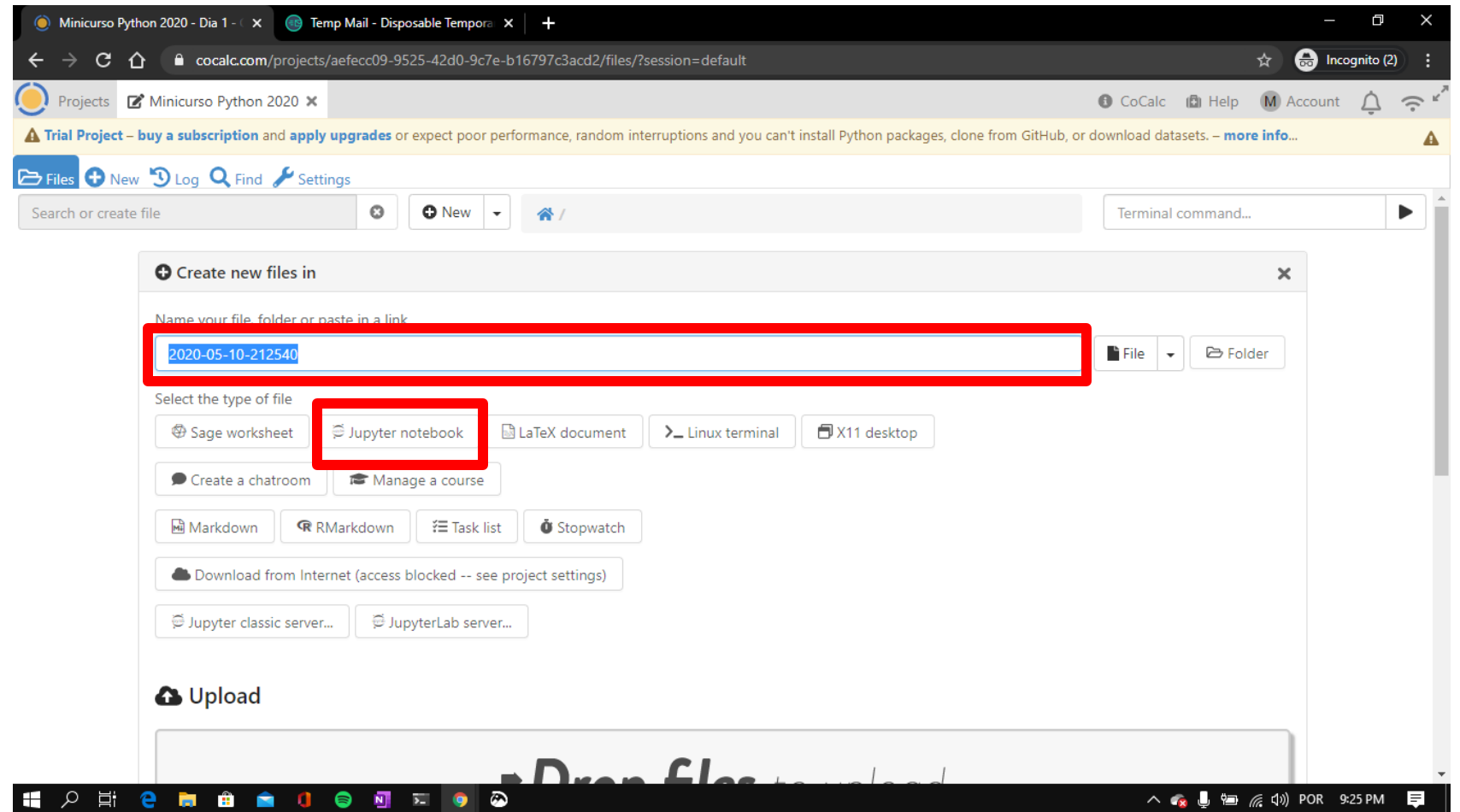
# Ferramenta

1. Clicar em "Create or Upload Files ..." ou em "New"



