



ESCOLA  
POLITÉCNICA



Programa IT Academy – Processo Seletivo – Edição #15

**Etapas 2 feito por:**

**João Pedro de Moura Medeiros**

## **Etapa 2**

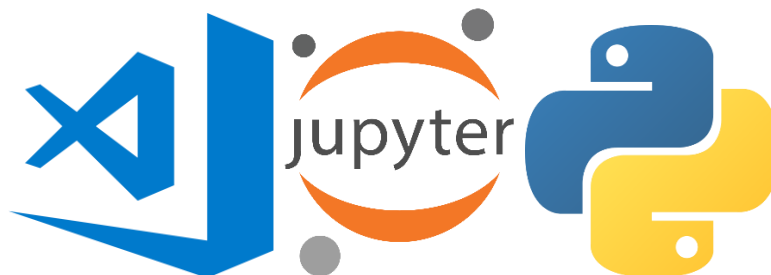
### **Enunciado**

Nesta etapa, você vai escrever um programa de computador. Para isso deve ser feita a leitura do arquivo .csv enviado junto com este enunciado. Neste arquivo você encontra dados sobre bolsas de estudo no Brasil. Você deve implementar as seguintes funcionalidades:

1. [Consultar bolsa zero/Ano] Permitir que o usuário informe o ano que deseja e como resultado o programa deverá exibir:
  - a. As informações sobre o bolsista zero, ou seja, o primeiro bolsista daquele ano (Nome, CPF, Entidade de Ensino e Valor da Bolsa);
2. [Codificar nomes] Em alguns casos o nome do aluno bolsista não deve ser exibido por questão de sigilo. Esta funcionalidade deverá codificar o nome de um bolsista. Para isso, permitir que o usuário busque um bolsista digitando todo o nome ou parte dele. Ao localizar o respectivo bolsista, seu nome deve ser codificado e exibido com as seguintes informações: Nome codificado, Ano, Entidade de ensino, Valor da Bolsa. A codificação dos caracteres deve ser deduzida a partir dos seguintes exemplos\*:  
  
PERIGO => OERIGP => PGIREO => QHJSFP  
FUGA => AUGF => FGUA => GHVB  
PAZ => ZAP => ABQ  
  
*\* Os nomes deverão ser sempre tratados apenas em letras maiúsculas. Letras acentuadas deverão ser convertidas para as respectivas letras sem os acentos. Outros sinais deverão ser descartados. Ex. Lúcia D'Ávila → LUCIA DAVILA.*
3. [Consultar média anual] Permitir que o usuário informe o ano desejado. Como resultado, o programa deverá exibir a média dos valores das bolsas daquele ano;
4. [Ranking valores de bolsa] O programa deverá listar dois tipos de colocações:
  - a. Os três alunos com os valores da bolsa mais altos;
  - b. Os três alunos com os valores da bolsa mais baixos;
5. [Terminar o programa] Permitir que o usuário saia do programa.

### **Programa**

O programa de computador foi feito no Visual Studio Code (VS Code), com a ferramenta auxiliar Jupyter-Notebook para a compilação do código na linguagem Python.



## Menu

Foi criado um menu com as opções enumeradas nos requisitos para a orientação do usuário no programa.

```
Menu

1  '''FUNÇÃO CRIADA PARA A CRIAÇÃO DO MENU E SER INSERIDA NO PROGRAMA'''
2
3  def menu():
4      print("_____")
5      print("|                                MENU                                |")
6      print("|_____|")
7      print("|          DIGITE OS NÚMERO DAS FUNCIONALIDADES ABAIXO PARA ELAS EXECUTAREM          |")
8      print("|_____|")
9      print("| Funcionalidade 1 [Consultar informações sobre o bolsista zero do ano informado] |")
10     print("| Funcionalidade 2 [Exibir informações do bolsista com nome codificado] |")
11     print("| Funcionalidade 3 [Consultar média anual dos valores das bolsas] |")
12     print("| Funcionalidade 4 [Ranking dos menores e maiores valores de bolsa] |")
13     print("| Funcionalidade 5 [Terminar programa] |")
14     print("|_____|")
15     print("|_____|")

✓ 0.8s
```

