

INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL  
DELEGAÇÃO REGIONAL DE LISBOA E VALE DO TEJO  
CENTRO DE EMPREGO E FORMAÇÃO PROFISSIONAL DE LISBOA



Adriana de Souza Gama  
Carlo Braga

Programador -Prog09  
UFCD 10793 – Formador : João Galamba

PROGRAMAÇÃO EM PYTHON - PROJECTO 1

28 de Março de 2022



## Índice

<b>1. INTRODUÇÃO.....</b>	<b>3</b>
<b>2. OBJECTIVOS .....</b>	<b>4</b>
<b>2.1. OBJECTIVOS ESPECÍFICOS.....</b>	<b>4</b>
<b>2.2. OBJECTIVOS GERAIS.....</b>	<b>4</b>
<b>3. DESENHO E ESTRUTURA .....</b>	<b>4</b>
<b>PARTE I – EFEITOS ESPECIAIS: .....</b>	<b>4</b>
• <b>FLUXOGRAMAS – EFEITOS ESPECIAIS:.....</b>	<b>5</b>
• <b>CÓDIGO – EFEITOS ESPECIAIS: .....</b>	<b>10</b>
• <b>IMAGENS DO OUTPUT – EFEITOS ESPECIAIS: .....</b>	<b>12</b>
<b>PARTE II – COORDENADAS EXCEL:.....</b>	<b>13</b>
• <b>FLUXOGRAMA – PROGRAMA CONVERSOR DE COORDENADAS EXCEL: .....</b>	<b>13</b>
• <b>CÓDIGO – PROGRAMA CONVERSOR DE COORDENADAS EXCEL: .....</b>	<b>14</b>
• <b>IMAGENS DO OUTPUT – PROGRAMA CONVERSOR DE COORDENADAS EXCEL:.....</b>	<b>15</b>
<b>4. IMPLEMENTAÇÃO .....</b>	<b>16</b>
<b>FERRAMENTAS DE DESENVOLVIMENTO E VERSÕES: .....</b>	<b>16</b>
<b>INVESTIGAÇÃO DE COMO PRODUZIR UM EXECUTÁVEL: .....</b>	<b>17</b>
<b>COMPILAÇÃO E INSTALAÇÃO:.....</b>	<b>17</b>
<b>DOWNLOAD DO EXECUTÁVEL: .....</b>	<b>19</b>
<b>5. CONCLUSÃO.....</b>	<b>20</b>



## 1. Introdução

Sendo um projecto com um objectivo definido à partida, em que a parte criativa estava já delineada no enunciado, por muitas voltas que se dê às palavras, trata-se de expor aqui, para avaliação, o nosso conhecimento teórico e prático da matéria ministrada pelo formador João Galamba.

Para chegar a esse objectivo, teremos que demonstrar a capacidade para implementar dois programas em Python. Duas implementações de natureza diferente, uma algorítmica de diversão e uma de natureza utilitária. Ainda a possibilidade de introduzir alguns elementos extra para consideração.



## 2. Objectivos

### 2.1. OBJECTIVOS ESPECÍFICOS

O objectivo específico deste projecto é demonstrar o nosso nível de absorção do conhecimento sobre a linguagem Python prestrado pelo formador João Galamba, ao longo das últimas semanas.

Com regras muito bem definidas sobre o resultado pretendido, estava também deliniado o objectivo de implementar um executável de coordenadas .xls, assim como a implementação de um algoritmo de encriptação e a biblioteca docopt.

### 2.2. OBJECTIVOS GERAIS

Verificar a nossa capacidade de invocar comandos, definir funções, e usar determinados argumentos para inspeção de variáveis para a implementação de um programa de efeitos num qualquer texto de duas palavras introduzido na linha de comandos, com um comando de invocação específico. Pontos extra para a implementação de temporizadores de execução;

Verificar o nível de conhecimento para usar bibliotecas de dados exteriores, na produção dum executável que converta esses mesmos dados (descrição em capítulo posterior deste relatório);

E ainda, determinar até que ponto temos a capacidade de implementar a Cifra de César para encriptar/descriptar um ficheiro.

## 3. Desenho e Estrutura

Este item é dedicado para a visualização de fluxogramas, imagens de testes e código dos scripts de acordo com as partes I, II e III deste trabalho:

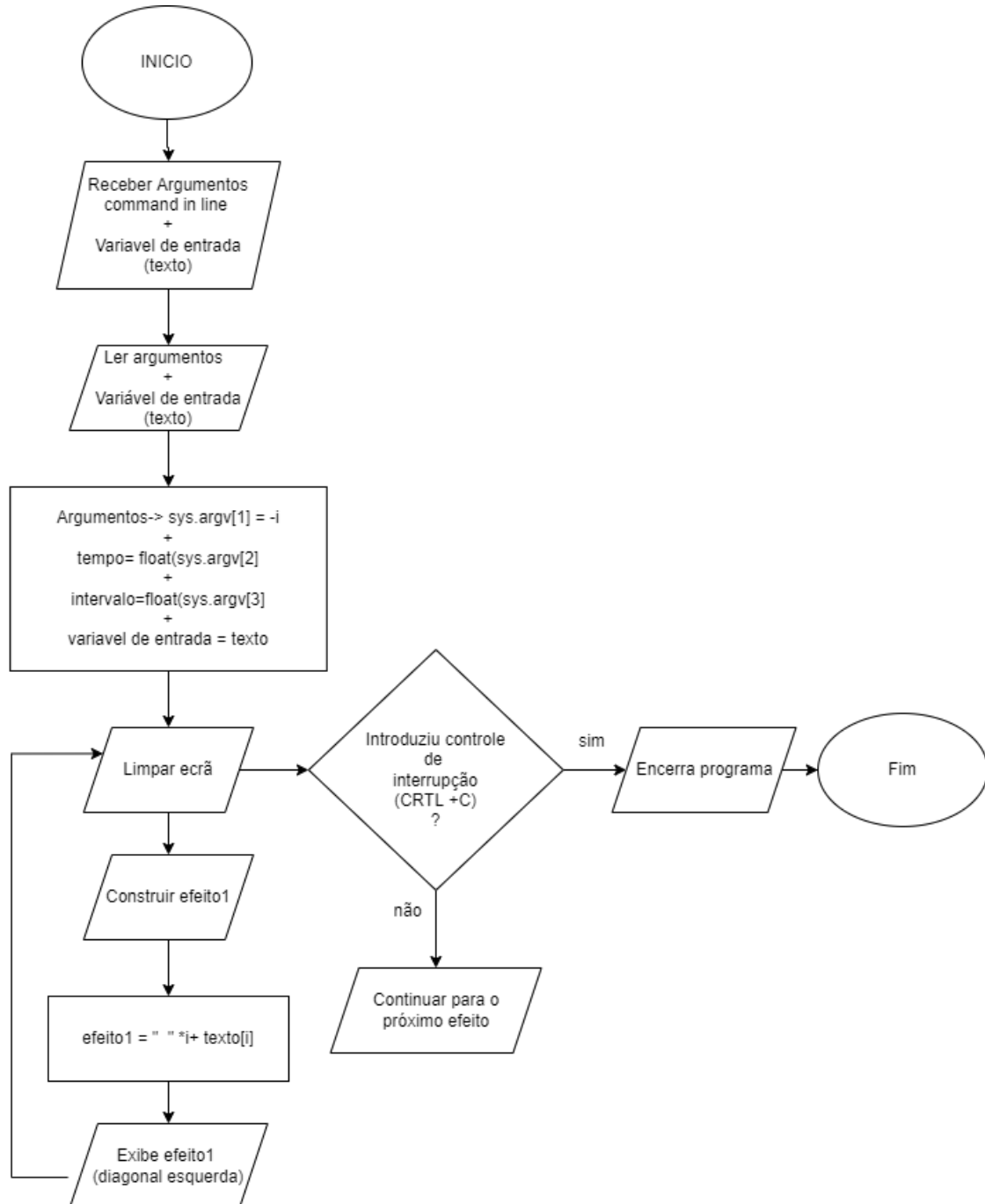
### PARTE I – EFEITOS ESPECIAIS:

