

Estrutura de Arquivos

Gerenciador de *Archives*

Brenda Alexsandra Januário

Felipe Carreiro Marchi

Pedro Henrique Bernini Silva

Descrição do problema

O projeto solicitou: criar um “arquivo de arquivos” (*archive*), ou seja, um arquivo que contém uma coleção de outros arquivos, em uma estrutura que torna possível recuperar individualmente cada arquivo nele armazenado.

Junto à descrição do problema encontrou-se restrições e obrigações, das quais retirou-se os seguintes critérios:

Fazer um arquivador que funcione em modo texto (Prompt do Windows ou terminal do Linux) e implemente os seguintes casos de uso:

- Criar um archive com base em uma lista de arquivos informados;
- Inserir um arquivo em um archive já criado;
- Listar os nomes dos arquivos armazenados em um archive;
- Extrair um arquivo de um archive, dado o nome do arquivo (sem remover esse arquivo de dentro do archive);
- Remover um arquivo de um archive, dado o nome do arquivo.

Solução do problema

Primeiramente, com base nos critérios retirados da descrição do problema, o grupo optou por programar em linguagem Python 3 e utilizar o terminal do Sistema Operacional Linux para execução do programa.

Para cada caso de uso foi reservado um comando que o programa receberá por linha de comando. Veja abaixo os comandos reservados para cada caso de uso:

- Comando “-c” está reservado para a criação de um novo *archive*;
- Comando “-i” está reservado para a inserção de um arquivo dentro de um *archive*;
- Comando “-l” está reservado para listar os arquivos contidos dentro de um *archive*;
- Comando “-e” está reservado para a extração de um arquivo contido em um *archive*;
- Comando “-r” está reservado para a remoção de um arquivo contido em um *archive*.

Descrição dos casos de uso do programa:

- CASO DE USO 01 - CRIAR ARCHIVE

Este caso de uso do programa cria um *archive* a partir de n arquivos fornecidos por linha de comando. O *archive* pode possuir qualquer nome e extensão também fornecidos por linha de comando, desde que não exista um *archive* idêntico já criado.

```
# ----- CRIAR ARQUIVADOR COM ARQUIVOS ----- #
def criar() :
    cont = 0
    try :
        open(arquivador, 'r+b')
        print("A CRIACAO FALHO!")
        print("Ja existe um arquivador de mesmo nome neste local.")
        print("")

    except FileNotFoundError :
        archive = open(arquivador, 'w+b')

        for arq in arquivos :
            qtdBytes = 0

            try :
                conteudo = (open(arq, 'r+b').read())

                for byte in conteudo :
                    qtdBytes += 1

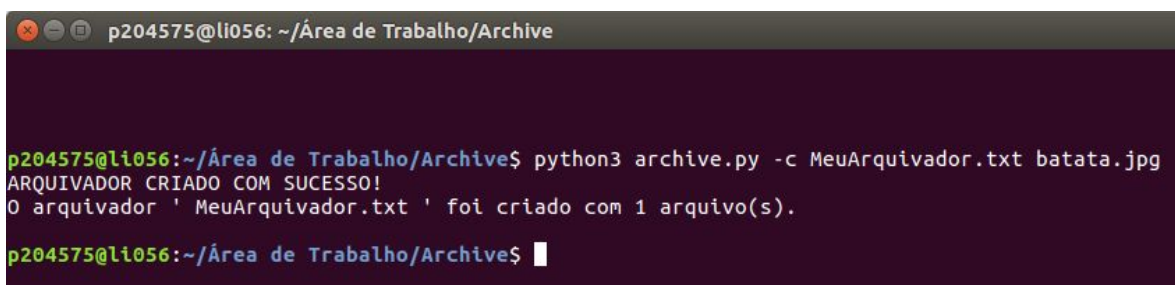
                # INSERIR
                nome = arq
                tamanho = qtdBytes
                novoArquivo = str.encode(nome) + str.encode("|") + str.encode(str(tamanho)) + str.encode("|") + conteudo
                archive.write(novoArquivo)
                cont += 1

            except FileNotFoundError :
                print("UMA INSERCAO FALHO!")
                print("Nao existe o arquivo '", arq, "' neste local.")
                print("")

        archive.close()

    print("ARQUIVADOR CRIADO COM SUCESSO!")
    print("O arquivador '", arquivador, "' foi criado com", cont, "arquivo(s).")
    print("")
```