## Testes de interação ao programa:

- Teste 1: Executando o menu

```
17  #TESTE
18  menu()

✓ 0.1s

|                                MENU                                |
|_____|
|          DIGITE OS NÚMERO DAS FUNCIONALIDADES ABAIXO PARA ELAS EXECUTAREM          |
|_____|
| Funcionalidade 1 [Consultar informações sobre o bolsista zero do ano informado] |
| Funcionalidade 2 [Exibir informações do bolsista com nome codificado] |
| Funcionalidade 3 [Consultar média anual dos valores das bolsas] |
| Funcionalidade 4 [Ranking dos menores e maiores valores de bolsa] |
| Funcionalidade 5 [Terminar programa] |
|_____|
|_____|
```

## Código para a interação do usuário no programa

Para o usuário interagir com o programa, foi implementado no código uma função `input()`, para poder digitar o número indicado de determinada funcionalidade, junto a uma estrutura de repetição `while()`, para o programa ficar rodando até que o usuário deseje encerrar ele, através da funcionalidade de terminar o programa.

Dentro da estrutura de repetição há as funções if() e elif(), se o número indicado da funcionalidade for igual ao número dentro das funções, a funcionalidade se executará, caso contrário exibirá uma mensagem de erro.

Foi implementado também as funções try() e except(), para se o usuário informar outro número sem ser os das funcionalidades, mostrar uma mensagem de erro ou não.

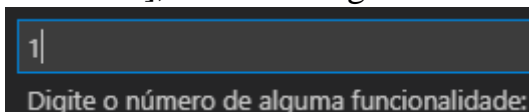
## Programa

```
1  #Aqui será o código do programa
2
3  #Chama a função menu
4  menu()
5
6  i = int
7  while i: #Foi criado um while() para ficar rodando o programa
8      try: #Inserido um try e um except para analisarem se o usuário inseriu algum dado inválido
9          i = int(input("Digite o número de alguma funcionalidade:"))
10         if(i == 1):
11             print("Funcionalidade 1 [Consultar bolsa zero/Ano]")
12             fun1() #Executará o código da funcionalidade
13         elif(i == 2):
14             print("Funcionalidade 2 [Codificar nomes]")
15             fun2() #Executará o código da funcionalidade
16         elif(i == 3):
17             print("Funcionalidade 3 [Consultar média anual]")
18             fun3() #Executará o código da funcionalidade
19         elif(i == 4):
20             print("Funcionalidade 4 [Ranking valores de bolsa]")
21             fun4() #Executará o código da funcionalidade
22         elif(i == 5):
23             print("Funcionalidade 5 [Terminar programa]")
24             print("O programa se encerrou")
25             break
26     else:
27         print("Erro de numeração, digite apenas os números\ndas funcionalidade de |1-5|")
28 except ValueError:
29     print("Digite apenas números de 1-5!!!")
```

✓ 6.9s

## Testes de interação com o programa:

- Teste 1: O usuário quer acessar a funcionalidade 1 [Consultar bolsa zero/Ano], então ele digita 1 e executa a funcionalidade.



1|  
Digite o número de alguma funcionalidade:

```

|                                     MENU                                     |
|                                                                           |
|      DIGITE OS NÚMERO DAS FUNCIONALIDADES ABAIXO PARA ELAS EXECUTAREM      |
|                                                                           |
| Funcionalidade 1 [Consultar informações sobre o bolsista zero do ano informado] |
| Funcionalidade 2 [Exibir informações do bolsista com nome codificado]         |
| Funcionalidade 3 [Consultar média anual dos valores das bolsas]              |
| Funcionalidade 4 [Ranking dos menores e maiores valores de bolsa]            |
| Funcionalidade 5 [Terminar programa]                                         |
|                                                                           |
|-----|
Funcionalidade 1 [Consultar bolsa zero/Ano]

```

NM_BOLSISTA	CPF_BOLSISTA	NM_ENTIDADE_ENSINO	ME_REFERENCIA	VL_BOLSISTA_PAGAMENTO
16263	JULIANE REIS ESCOBAR	***.781.156-**	UNIVERSIDADE FEDERAL DE JUIZ DE FORA	1 765

- Teste 2: O usuário quer acessar a funcionalidade 3 [Consultar média anual], então ele digita 3 e executa a funcionalidade.

```

3|
|
| Digite o número de alguma funcionalidade:
|
|-----|
|                                     MENU                                     |
|                                                                           |
|      DIGITE OS NÚMERO DAS FUNCIONALIDADES ABAIXO PARA ELAS EXECUTAREM      |
|                                                                           |
| Funcionalidade 1 [Consultar informações sobre o bolsista zero do ano informado] |
| Funcionalidade 2 [Exibir informações do bolsista com nome codificado]         |
| Funcionalidade 3 [Consultar média anual dos valores das bolsas]              |
| Funcionalidade 4 [Ranking dos menores e maiores valores de bolsa]            |
| Funcionalidade 5 [Terminar programa]                                         |
|                                                                           |
|-----|
Funcionalidade 3 [Consultar média anual]
A média dos valores das bolsas do ano de 2015 é de R$ 938