Na parte I deste projecto, era proposto o desenvolvimento de um script em Python para exibir um texto introduzido na linha de comandos de acordo com os argumentos sugeridos no enunciado, onde o output deveriam ser exibidos individualmente, dado um temporizador entre os efeitos.



- **FLUXOGRAMAS – EFEITOS ESPECIAIS:**

**1. Efeito 1 – Diagonal Esquerda:**



*Figura 1- Fluxograma efeito1*



Projecto 1 - Programação em Python – UFCD 10793 - PROG09

2. Efeito 2 – Diagonal Esquerda:

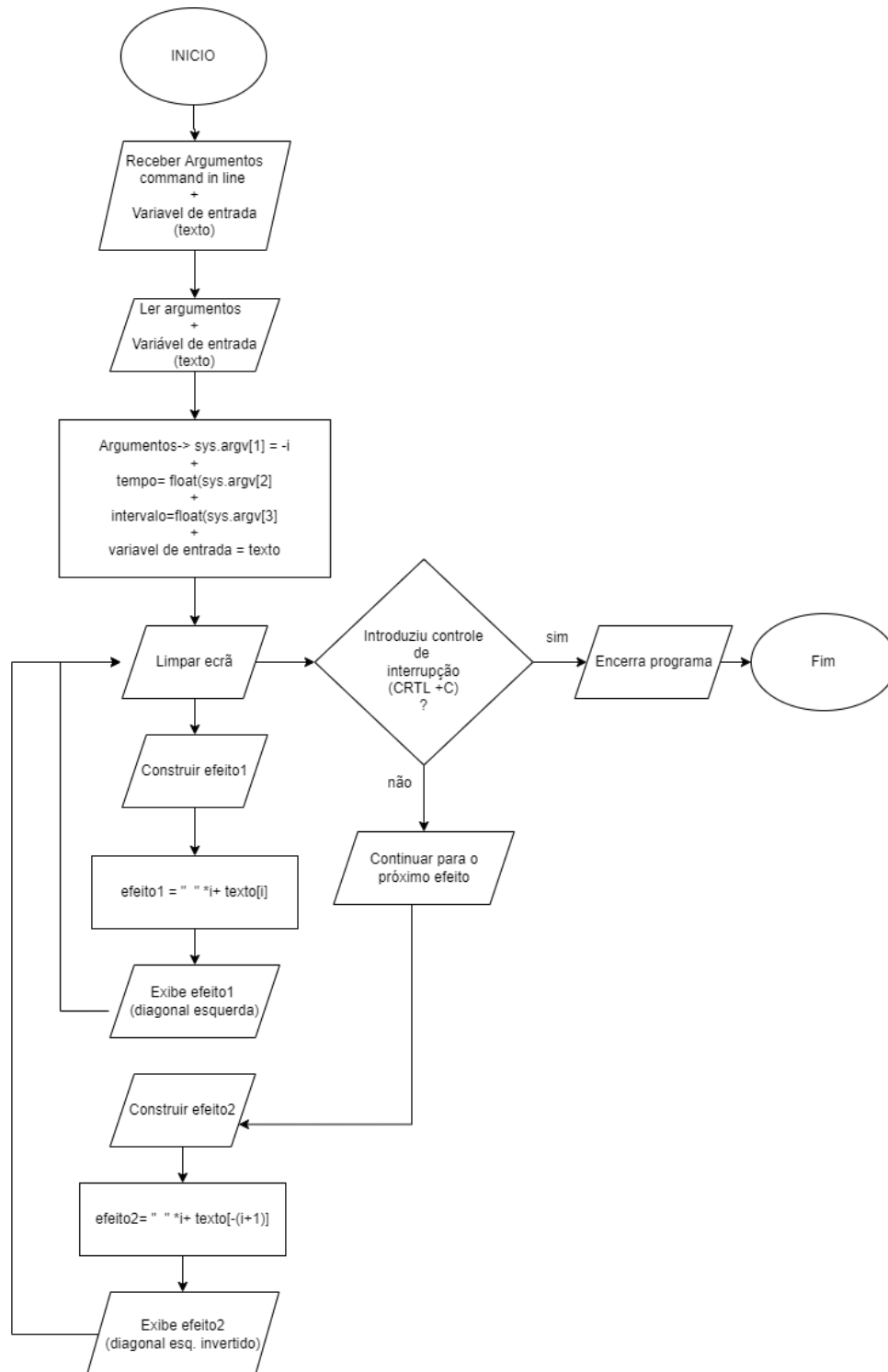


Figura 2-Fluxograma efeito2



### 3. Efeito 3 – Diagonais Cruzadas:

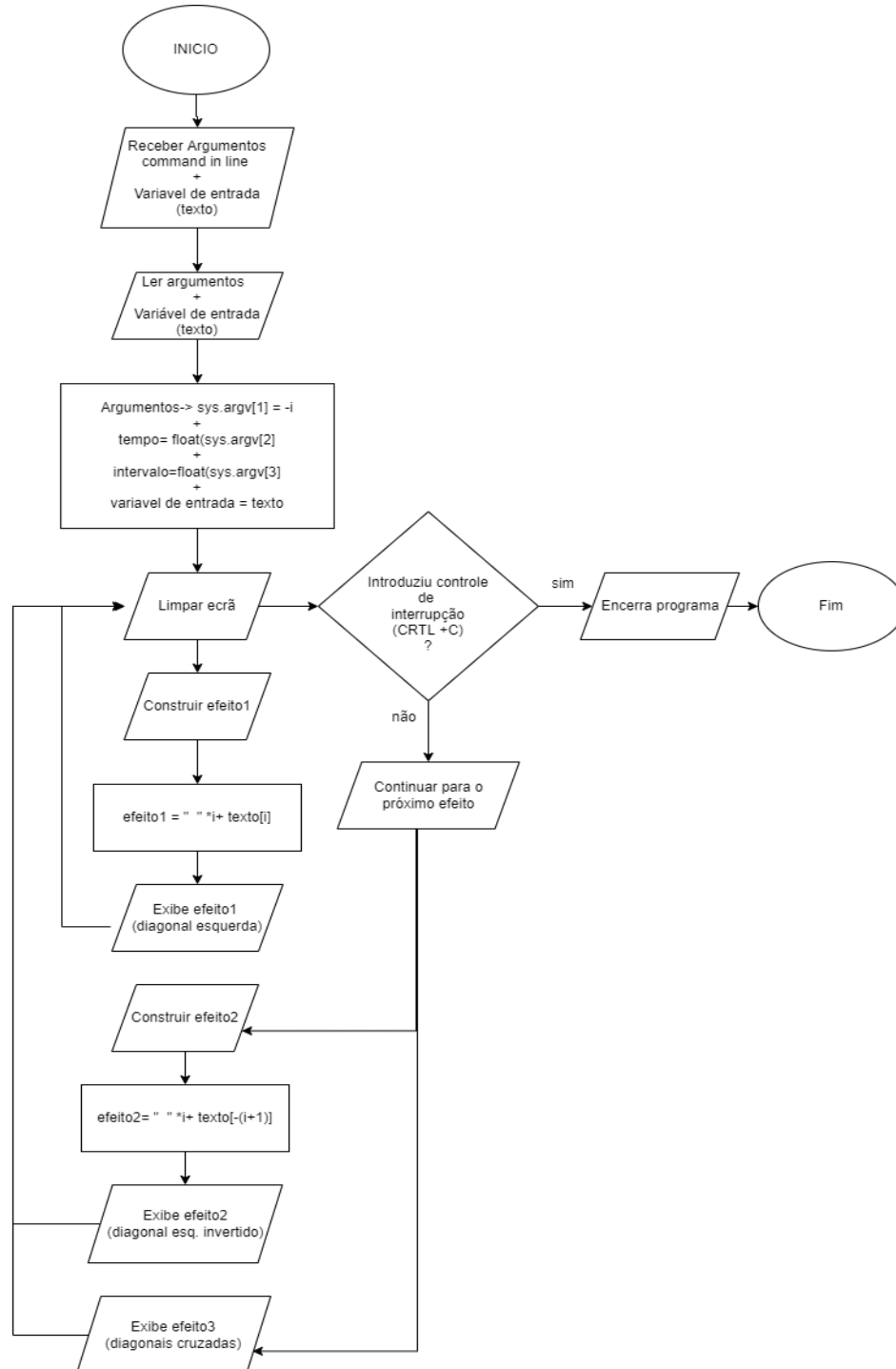


Figura 3- Fluxograma efeito3



Projecto 1 - Programação em Python – UFCD 10793 - PROG09

4. Efeito 4 – Diagonal direita, ordem inversa:

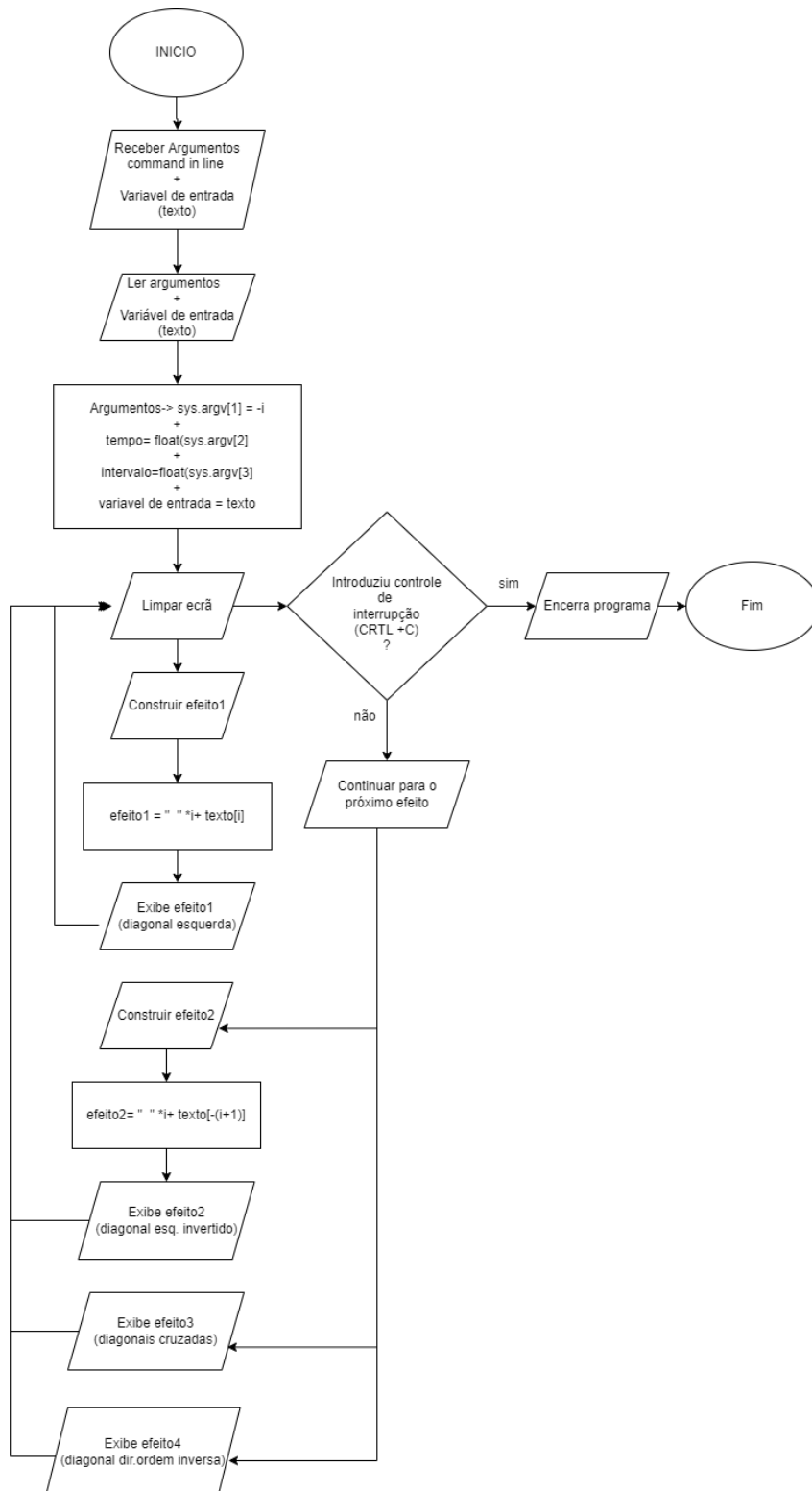


Figura 4- Fluxograma Efeito 4





Projecto 1 - Programação em Python – UFCD 10793 - PROG09

5. Efeito 5 – Texto deslizando em ciclos:

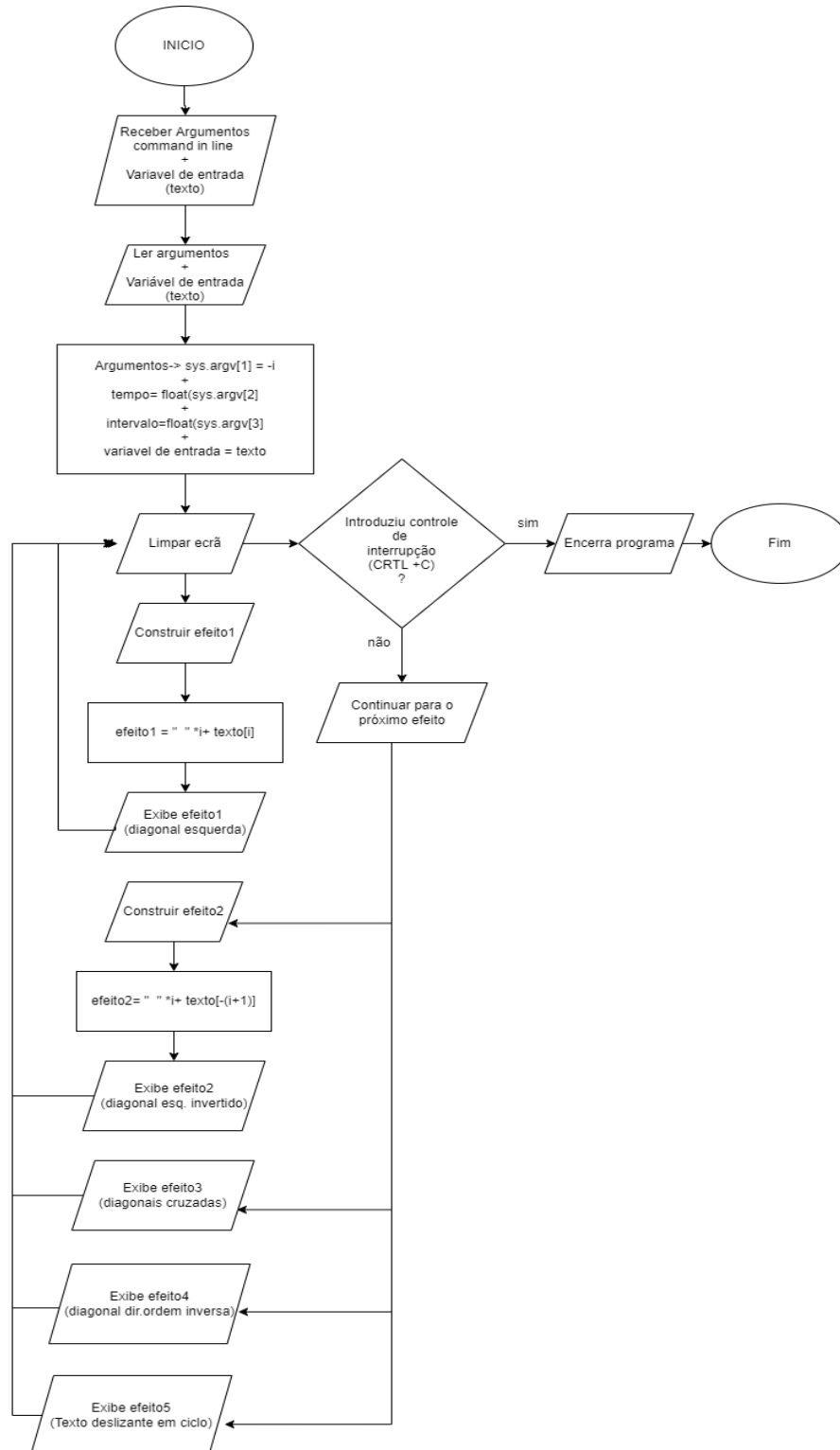


Figura 5-Fluxograma efeito5



- **CÓDIGO – EFEITOS ESPECIAIS:**

```
• #Libraries a importar
• import time
• import sys
• import os
•