# Ferramenta

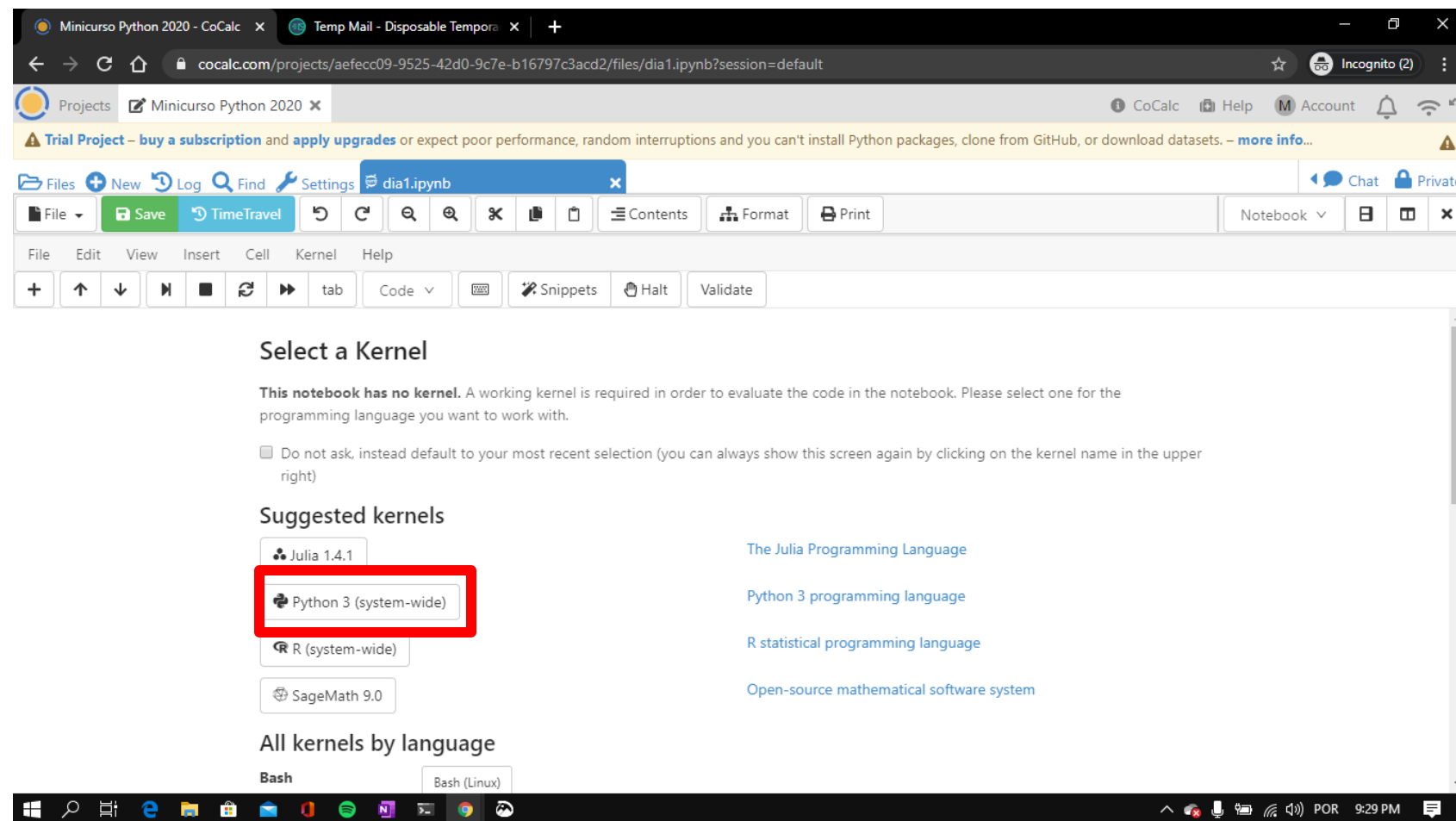
1. Inserir um nome para o script
2. E selecionar o tipo *"Jupyter Notebook"*.