```

- Teste 3: O usuário quer sair do programa, então ele digita o número da funcionalidade 5 [Terminar o programa] e executa a funcionalidade.

```

5|
|
| Digite o número de alguma funcionalidade:
|
|-----|
|                                     MENU                                     |
|                                                                           |
|      DIGITE OS NÚMERO DAS FUNCIONALIDADES ABAIXO PARA ELAS EXECUTAREM      |
|                                                                           |
| Funcionalidade 1 [Consultar informações sobre o bolsista zero do ano informado] |
| Funcionalidade 2 [Exibir informações do bolsista com nome codificado]         |
| Funcionalidade 3 [Consultar média anual dos valores das bolsas]              |
| Funcionalidade 4 [Ranking dos menores e maiores valores de bolsa]            |
| Funcionalidade 5 [Terminar programa]                                         |
|                                                                           |
|-----|
Funcionalidade 5 [Terminar programa]
O programa se encerrou

```

- Teste 4: O usuário inseriu um número errado.

5000

Digite o número de alguma funcionalidade:

MENU

DIGITE OS NÚMERO DAS FUNCIONALIDADES ABAIXO PARA ELAS EXECUTAREM

Funcionalidade 1 [Consultar informações sobre o bolsista zero do ano informado]

Funcionalidade 2 [Exibir informações do bolsista com nome codificado]

Funcionalidade 3 [Consultar média anual dos valores das bolsas]

Funcionalidade 4 [Ranking dos menores e maiores valores de bolsa]

Funcionalidade 5 [Terminar programa]

Erro de numeração, digite apenas os números das funcionalidade de [1-5]

## Leitura do arquivo .csv

A leitura do arquivo .csv foi feita através de um método da biblioteca Pandas, do Python, para isso foi necessário importar a biblioteca Pandas.

## Importação da biblioteca Pandas:

# Importação das bibliotecas

```
1 '''AQUI SERA FEITA A IMPORTAÇÃO DAS BIBLIOTECAS'''
2
3 #Importação da biblioteca 'pandas' para ser feita a leitura, análise e manipulação de dados do arquivo .csv
4 import pandas as pd
```

✓ 0.8s

Observação: A biblioteca Pandas também ajudará na implementação de funcionalidades pedidas com a análise e manipulação dos dados do arquivo .csv.

**Leitura do arquivo .csv com o método read\_csv() do pandas:**

# Leitura do arquivo .csv

```
1 '''AQUI SERA FEITA A LEITURA DO ARQUIVO .CSV'''
2
3 #A variável 'csv' receberá a leitura de um arquivo .csv feito pelo método 'read_csv()' do pandas, com um separador de dados 'sep'
4 csv = pd.read_csv("./br-capes-bolsistas-uab.csv", sep=";")
5
```

## Execução e resultado de leitura do arquivo csv:

### Leitura do arquivo .csv

+ Código
+

```

1  '''AQUI SERA FEITA A LEITURA DO ARQUIVO .CSV'''
2
3  #A variável 'csv' receberá a leitura de um arquivo .csv feito pelo método
4  csv = pd.read_csv("./br-capes-bolsistas-uab.csv", sep=";")
5
6  #TESTE
7  display(csv) #Display printará a leitura do arquivo .csv

```

✓ 0.3s

	NM_BOLSISTA	CPF_BOLSISTA	NM_ENTIDADE_ENSINO	ME_REFERENCIA	AN_REFERENCIA	SG_DIRETORIA
0	ALEXANDRE RIBEIRO NETO	***.195.647-**	UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO	1	2016	DED
1	LUIZ GUSTAVO RIBEIRO ROLANDO	***.866.306-**	UNIVERSIDADE ESTADUAL DO NORTE FLUMINENSE DARC...	1	2016	DED
2	MARIA ELZA BOCATTI ROSSINI	***.180.719-**	UNIVERSIDADE ESTADUAL DE LONDRINA	1	2016	DED
3	ANTONIO RODRIGUES DE ARAUJO	***.060.403-**	UNIVERSIDADE FEDERAL DO PIAUI	1	2016	DED
4	ROBERTO ARLUNDO PINTO	***.352.736-**	UNIVERSIDADE FEDERAL DE OURO PRETO	1	2016	DED
...	...	...	...	...	...	...
80010	NATALIA NOGUEIRA SARAIVA	***.778.373-**	UNIVERSIDADE ESTADUAL DO CEARA	1	2013	DED
80011	FABIO DA PURIFICACAO DE BASTOS	***.603.607-**	UNIVERSIDADE FEDERAL DE SANTA MARIA	1	2013	DED
80012	GLAUCIA ANK GUARINO	***.184.977-**	UNIVERSIDADE FEDERAL DO RIO DE JANEIRO	1	2013	DED
80013	NILZETE CRUZ SILVA	***.730.875-**	UNIVERSIDADE DO ESTADO DA BAHIA	1	2013	DED
80014	ROSILENE DE LIMA	***.474.229-**	UNIVERSIDADE ESTADUAL DE MARINGA	1	2013	DED