• #Primeiro o programa recebe os argumentos da linha de comandos para depois
  ser juntado numa única variável string
• if sys.argv[1] == "-i":
•     try:
•         tempo = float(sys.argv[2])
•         intervalo = float(sys.argv[3])#Tempo de mudança de um Efeito para
  o outro
•     except:
•         pass
•
• texto = sys.argv[4] + " " + sys.argv[5]
•
• #Limpamos a janela de comandos
• os.system("cls")
•
• #Efeito1-Diagonal Esquerda
• for i in range(len(texto)):
•     print(" " * i + texto[i])
•
• time.sleep(intervalo)
• os.system("cls")
•
• #Efeito2-Diagonal Esquerda -->Invertida
• for i in range(len(texto)):
•     print(" " * i + texto[-(i + 1)])
•
• time.sleep(intervalo)
• os.system("cls")
•
• #Efeito3-Diagonais Cruzadas
•
• marcador = len(texto)//2 #Para ser mais rápido, o programa calcula este va
  lor uma única vez
•
• for i in range(marcador): #Ao chegar a metade da diagonal, o programa tem
  de saber o que tem de fazer a seguir
•     print(" " * i + texto[i] + " " * (len(texto) - 2*i) + texto[i])
```



---

Projecto 1 - Programação em Python – UFCD 10793 - PROG09

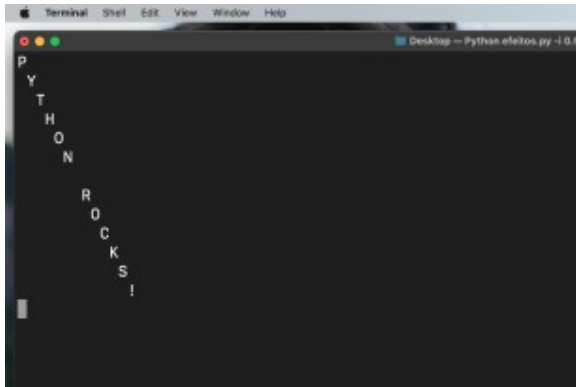
```