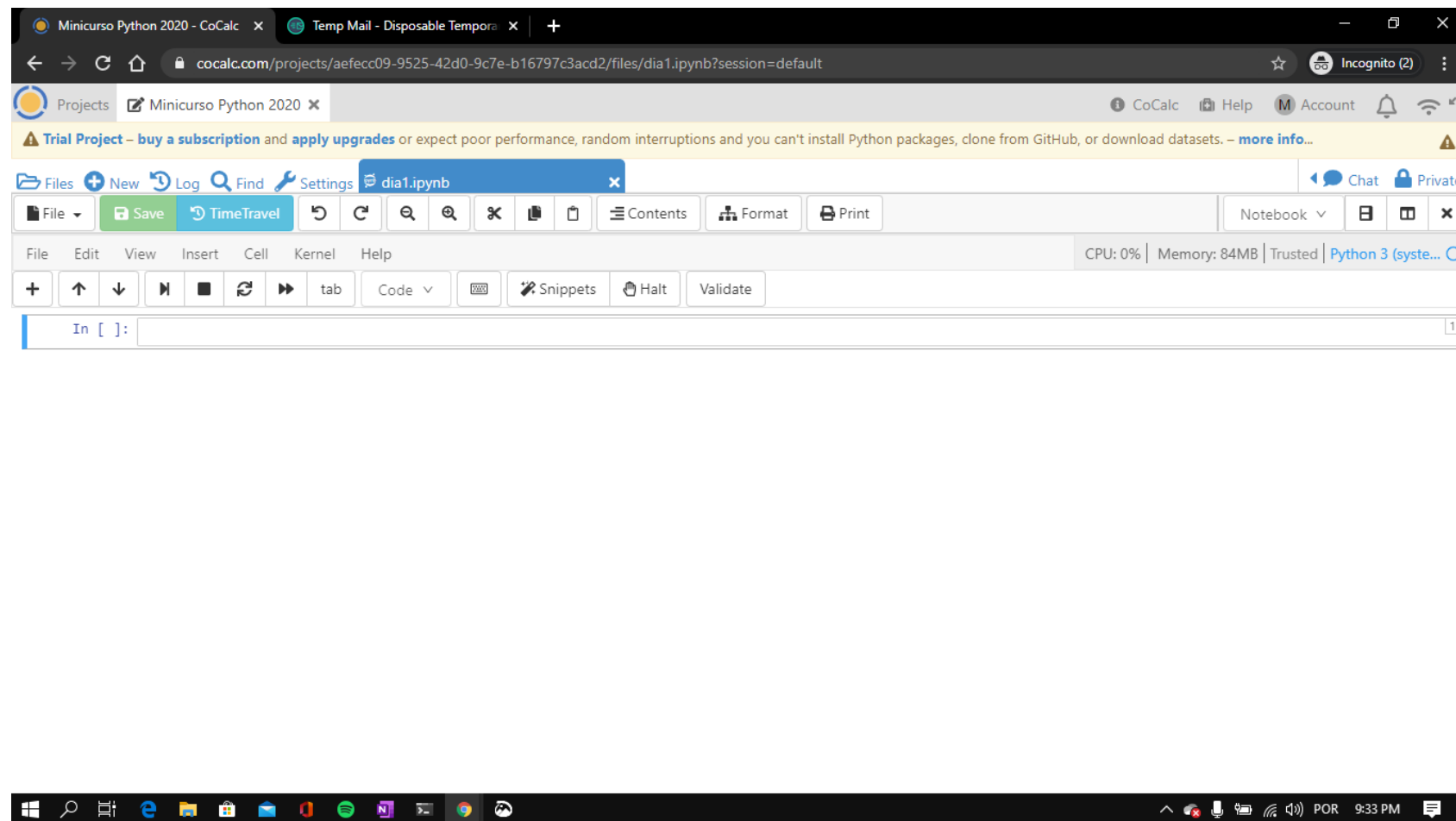
# Ferramenta

1. E, por fim, selecionar o tipo "Python 3" como o kernel deste *Jupyter notebook*.



# Ferramenta

1. Ao chegarem nesta janela o projeto e o script foram criados com sucesso.



# Estruturas de Condição e Operadores Booleanos

---

- Operações Booleanas(Bool Type, and, or, not)
- Operadores relacionais ( $=>$ ,  $=<$ ,  $!=$ ,  $==$ )
- *Statements* de condição(if/elif/else)