80015 rows x 11 columns

Figura 1 Arquivo .csv

SG_SISTEMA_ORIGEM	CD_MODALIDADE_SGB	DS_MODALIDADE_PAGAMENTO	CD_MOEDA	VL_BOLSISTA_PAGAMENTO
SGB	253	TUTOR	R\$	765
SGB	253	TUTOR	R\$	765
SGB	245	COORDENADOR DE TUTORIA	R\$	1300
SGB	253	TUTOR	R\$	765
SGB	253	TUTOR	R\$	765
...	...	...	...	...
SGB	6	TUTOR	R\$	765
SGB	242	COORDENADORIA ADJUNTA	R\$	1500
SGB	7	TUTOR	R\$	765
SGB	5	PROFESSOR FORMADOR I	R\$	1300
SGB	6	TUTOR	R\$	765

Figura 2 Continuação da Figura 1

## Funcionalidades – Explicações e lógica usada

Os códigos das funcionalidades 1 a 4 estão dentro de funções criadas, para serem depois executadas na aba do programa.

```
def fun1():
```

```
if(i == 1):  
    print("Funcionalidade 1 [Consultar bolsa zero/Ano]")  
    fun1() #Executará o código da funcionalidade
```

1. [Consultar bolsa zero/Ano]

### Permitir que o usuário informe o ano que desejar:

Foi implementada a função input() para o usuário poder digitar o ano desejado. Se ele digitar o ano errado, foram implementadas as funções try() e except(), para executar a funcionalidade ou mostrar uma mensagem de erro.

```
#O ano inserido teve que ser transformada em 'int', pois não conseguiu se localizar no arquivo .csv a variável como 'Str'  
#Foi criado uma variável chamada 'ano', para receber o ano desejado, informado pelo usuário  
ano = int(input("Informe o ano desejado:"))
```

### Filtrar linhas e colunas que tenham o mesmo índice informado pelo usuário:

- a) Para pegar as informações apenas daquele ano desejado, foi adicionado o método chamado loc(), o qual extrai do Data Frame as linhas, com suas respectivas colunas, que apresentam o valor informado pelo usuário.

```
#A variável bolsistaZero recebe linhas extraídas, através da função 'loc()', com determinado ano indicado  
bolsistaZero = csv.loc[csv["AN_REFERENCIA"]==ano]
```

### Retirar informações indesejadas e printa apenas o solicitado:

Realizada a filtragem, basta apenas extrair o que é pedido no enunciado de informação, usando o método drop() para retirar as colunas desnecessárias. E para pegar a primeira pessoa daquele ano, foi incluído o método tail(), tendo o número 1 dentro da função, puxando do Data Frame apenas o primeiro bolsista.

Observação: Usou-se tail() para puxar o último bolsista do Data Frame pois acredita-se que ele é o primeiro daquele ano, porém como não há data diária, apenas mensal no arquivo .csv, não podemos confirmar que aquele bolsista é de fato o primeiro. Caso não seja, modifica-se o método tail() pelo head().



```
#É usado o método do pandas chamado 'drop()', o qual remove algo indesejado
#O método 'tail()' é usado para pegar o primeiro bolsista zero do DataFrame
Df = bolsistaZero.tail(1).drop(columns=["AN_REFERENCIA", "SG_DIRETORIA", "SG_SISTEMA_ORIGEM",
                                         "CD_MODALIDADE_SGB", "CD_MOEDA", "DS_MODALIDADE_PAGAMENTO"])
display(Df)
```

## Testes da funcionalidade 1. [Consultar bolsa zero/Ano]:

- Teste 1: O usuário executou a funcionalidade 1 [Consultar bolsa zero/Ano].