• if len(texto) % 2 == 0: #Se o numero de caracteres for par, o programa far
  á a cruz de uma forma
•     print(" " * marcador + texto[marcador] * 2)
• else: #Se for impar, o programa irá escrever apenas uma vez o caracter do
  meio
•     print(" " * marcador + texto[marcador])
•
• #Agora a cruz continua no sentido inverso
•
• for i in range(marcador - 1):
•     print(" " * (marcador - 1 -
      i) + texto[marcador + 1 + i] + " " * 2*(i + 1) + texto[marcador + 1 + i])
•
• time.sleep(intervalo)
• os.system("cls")
•
• #Efeito4-Diagonal Direita-->Inversa
• for i in range(len(texto)):
•     print(" " * (len(texto) - i) + texto[-(i + 1)])
•
• time.sleep(intervalo)
• os.system("cls")
•
• #Efeito5-Texto Deslizante em Ciclo
•
• novoTexto = texto + " " * (40 - len(texto))
•
• iteracao = 0 #Esta variavel ira contar quantas iterações passaram no efeito
  o 5
•
• while True:
•     os.system("cls")
•     output = ""
•
•     i = 0 #0 i é apenas um contador
•     while len(output) < 40:
•         output += novoTexto[-iteracao + i]
•         i += 1
•     iteracao += 1
•     print(output)
•     if iteracao == 41:
•         iteracao = 0
•     time.sleep(tempo)
```