Figura 01 - Trecho de código que cria archive



```
p204575@li056: ~/Área de Trabalho/Archive
p204575@li056:~/Área de Trabalho/Archive$ python3 archive.py -c MeuArquivador.txt batata.jpg
ARQUIVADOR CRIADO COM SUCESSO!
O arquivador 'MeuArquivador.txt' foi criado com 1 arquivo(s).
p204575@li056:~/Área de Trabalho/Archive$
```

Figura 02 - Exemplo de execução em terminal Linux

- CASO DE USO 02 - INSERIR ARQUIVO EM ARCHIVE

Este caso de uso do programa insere um arquivo fornecido por linha de comando dentro *archive*, desde que não exista nenhum arquivo idêntico a ele contido no *archive*.

```
# ----- INSERIR UM ARQUIVO ----- #
def inserir() :
    try :
        open(arquivador, 'r+b')
        archive = open(arquivador, 'a+b')

        for arq in arquivos :
            qtdBytes = 0

            if arq in lista() :
                print("A INSERCAO FALHO!")
                print("Ja existe o arquivo '", arq, "' neste arquivador.")
                print("")
                return

            else :
                try :
                    conteudo = (open(arq, 'r+b').read())

                    for byte in conteudo :
                        qtdBytes += 1

                    # INSERIR
                    nome = arq
                    tamanho = qtdBytes
                    novoArquivo = str.encode(nome) + str.encode("|") + str.encode(str(tamanho)) + str.encode("|") + conteudo
                    archive.write(novoArquivo)

                except FileNotFoundError :
                    print("A INSERCAO FALHO!")
                    print("Nao existe o arquivo '", arq, "' neste local.")
                    print("")

        archive.close()

        print("INSERIDO COM SUCESSO!")
        print("O arquivo '", arq, "' foi inserido no arquivador.")
        print("")

    except FileNotFoundError :
        print("A INSERCAO FALHO!")
        print("Nao existe o arquivador '", arquivador, "' neste local.")
        print("")
```

Figura 03 - Trecho de código que insere arquivo em archive

```
p204575@li056: ~/Área de Trabalho/Archive

p204575@li056:~/Área de Trabalho/Archive$ python3 archive.py -i MeuArquivador.txt meuarquivo.txt
INSERIDO COM SUCESSO!
O arquivo ' meuarquivo.txt ' foi inserido no arquivador.

p204575@li056:~/Área de Trabalho/Archive$
```

Figura 04 - Exemplo de execução em terminal Linux

- CASO DE USO 03 - LISTAR ARQUIVOS DO ARCHIVE

Este caso de uso do programa lista todos os arquivos contidos dentro do *archive*.

```
# ----- LISTAR ARQUIVOS DO ARQUIVADOR ----- #
def lista() :
    try :
        listaNomes = []
        archive = open(arquivador, 'r+b')
        caractere = archive.read(1).decode("utf-8")

        while(caractere != '') :

            # PEGA NOME
            nome = ""
            while(caractere != '|') :
                nome += caractere
                caractere = archive.read(1).decode("utf-8")
            listaNomes.append(nome)

            # PEGA QTDBYTES
            qtdBytes = ""
            caractere = archive.read(1).decode("utf-8")
            while(caractere != '|') :
                qtdBytes += caractere
                caractere = archive.read(1).decode("utf-8")

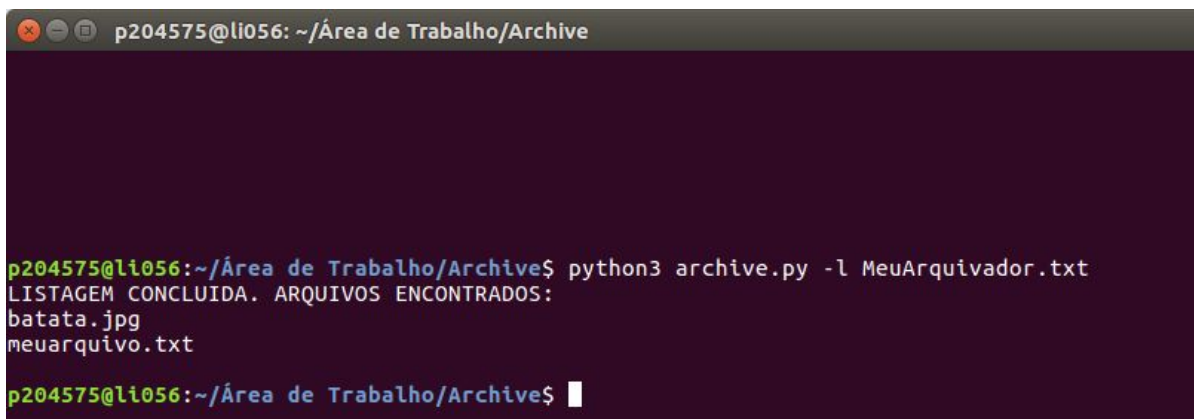
            # PULA CONTEUDO
            archive.read(int(qtdBytes))
            caractere = archive.read(1).decode("utf-8")

        archive.close()

        # RETORNAR LISTA DE NOMES
        return listaNomes

    except FileNotFoundError :
        print("A LISTAGEM FALHO!")
        print("Nao existe o arquivador '", arquivador, "' neste local.")
        print("")
```