# Estruturas de Condição e Operadores Booleanos

---

- Exercício: Avaliar uma nota utilizando a métrica de letras com "A", "B", "C", "D", "E", "F" e notas de 0 à 100.
- ✓ Se a nota = 100: 'A'
- ✓ Senão se a nota está entre 99 e 91: 'B'
- ✓ Senão se a nota está entre 90 e 81: 'C'
- ✓ Senão se a nota está entre 80 e 71: 'D'
- ✓ Senão se a nota está entre 70 e 61: 'E'
- ✓ Senão se a nota está entre 60 e 0: 'F'



# Sequências

---

- Listas
- Tuplas
- Ranges
- Strings
- Biblioteca random (choice e shuffle)

# Sequências

---

lista = [

Item 0

Item 1

Item 2

...

Item N

]

Listas:

- Uma lista depois de criada pode ser modificada. (Objeto Dinâmico)
- Uma lista tem tamanho N, que em bytes, varia de acordo com o que a compõe.

# Sequências

---

tupla = (

Item 0
Item 1
Item 2
...
Item N

)

Tuplas:

- Uma tupla depois de criada **não** pode ser modificada. (Objeto Estático)
- Uma tupla tem tamanho N, que em bytes, varia de acordo com o que a compõe.

# Sequências

---

range(Inicio, Fim, Step)

Inicio
Step
Fim
Acumulador

## Ranges:

- Um range depois de criado **não** pode ser modificado. (Objeto Estático)
- Um range tem um tamanho fixo, pois não armazena de fato elemento a elemento.



# Sequências

lista = [

Item 0
Item 1
Item 2
...
Item N

]

tupla = (

Item 0
Item 1
Item 2
...
Item N

)

range(Inicio, Fim, Step)

Inicio
Step
Fim
Acumulador

Para  $N = 1.000.000$ , com valores igualmente espaçados, utilizando somente inteiros de 8 bytes:

- Uma lista tem 8.6 megabytes (MB).
- Uma tupla tem 7.6 megabytes (MB).
- Um range tem 48 bytes.

# Sequências

---

## Exercício 2: Manipulação de uma lista

1. Criar uma lista com 200 elementos com `range(200)`
2. Misturá-la `random.shuffle(lista)`
3. Printar e salvar numa nova lista somente as componentes múltiplas de 3.  
Usar `nova_lista = lista[::3]`
4. Com a nova lista fazer uma média destes valores amostrados.  
Usar `soma_lista = sum(lista)`

# Sequências

---

## Strings

Uma string é uma sequência com somente caracteres. Ela é declarada utilizando 'x' (aspas simples) ou "x" (aspas duplas).

```
>>> word = "Word"
```

```
>>> word.lower()
```

```
word
```

```
>>> word.upper()
```

```
WORD
```

```
>>> len(word)
```

```
4
```

4

# Estruturas de repetição

---

- **for**
- while

O *for* possui um controle maior de iteração, você utiliza quando sabe a quantidade de iterações necessárias para o bloco de instruções. Por exemplo: Um vetor com tamanho definido ou um contador.

# Estruturas de repetição

---

- for
- **while**

○ *while* você utiliza quando não sabe a quantidade de iterações necessárias para o bloco de instruções. Por exemplo: Um acumulador que deve ser limitado ou uma condição (flag) modificada pelo próprio bloco de instruções.

# Estruturas de repetição

---

## Exercício 3: Simulador de Dado

Fazer um programa que simule um dado. Ele deve ser jogado 10 vezes e os valores armazenados em uma lista.

Mostrar quanto cada um dos lados foi sorteado.

# Estruturas de repetição

---

## Exercício 4: Entrada de notas

Fazer um programa que entre com notas de um aluno de forma continua até quando for inserida uma nota  $-1$ . Nisto o programa deve parar de ler as notas e retornar a média do aluno.