Projecto 1 - Programação em Python – UFCD 10793 - PROG09

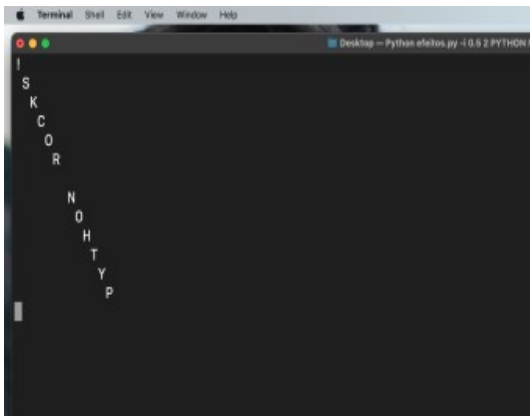
- **IMAGENS DO OUTPUT – EFEITOS ESPECIAIS:**

**1. Efeito 1 – Diagonal Esquerda:**



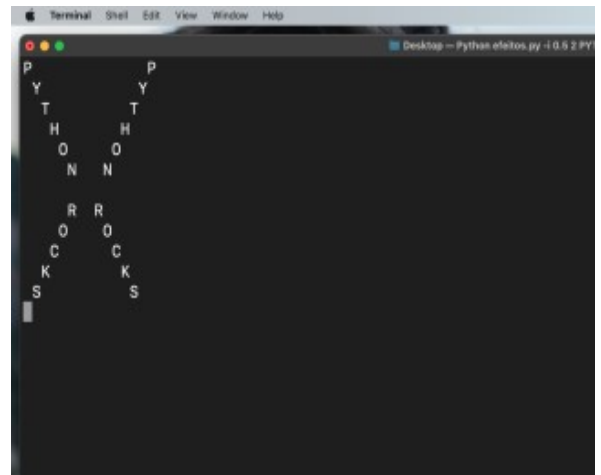
*Figura 6-Output Efeito1 Diagonal Esquerda*

**2. Efeito 2 – Diagonal Esquerda, Palavras Invertidas:**



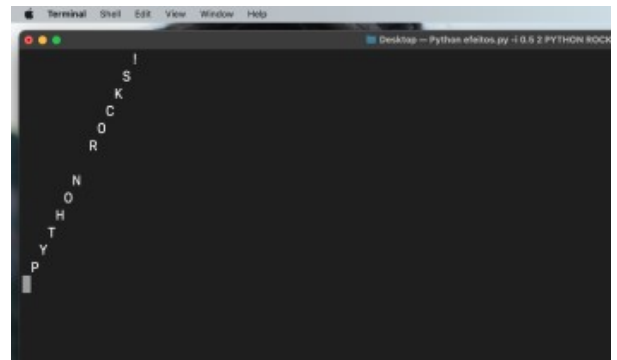
*Figura 7-Output Efeito2 Diagonal Esquerda Invertida*

**3. Efeito 3 – Diagonais Cruzadas:**



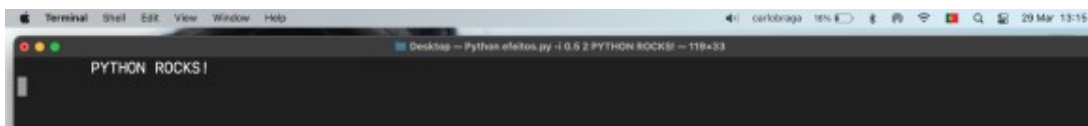
*Figura 8- Efeito 3 Diagonais Cruzadas*

**4. Efeito 4 – Diagonal Direita, Palavras por Ordem Inversa:**



*Figura 9- Efeito 4- Diagonal Direita, Ordem inversa*

**5. Efeito 5 – Efeito Deslizante em Ciclo ao longo da linha**



*Figura 10- Efeito5 Texto deslizante em Ciclo*



## PARTE II – COORDENADAS EXCEL:

Desenvolver um programa para traduzir as coordenadas "simbólicas" do Excel para coordenadas lineares. Por exemplo, em Excel, internamente, a célula A1 corresponde à célula na linha 0 e coluna 0.