Figura 05 - Trecho de código que lista arquivos de um archive



```
p204575@li056: ~/Área de Trabalho/Archive

p204575@li056:~/Área de Trabalho/Archive$ python3 archive.py -l MeuArquivador.txt
LISTAGEM CONCLUIDA. ARQUIVOS ENCONTRADOS:
batata.jpg
meuarquivo.txt

p204575@li056:~/Área de Trabalho/Archive$
```

Figura 06 - Exemplo de execução em terminal Linux

- CASO DE USO 04 - EXTRAIR ARQUIVO DO ARCHIVE

Este caso de uso do programa extrai um arquivo fornecido por linha de comando se contido no *archive*.

```
# ----- EXTRAIR UM ARQUIVO ----- #
def extrair() :
    try :
        nome = ""
        archive = open(arquivador, 'r+b')
        caractere = archive.read(1).decode("utf-8")

        while(caractere != '' and nome != arquivos[0]) :

            # PEGA NOME
            nome = ""
            while(caractere != '|') :
                nome += caractere
                caractere = archive.read(1).decode("utf-8")

            # PEGA QTDBYTES
            qtdBytes = ""
            caractere = archive.read(1).decode("utf-8")
            while(caractere != '|') :
                qtdBytes += caractere
                caractere = archive.read(1).decode("utf-8")

            # PEGA CONTEUDO
            conteudo = archive.read(int(qtdBytes))
            caractere = archive.read(1).decode("utf-8")

        archive.close()

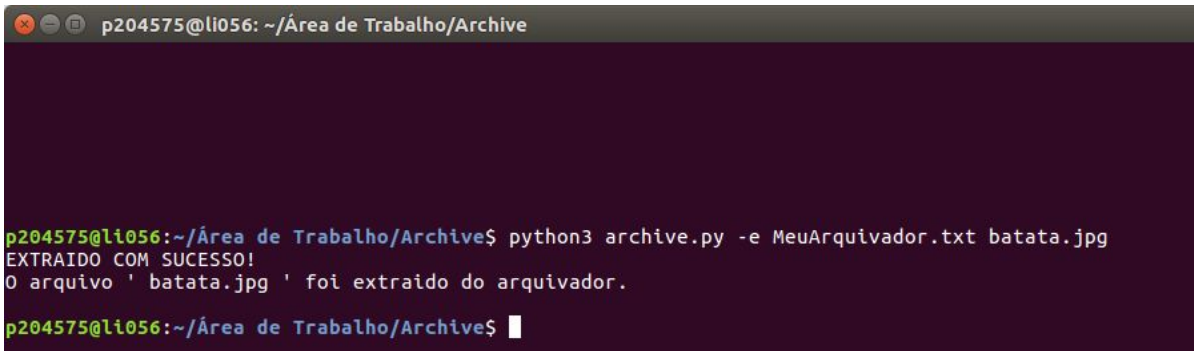
        # EXTRAIR
        if (nome == arquivos[0]) :
            arq = open(nome, 'w+b')
            arq.write(conteudo)
            arq.close()

            print("EXTRAIDO COM SUCESSO!")
            print("O arquivo '", arquivos[0] ,"' foi extraido do arquivador.")
            print("")

        else :
            print("A EXTRACAO FALHO!")
            print("Nao existe o arquivo '", arquivos[0] ,"' neste arquivador.")
            print("")

    except FileNotFoundError :
        print("A EXTRACAO FALHO!")
        print("Nao existe o arquivador '", arquivador ,"' neste local.")
        print("")
```