Pede-se para informar o ano, como exemplo foi digitado o ano de 2013:

Informe o ano desejado:

Printa na tela as informações:

	NM_BOLSISTA	CPF_BOLSISTA	NM_ENTIDADE_ENSINO	VL_BOLSISTA_PAGAMENTO
80014	ROSILENE DE LIMA	***.474.229-**	UNIVERSIDADE ESTADUAL DE MARINGÁ	765

- Teste 2: O usuário executou a funcionalidade 1 [Consultar bolsa zero/Ano].

Pede-se para digitar o ano, como exemplo foi digitado o ano de 2014:

Informe o ano desejado:

Printa na tela as informações:

	NM_BOLSISTA	CPF_BOLSISTA	NM_ENTIDADE_ENSINO	VL_BOLSISTA_PAGAMENTO
59676	FRANCISCO GOMES DE LOIOLA NETO	***.062.603-**	UNIVERSIDADE ESTADUAL DO CEARÁ	765

- Teste 3: O usuário executou a funcionalidade 1 [Consultar bolsa zero/Ano].

Pede-se para digitar o ano, como foi digitado o ano de 2016:

Informe o ano desejado:

Printa na tela as informações:

	NM_BOLSISTA	CPF_BOLSISTA	NM_ENTIDADE_ENSINO	VL_BOLSISTA_PAGAMENTO
16263	JULIANE REIS ESCOBAR	***.781.156-**	UNIVERSIDADE FEDERAL DE JUIZ DE FORA	765

- Teste 4: O usuário inseriu o ano errado na funcionalidade 1 [Consultar bolsa zero/Ano].

Informe o ano desejado:

✓ 5.9s

Dado de entrada inválido, por favor execute novamente a funcionalidade e insira o dado correto

## 2. [Codificar nomes]

### Permitir que o usuário informe o nome do bolsista, seguindo do tratamento das letras maiúsculas:

Foi criado um input() para receber o nome do bolsista indicado pelo usuário, junto a um upper(), os quais tratam e apresentam toda a string em letras maiúsculas como pedido no enunciado.

```
#Foi criado uma variável chamada 'nome', para receber o nome do bolsista desejado, informado pelo usuário  
#Há o método 'upper' para deixar o nome com letra maiúsculas  
nome = input("Digite o nome do bolsista:").upper()
```

### Tratamento de erros:

Foi inserido um if(), else, try() e except() para tratamento de erros no código.

```
#Apenas pode serem digitados nomes, senão aparecerá uma mensagem de erro  
if not nome.isdigit():
```

```
else:  
    print("Dado de entrada inválido, por favor execute novamente a funcionalidade e insira o dado correto")
```

```
#If criada para apenas nomes que tiverem 3 ou menos caracteres, alterando somente o caractere inicial e o final  
#Criado pois não se pode inverter o meio  
if(tamNome<=3):
```

```
    nomeJunto1 = nome[0].join(list(reversed(nome[1:]))).join(reversed(nome[0]))  
else: #Else criada para nomes que tiverem mais de 3 caracteres
```

```
#Try e except ve se o nome está certo  
try:  
    #Criado um novo DataFrame e adiciona a coluna com o nome codificado  
    novoDF = df.assign(NM_CODIFICADO=nomeJunto4)  
    display(novoDF)  
except UnboundLocalError:  
    print("Data de entrada inválido, por favor execute novamente a funcionalidade e insira o dado correto")
```

## Codificação

A codificação dos caracteres do usuário foi feita por etapas:

- 1) Primeiro se confere se o nome inserido tem 3 caracteres ou menos, pois seguindo o exemplo da “PAZ” no enunciado, não daria para mudar o seu meio, então se conferiu seu tamanho.

```
#Separa o nome do sobrenome do usuário e adiciona em uma lista para se fazer análise dos dados
nomeSep = list(nome.split(" "))

#Variável criada para saber quantos caracteres o tamanho do nome do bolsista tem
tamNome = len(nomeSep[0])

#If criada para apenas nomes que tiverem 3 ou menos caracteres, alterando somente o caractere inicial e o final
#Criado pois não se pode inverter o meio
if(tamNome<=3):

    else: #Else criada para nomes que tiverem mais de 3 caracteres
```