- **FLUXOGRAMA – PROGRAMA CONVERSOR DE COORDENADAS EXCEL:**

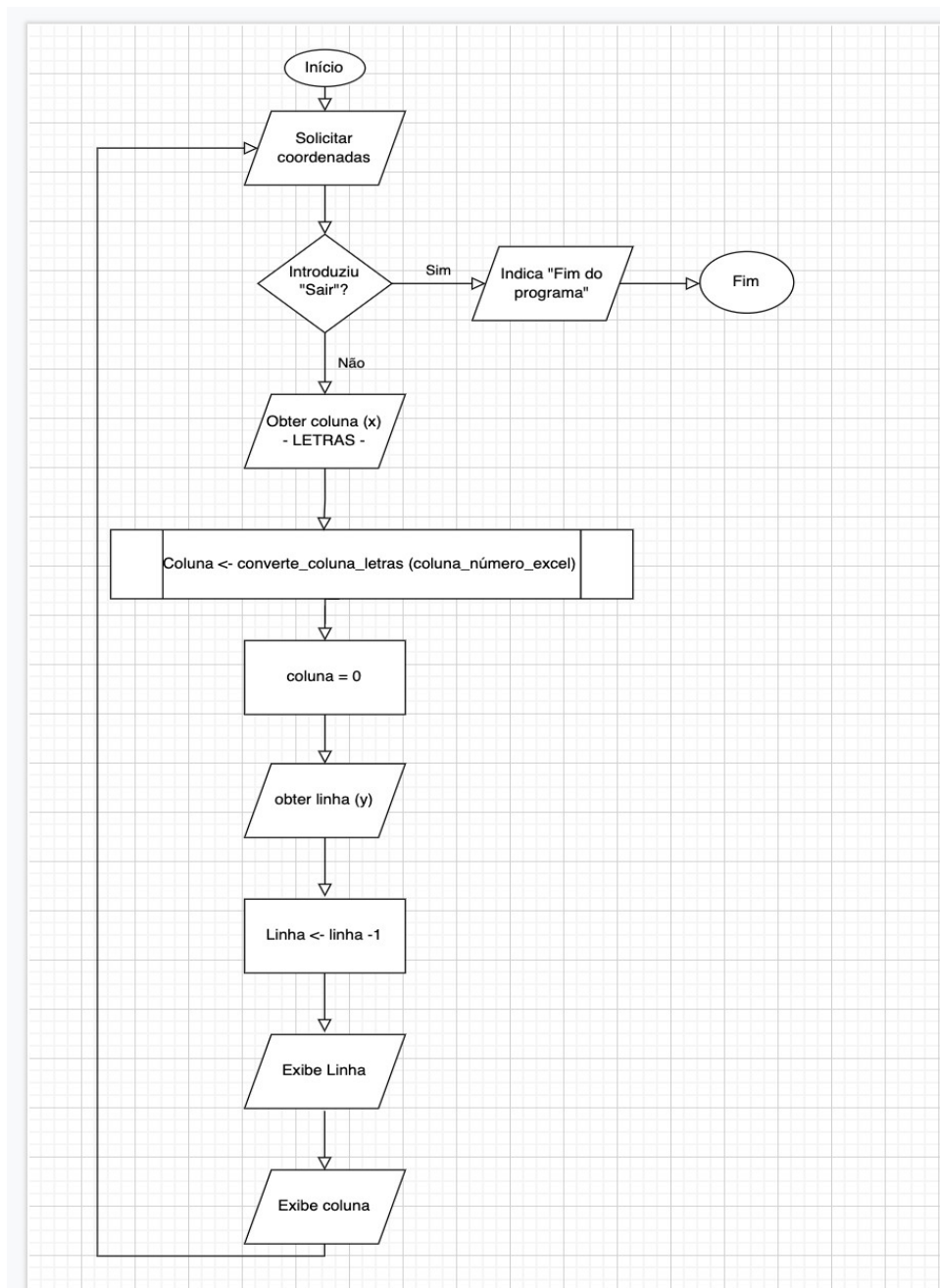


Figura 11-Fluxograma - Programa conversor de coordenadas



---

Projecto 1 - Programação em Python – UFCD 10793 - PROG09

• **CÓDIGO – PROGRAMA CONVERSOR DE COORDENADAS EXCEL:**

```
• '''
• Script referente a PARTE II do enunciado do Projecto 1 - UFCD10793
• Faça um programa para traduzir as coordenadas "simbólicas" do Excel para coord
• enadas lineares.
• Por exemplo, em Excel, internamente, a célula A1 corresponde à célula na linha
• 0 e coluna 0.
•
• Projecto realizado por:
• Adriana de Souza Gama
• Carlo Braga
• '''
• while True:
•     #Primeiro pedimos ao utilizador para nos indicar as coordenadas pretendida
• s
•     raw = input("Indique as coordenadas: ")
•
•     if raw.lower() == "sair": #Verifica se o utilizador quer continuar a usar
• o programa
•         break
•
•     dados = raw.split()
•     colunaTemp = dados[0] #Primeiro a coluna é carregada (como letras)
•     coluna = 0
•     linha = int(dados[1]) - 1 #Os números das linhas começam no 0
•
•     def converter(char): #Para facilitar o código, podemos definir uma função
• para converter letras em números
•         letras = ["a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l",
• "m", "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z"]
•         for i in range(len(letras)):
•             if char.lower() == letras[i]:
•                 return i + 1 #0 índice da letra vai ser igual à sua posição no
• alfabeto (a = 1, b = 2, c = 3, ...)
•
•         for i in range(len(colunaTemp)):
•             coluna += converter(colunaTemp[-i -
• 1]) * 26 ** i #0 número é convertido da base 26 para a base 10 segundo esta f
• órmula
•
•             coluna -= 1 #Os números das colunas começam no 0
•             print("Linha:", linha, "Coluna:", coluna, "\n-----
• ") #0 resultado é apresentado
•         print("fim do programa")
```



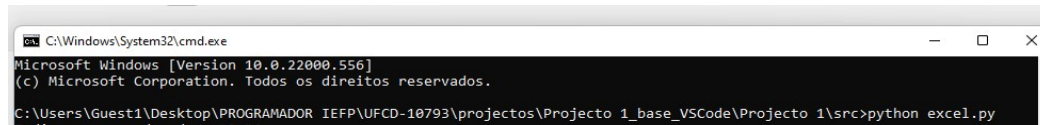
---

Projecto 1 - Programação em Python – UFCD 10793 - PROG09

- **IMAGENS DO OUTPUT – PROGRAMA CONVERSOR DE COORDENADAS EXCEL:**

### Sistema Operacional Windows:

#### Invocação do script no prompt de comandos

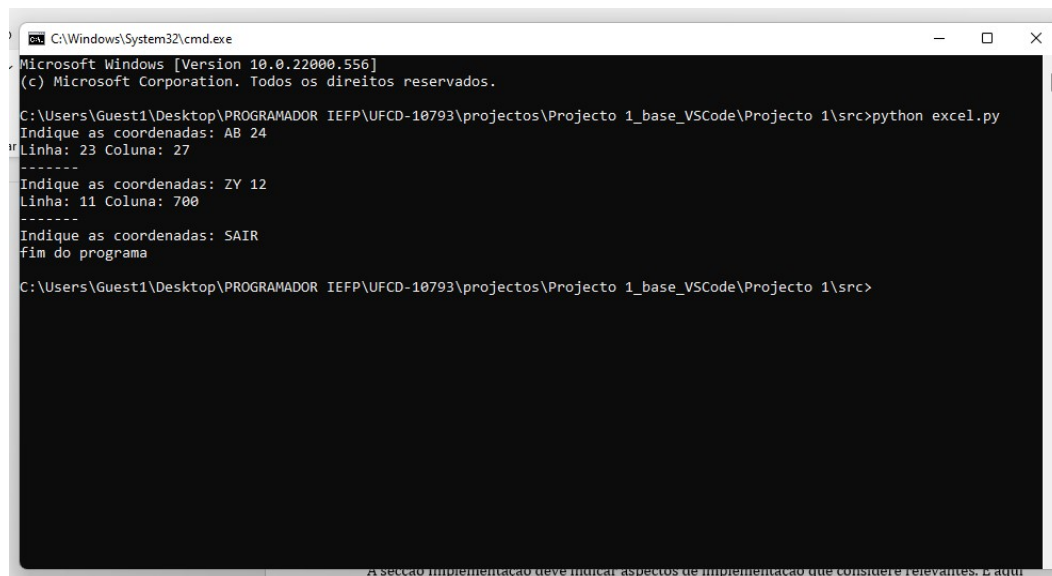


```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.22000.556]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\Guest1\Desktop\PROGRAMADOR IEF\UFCD-10793\projectos\Projecto 1_base_VSCode\Projecto 1\src>python excel.py
```

*Figura 12-Invocação de programa no prompt de comandos*

#### Output do programa



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.22000.556]
(c) Microsoft Corporation. Todos os direitos reservados.