Figura 07 - Trecho de código que extrai um arquivo do archive



```
p204575@li056: ~/Área de Trabalho/Archive

p204575@li056:~/Área de Trabalho/Archive$ python3 archive.py -e MeuArquivador.txt batata.jpg
EXTRAIDO COM SUCESSO!
O arquivo ' batata.jpg ' foi extraído do arquivador.

p204575@li056:~/Área de Trabalho/Archive$
```

Figura 08 - Exemplo de execução em terminal Linux

- CASO DE USO 05 - REMOVER ARQUIVO DO ARCHIVE

Este caso de uso do programa remove um arquivo fornecido por linha de comando se contido no *archive*.

```
# ----- REMOVER UM ARQUIVO ----- #
def remover() :
    try :
        nome = ""
        archive = open(arquivador, 'r+b')
        caractere = archive.read(1).decode("utf-8")

        while(caractere != '' and nome != arquivos[0]) :

            # PEGA POSICAO INICIAL DO ARQUIVO
            posicaoInicial = archive.tell() - 1

            # PEGA NOME
            nome = ""
            while(caractere != '|') :
                nome += caractere
                caractere = archive.read(1).decode("utf-8")

            # PEGA QTDBYTES
            qtdBytes = ""
            caractere = archive.read(1).decode("utf-8")
            while(caractere != '|') :
                qtdBytes += caractere
                caractere = archive.read(1).decode("utf-8")

            # PULA CONTEUDO
            archive.read(int(qtdBytes))

            # PEGA POSICAO FINAL DO ARQUIVO
            posicaoFinal = archive.tell()

            caractere = archive.read(1).decode("utf-8")

    archive.close()
```



```

# REMOVER
if (nome == arquivos[0]) :

    # SALVAR CONTEUDO PERSISTENTE DO ARQUIVADOR
    archive = open(arquivador, 'r+b')
    archive.seek(posicaoFinal)
    conteudoPersistente = archive.read()
    archive.close()

    # REMOVER ARQUIVO DO ARQUIVADOR
    archive = open(arquivador, 'a+b')
    archive.truncate(posicaoInicial)
    archive.close()

    # RESTAURAR ARQUIVOS PERSISTENTES NO ARQUIVADOR
    archive = open(arquivador, 'a+b')
    archive.seek(posicaoInicial)
    archive.write(conteudoPersistente)
    archive.close()

    print("REMOVIDO COM SUCESSO!")
    print("O arquivo '", arquivos[0] ,"' foi removido do arquivador.")
    print("")

else :
    print("A REMOCAO FALHO!")
    print("Nao existe o arquivo '", arquivos[0] ,"' neste arquivador.")
    print("")

except FileNotFoundError :
    print("A REMOCAO FALHO!")
    print("Nao existe o arquivador '", arquivador ,"' neste local.")
    print("")

```

Figura 09 - Trecho de código que remove um arquivo do archive



```

p204575@li056: ~/Área de Trabalho/Archive

p204575@li056:~/Área de Trabalho/Archive$ python3 archive.py -r MeuArquivador.txt batata.jpg
REMOVIDO COM SUCESSO!
O arquivo ' batata.jpg ' foi removido do arquivador.

p204575@li056:~/Área de Trabalho/Archive$

```

Figura 08 - Exemplo de execução em terminal Linux

Instruções para compilar/executar

A pasta projeto contém o arquivo *archive.py*. Mova-se até esta pasta via terminal Linux e insira o seguinte comando :

\$ python3 archive.py <comando> <archive.arq> <arq1> <arq2> ...

Onde:

- <comando> representa um dos 5 comandos reservados para cada caso de uso;
- <archive.arq> representa o nome do arquivador e sua extensão;
- <arq1> <arq2> ... representa um conjunto de arquivos que irão fazer parte do caso de uso.

Observação:

- Os arquivos passados por parâmetros também devem ter extensão, e devem ser colocados dentro do diretório do projeto antes de sua execução.