- 2) Após isso se executará um comando que inverterá a letra inicial pela letra final e vice-versa. Foi criada uma função para fazer essa etapa.

```
#Função criada para codificação do nome e sobrenome, seguindo os exemplos da funcionalidade 2, trocando o primeiro caracter pelo último string
def troca(x):
    return x[-1] + x[1:-1] + x[0] if len(x) > 1 else x

for i in nomeSep: #For criado para o fazer a codificação do nome, seguindo o SEGUNDO exemplo do enunciado
    listaCod2.append(troca(i)) #Adiciona-se o nome ou sobrenome na função 'troca' para serem codificados e adicionados na lista criada
nomeJunto1 = " ".join(listaCod2) #Junta o nome e o sobrenome codificados
```

OERIGP

AUGF

ZAP

- 3) Com as palavras codificadas, pelo segundo exemplo do enunciado, se pega essas mesmas palavras e codifica, mudando o meio delas igual no exemplo 3.

```
for j in listaCod2: #For criado para o fazer a codificação do nome, seguindo o TERCEIRO exemplo do enunciado
    listaCod3.append(j[::-1]) #Faz a codificação seguindo o terceiro exemplo, e adiciona em uma lista
nomeJunto3 = " ".join(listaCod3) #Cria-se o método 'join' para juntar o nome e sobrenome codificados da lista criada
```

PGIREO

FGUA

LLED

- 4) Com as palavras da etapa 3, elas são analisadas, e cada letra delas recebe sua próxima do alfabeto como no exemplo 4 do enunciado. Foi criado uma lista com cada letra do alfabeto.

```
alfabeto = ['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 k in listaCod3: #For criado para o fazer a codificação do nome, seguindo o QUARTO exemplo do enunciado
    listaSepChar.extend(k) #Extend criado para separar o nome em letras e adicionar a uma lista para a análise seguinte
    for i in listaSepChar: #For criado para passar a letra separado para análise
        for j in alfabeto: #Aqui será feita a análise
            if(i == j): #Se a letra for igual a que consta no alfabeto, será feita a codificação
                valor = (alfabeto.index(i) + 1) #Indice da próxima letra do alfabeto após a análise
                listaCod4.append(alfabeto[valor]) #Adiciona a próxima letra em uma lista
            nomeJunto4 = "".join(listaCod4) #Junta o nome e o sobrenome codificados
```

QHJSFP

GHVB

EMFM

## Junção do nome codificado do bolsista e suas informações

E finalmente temos a exibição do nome codificado, junto com suas informações solicitadas no enunciado

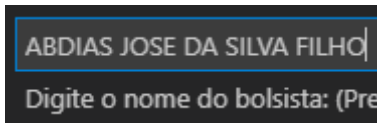
```
#Pega as informações apenas daquele nome
nome = csv.loc[csv["NM_BOLSISTA"]==nome]

#A linha abaixo retira as informações desnecessárias
df = nome.head().drop(columns=["NM_BOLSISTA", "ME_REFERENCIA", "SG_DIRETORIA", "SG_SISTEMA_ORIGEM", "CPF_BOLSISTA",
                               "CD_MODALIDADE_SGB", "DS_MODALIDADE_PAGAMENTO", "CD_MOEDA"])

#Try e except ve se o nome está certo
try:
    #Criado um novo DataFrame e adiciona a coluna com o nome codificado
    novoDF = df.assign(NM_CODIFICADO=nomeJunto4)
    display(novoDF)
except UnboundLocalError:
    print("Data de entrada inválido, por favor execute novamente a funcionalidade e insira o dado correto")
```

## Testes da funcionalidade 2. [Codificar nomes]:

- Teste 1: O usuário executou a funcionalidade 2[Codificar nomes].  
Pede-se para digitar o nome do bolsista, foi digitado o nome ABDIAS JOSE DA SILVA FILHO:



Printa na tela as informações:

	NM_ENTIDADE_ENSINO	AN_REFERENCIA	VL_BOLSISTA_PAGAMENTO	NM_CODIFICADO
16237	UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO	2016	765	BBJECTBBJECTKTPFBBJECTKTPFEBBBJECTKTPFEBTWMJBB...
19604	UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO	2015	765	BBJECTBBJECTKTPFBBJECTKTPFEBBBJECTKTPFEBTWMJBB...
40878	UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO	2014	765	BBJECTBBJECTKTPFBBJECTKTPFEBBBJECTKTPFEBTWMJBB...
72734	UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO	2013	765	BBJECTBBJECTKTPFBBJECTKTPFEBBBJECTKTPFEBTWMJBB...