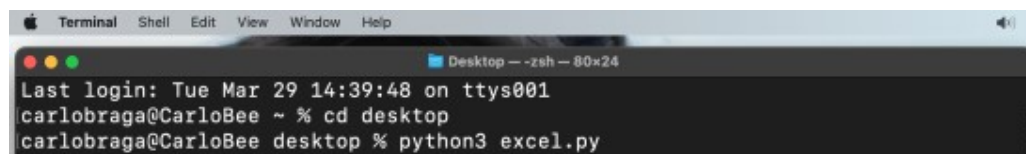
C:\Users\Guest1\Desktop\PROGRAMADOR IEF\UFCD-10793\projectos\Projecto 1_base_VSCode\Projecto 1\src>python excel.py
Indique as coordenadas: AB 24
Linha: 23 Coluna: 27
-----
Indique as coordenadas: ZY 12
Linha: 11 Coluna: 700
-----
Indique as coordenadas: SAIR
fim do programa

C:\Users\Guest1\Desktop\PROGRAMADOR IEF\UFCD-10793\projectos\Projecto 1_base_VSCode\Projecto 1\src>
```

*Figura 13 - Output Programa em Windows*

### Sistema Operacional macOS:

#### Invocação do script em command in line:



```
Terminal  Shell  Edit  View  Window  Help
Desktop -- -zsh -- 80x24

Last login: Tue Mar 29 14:39:48 on ttys001
carlobraga@CarloBee ~ % cd desktop
carlobraga@CarloBee desktop % python3 excel.py
```

*Figura 14- Invocação do programa*



Output do programa:

```
Terminal Shell Edit View Window Help
Desktop -- zsh -- 80x24
Last login: Tue Mar 29 14:39:48 on ttys001
carlobraga@CarloBee ~ % cd desktop
carlobraga@CarloBee desktop % python3 excel.py
Indique as coordenadas: BU 9
Linha: 8 Coluna: 72
-----
Indique as coordenadas: ZA 1
Linha: 0 Coluna: 676
-----
Indique as coordenadas: AB 6
Linha: 5 Coluna: 27
-----
Indique as coordenadas: sair
fim do programa
carlobraga@CarloBee desktop %
```

Figura 15-Output Programa em MacOS

## 4. Implementação

### FERRAMENTAS DE DESENVOLVIMENTO E VERSÕES:

Neste projeto, foram utilizados as seguintes ferramentas:

- **IDE** - Visual Studio Code – versão 1.57.1
- Python 3.10
- **Plataforma de desenvolvimento e teste** : Windows 11 e macOS V12.3
- **Packages**: pyinstaller versão 4.10 – Utilizado para a compilação na produção de um executável.
- PIP versão 22.0.4
- **Libraries**: Time, Sys, OS – Utilizados no desenvolvimento dos script “efeitos.py”





## Projecto 1 - Programação em Python – UFCD 10793 - PROG09

### INVESTIGAÇÃO DE COMO PRODUZIR UM EXECUTÁVEL:

Como extra deste trabalho, pretende-se investigar como se produz um executável para os programas desenvolvidos neste projecto. Abaixo, recomenda-se a instalação da biblioteca em sistema Linux ou Windows, acompanha um tutorial utilizando a biblioteca **PyInstaller**.

#### Instalação no **Linux**:

```
pip3 install PyInstaller  
pyinstaller nomedoarquivo.py -> configurar o executável;
```

#### Alternativas para **Windows**:

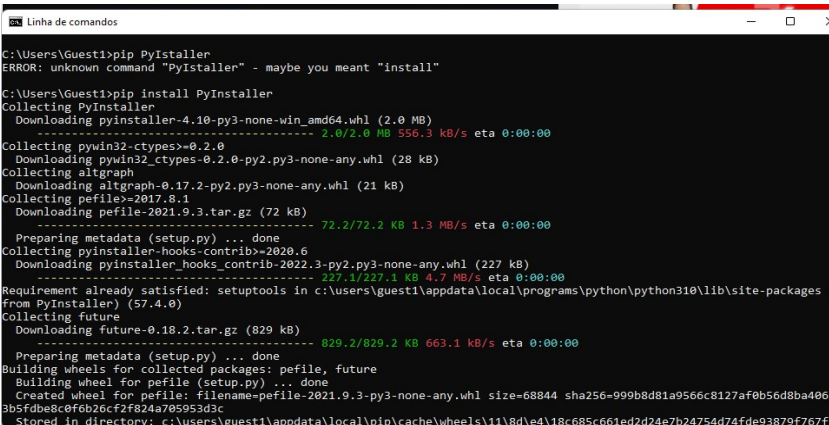
1. [PyInstaller](#);
2. [cx Freeze](#);
3. [PyOxidizer](#);
4. [Shiv](#) : Zipa o aplicativo e dependências contidas no pip da máquina em um arquivo só e usa a funcionalidade do Python zipapp. Instalação do Python necessária antes de executar.

### COMPILAÇÃO E INSTALAÇÃO:

#### Tutorial de como criar um executável para Windows usando a PyInstaller:

1. Instale o **PyInstaller**

Comando → `pip install pyinstaller`



```
C:\Users\Guest1>pip PyInstaller  
ERROR: unknown command "Pyinstaller" - maybe you meant "install"  
  