### 3. [Consultar média anual]

## Permitir que o usuário informe o ano desejado

É implementado um input() para o usuário informar o ano desejado, mudando o tipo para Int para poder funcionar na busca dele pelo Data Frame nos próximos passos. Foi implementado as funções try() e except(), para executar a funcionalidade ou mostrar uma mensagem de erro.

```
#Foi criado uma variável chamada 'ano', para receber o ano desejado, informado pelo usuário
ano = int(input("Informe o ano desejado:"))
```

## Novo Data Frame com apenas informações daquele ano informado

Foi implementado um método `loc()`, para que se forme um novo Data Frame, com apenas as informações do ano informado.

```
#Criado um método 'loc' para a busca apenas do ano indicado na coluna, e extraído sua linha, cocatenando e formando um novo DF com info. desejadas
dfAno = csv.loc[csv["AN_REFERENCIA"]==ano]
```

## Média da coluna, do novo Data Frame, de valores das bolsas

Pega-se a coluna com todos os valores das bolsas daquele ano, e se implanta o método `mean()` para média dos valores das bolsas do ano informado. Foi implementado também um tipo `Int` para o valor ficar inteiro.

```
8
9 #Faz a média dos valores das bolsas, do ano de toda a coluna
10 mediaValorBolsas = int(dfAno["VL_BOLSISTA_PAGAMENTO"].mean())
11
```

## Testes da funcionalidade 3. [Consultar média anual]:

- Teste 1: O usuário executou a funcionalidade 3. [Consultar média anual]

Pede-se para informar o ano, foi digitado o ano de 2013:

```
2013|
Informe o ano desejado:
```

Printa na tela a média do valor da bolsa daquele ano:

```
13 #Printa na tela a média do valores das bolsas de determinado ano
14 print("A média dos valores das bolsas do ano de", ano, "é de R$", mediaValorBolsas)
✓ 2.2s
A média dos valores das bolsas do ano de 2013 é de R$ 935
```

- Teste 2: O usuário executou a funcionalidade 3. [Consultar média anual]

Pede-se para digitar o ano, foi digitado o ano de 2014:

```
2014|
Informe o ano desejado:
```

Printa na tela a média do valor da bolsa daquele ano:

```
13 #Printa na tela a média do valores das bolsas de determinado ano
14 print("A média dos valores das bolsas do ano de", ano, "é de R$", mediaValorBolsas)
✓ 1.5s
A média dos valores das bolsas do ano de 2014 é de R$ 936
```

- Teste 3: O usuário executou a funcionalidade 3. [Consultar média anual]

Pede-se para digitar o ano, foi digitado o ano de 2016:

2016

Informe o ano desejado:

Printa na tela a média do valor da bolsa daquele ano:

```
13 #Printa na tela a média do valores das bolsas de determinado ano
14 print("A média dos valores das bolsas do ano de", ano, "é de R$", mediaValorBolsas)
✓ 1.4s

A média dos valores das bolsas do ano de 2016 é de R$ 966
```

4. [Consultar média anual]

## Ranking

Quando a funcionalidade 4 é executada, printa na tela duas colocações, um dos três alunos com bolsas mais altas e outra com os três alunos com bolsas mais baixas.

```
18 fun4()
✓ 0.1s

RANKING DOS BOLSISTAS COM A BOLSA MAIS ALTAS
NM_BOLSISTA
MARIA APARECIDA DA SILVA      16785
MARIA DE FATIMA DO NASCIMENTO  8460
LILIAN CRISTINA DOS SANTOS    8300
Name: VL_BOLSISTA_PAGAMENTO, dtype: int64

RANKING DOS BOLSISTAS COM A BOLSA MAIS BAIXAS
NM_BOLSISTA
ABDON MENDES BORGES SANTANA    765
ABDON SILVA RIBEIRO DA CUNHA   765
ABEDENEGU DOS SANTOS RIBEIRO  765
Name: VL_BOLSISTA_PAGAMENTO, dtype: int64
```

## Agrupamento de dados

Esse código foi feito com o método groupby(), juntando todos os nomes iguais e somando com sum() as colunas de valores das bolsas, para obter as pessoas com maior e menor ranking.

```
#Agrupar os nomes repetidos e faz a soma dos valores das bolsas dos bolsistas
agrupamento = csv.groupby("NM_BOLSISTA")["VL_BOLSISTA_PAGAMENTO"].sum()
```