C:\Users\Guest1>pip install PyInstaller  
Collecting PyInstaller  
  Downloading pyinstaller-4.10-py3-none-win_amd64.whl (2.0 MB)  
    ----- 2.0/2.0 MB 556.3 kB/s eta 0:00:00  
Collecting pywin32-ctypes>=0.2.0  
  Downloading pywin32-ctypes-0.2.0-py2.py3-none-any.whl (28 kB)  
Collecting altgraph  
  Downloading altgraph-0.17.2-py2.py3-none-any.whl (21 kB)  
Collecting pefile>=2017.8.1  
  Downloading pefile-2021.9.3.tar.gz (72 kB)  
    ----- 72.2/72.2 KB 1.3 MB/s eta 0:00:00  
  Preparing metadata (setup.py) ... done  
Collecting pyinstaller-hooks-contrib>=2020.6  
  Downloading pyinstaller_hooks_contrib-2022.3-py2.py3-none-any.whl (227 kB)  
    ----- 227.1/227.1 KB 4.7 MB/s eta 0:00:00  
Requirement already satisfied: setuptools in c:\users\guest1\appdata\local\programs\python\python310\lib\site-packages  
  (from PyInstaller) (57.4.0)  
Collecting future  
  Downloading future-0.18.2.tar.gz (829 kB)  
    ----- 829.2/829.2 KB 663.1 kB/s eta 0:00:00  
  Preparing metadata (setup.py) ... done  
Building wheels for collected packages: pefile, future  
  Building wheel for pefile (setup.py) ... done  
  Created wheel for pefile: filename=pefile-2021.9.3-py3-none-any.whl size=68844 sha256=999b8d81a9566c8127af0b56d8ba4063b5f5dbec8c0f6b26cf2f824a70953d3c  
  Stored in directory: c:\users\guest1\appdata\local\pip\cache\wheels\11\8d\ed\18c685c661ed2d24e7b24754d74fde93879f767f
```

Figura 16 - Instalação da biblioteca PyInstaller



## Projecto 1 - Programação em Python – UFCD 10793 - PROG09

2. Execute o seguinte comando para criar um executável(caso não queira que ele crie um terminal use a opção `--noconsole`):

Comando → `pyinstaller --onefile nome-do-arquivo-python.py`

Este comando compila o programa em um único arquivo.

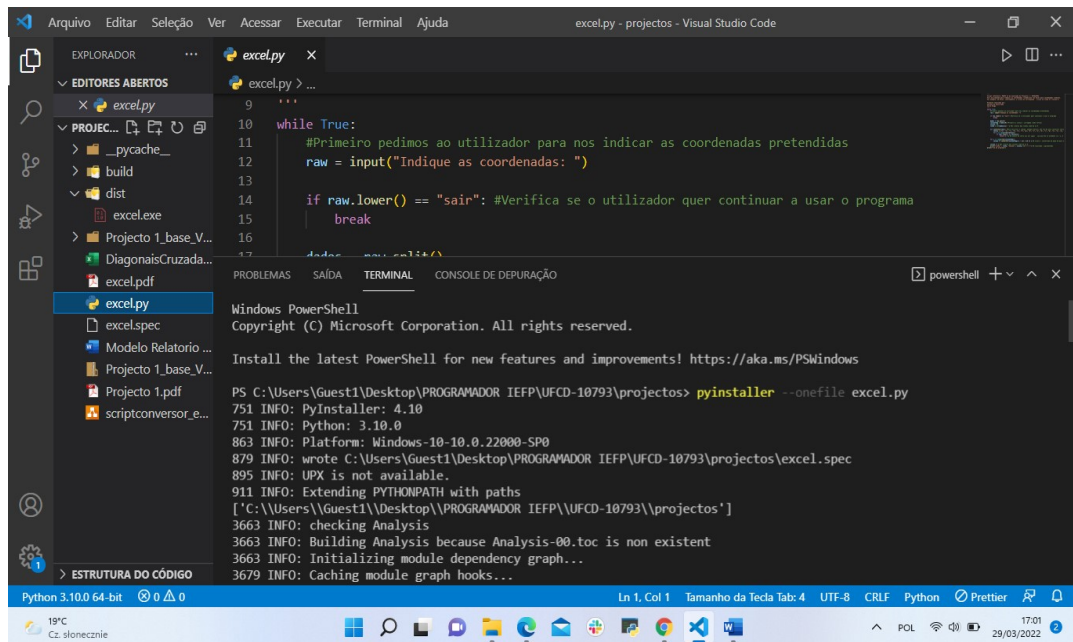


Figura 17- Compilação para executável

Após compilação, o executável encontra-se dentro da pasta **dist**:

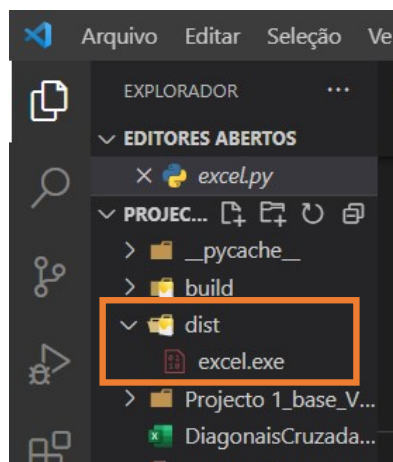
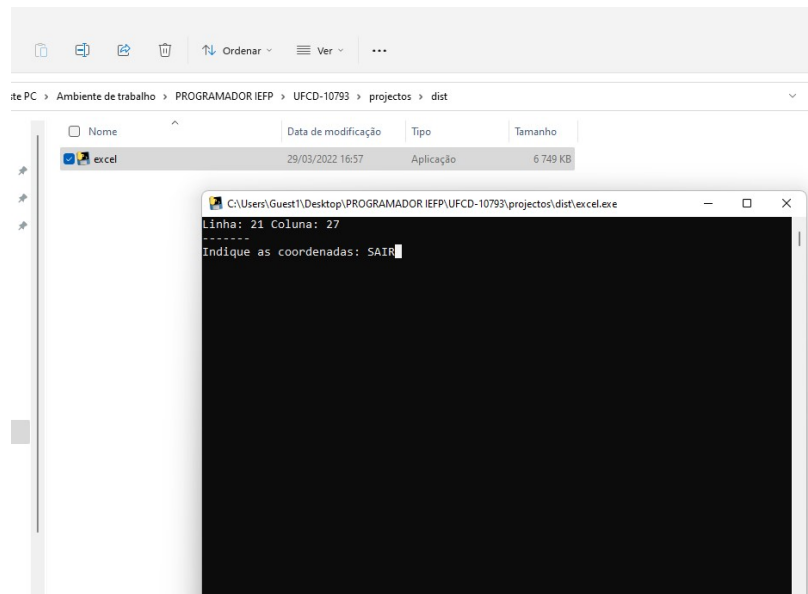


Figura 18 - Ficheiro do Executável



## Projecto 1 - Programação em Python – UFCD 10793 - PROG09



*Figura 19-Ecrã do executável*

### DOWNLOAD DO EXECUTÁVEL:

A produção deste executável, encontra disponível no link abaixo:

[https://github.com/Adriana/Projecto1-Python/tree/main/Executavel\\_conversor\\_coordenadas/dist](https://github.com/Adriana/Projecto1-Python/tree/main/Executavel_conversor_coordenadas/dist)



## 5. Conclusão

Com o desenvolvimento deste trabalho, foi possível criar e implementar um script utilizando argumentos (-i) para invocação através do command in line, especificamente para o script efeitos.py, respeitando o temporizador definindo(sleep) e o uso de comandos cls (Windows).

No script excel.py foi desenvolvido para que o programa corresse em loop, onde é solicitado ao usuário as coordenadas em letras e números para ser convertido para coluna (x) e linhas (y). O programa continua a correr até receber o comando de interrupção (CTRL+C) pelo usuário, desta forma encerra o programa. Para este mesmo script também foi compilado para a transformação de um executável, sua primeira versão encontra-se disponível em repositório GitHub como informado na sessão de compilação e instalação.

A parte III – (Cifra de César) deste trabalho não foi concluída a tempo para inclusão no referido projecto.

O desenvolvimento dos scripts tiveram como base as aulas, laboratórios fornecidos na UFCD 10793 – Programação em Python e a documentação oficial de Python 3.10.4.