## Pegar maiores e menores valores

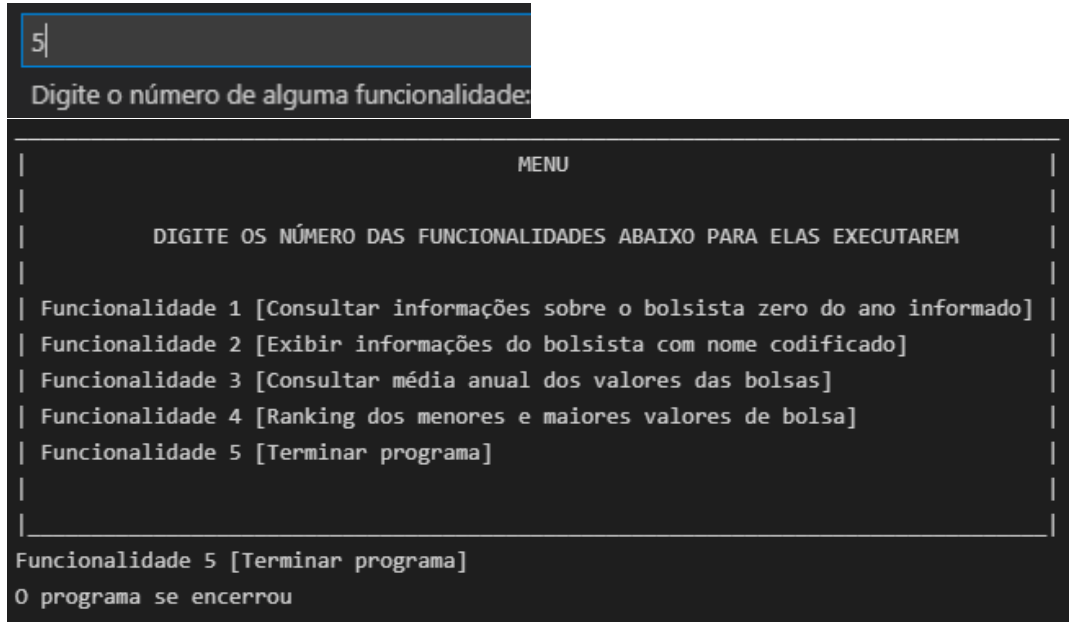
Para pegar os três maiores e os três menores valores de bolsa de determinados alunos, foram usados os métodos `nlargest()` e `nsmallest()`.

```
#Aqui será feito a análise do três alunos com os valores da bolsas mais altas e baixas
bolsasAltas = agrupamento.nlargest(3)
bolsasBaixas = agrupamento.nsmallest(3)
```

### 5. [Terminar o programa]

A funcionalidade 5 foi feita a partir de um `while()`, como mostra no tópico de **interação do usuário ao programa**. Essa funcionalidade foi feita a partir apenas de um comando `input()`, para executá-la dentro uma estrutura de repetição `while()`.

- Teste 1: O usuário executou a funcionalidade 5. [Terminar o programa]  
Então ele digita no menu o número da funcionalidade 5 [Terminar o programa], sendo assim encerra-se o programa:



```
5|
Digite o número de alguma funcionalidades:

|                                     MENU                                     |
|                                                                           |
|          DIGITE OS NÚMERO DAS FUNCIONALIDADES ABAIXO PARA ELAS EXECUTAREM          |
|                                                                           |
| Funcionalidade 1 [Consultar informações sobre o bolsista zero do ano informado] |
| Funcionalidade 2 [Exibir informações do bolsista com nome codificado]          |
| Funcionalidade 3 [Consultar média anual dos valores das bolsas]                |
| Funcionalidade 4 [Ranking dos menores e maiores valores de bolsa]              |
| Funcionalidade 5 [Terminar programa]                                           |
|                                                                           |
|                                                                           |
Funcionalidade 5 [Terminar programa]
O programa se encerrou
```

## ***Autoavaliação***

Após ter terminado o programa, me senti muito bem, pois consegui um desempenho bom na programação de todas as funcionalidades, fiz elas funcionarem com eficiência. Tive muita ajuda dos métodos, inclusive dos da biblioteca Pandas, os quais facilitaram minha programação nas funcionalidades e na leitura do arquivo .csv, pois ajudaram na leitura, análise e manipulação de dados do arquivo .csv.

Tive apenas dificuldades na programação da funcionalidade 2 [Codificar nomes], mais especificamente em deixar a codificação igual ao do exemplo 4 do enunciado, que era pegar a próxima letra do alfabeto. Na funcionalidade 4 [Ranking valores de bolsa], tive dificuldade também, porém o problema não era na programação e sim em como ela estava escrita no enunciado, achei mal especificada, dificultando a minha programação.