# Arquivos

Leonardo Murta
leomurta@ic.uff.br

# Aula de hoje

- Até então só consideramos teclado e monitor como mecanismos de entrada e saída

- Veremos como ler e escrever em arquivos

# Motivação

- Em algumas situações é desejado ler dados de arquivos e escrever dados em arquivos
  - Não é necessário digitar via teclado os dados a cada execução do programa
  - Os resultados do programa podem ser impressos ou enviados para outras pessoas com mais facilidade
  - O estado do programa (jogo, por exemplo) pode ser salvo e recarregado em outro momento

# Operações Básicas

- Abertura do arquivo
  - Liga uma variável do programa com o arquivo físico
  - Essa variável deve ser usada no programa para manipular o arquivo (ler e escrever no arquivo)
- Leitura/Escrita no arquivo
- Fechamento do arquivo
  - Encerramento da conexão da variável com o arquivo físico

# Escrita de arquivos

- É muito parecido com escrita no monitor, só que precisa conectar com o arquivo antes (abrir o arquivo)
- Vamos ver um exemplo...

# Exemplo: escrevendo números aleatórios **no monitor**

```
import random

def escrever_numeros_aleatorios(qtd_numeros):
    for i in range(qtd_numeros):
        print(random.randint(0,100))

escrever_numeros_aleatorios(100)
```

# Exemplo: escrevendo números aleatórios **em um arquivo**

```
import random

def escrever_numeros_aleatorios(qtd_numeros, nome_arquivo):
    arquivo_numeros = open(nome_arquivo, 'w')
    for i in range(qtd_numeros):
        arquivo_numeros.write(str(random.randint(0,100))
        arquivo_numeros.write("\n")
    arquivo_numeros.close()

escrever_numeros_aleatorios(100, 'aleatorios.txt')
```

# Exemplo: escrevendo números aleatórios **em um arquivo**

O arquivo aparecerá na raiz do projeto do PyCharm

```
import random

def escrever_numeros_aleatorios(qtd_numeros, nome_arquivo):
    arquivo_numeros = open(nome_arquivo, 'w')
    for i in range(qtd_numeros):
        arquivo_numeros.write(str(random.randint(0,100))
        arquivo_numeros.write("\n")
    arquivo_numeros.close()

escrever_numeros_aleatorios(100, 'aleatorios.txt')
```

**Abertura do arquivo para escrita e posterior fechamento**

# Exemplo: escrevendo números aleatórios **em um arquivo**

```
import random

def escrever_numeros_aleatorios(qtd_numeros, nome_arquivo):
    arquivo_numeros = open(nome_arquivo, 'w')
    for i in range(qtd_numeros):
        arquivo_numeros.write(str(random.randint(0,100))
        arquivo_numeros.write("\n")
    arquivo_numeros.close()

escrever_numeros_aleatorios(100, 'aleatorios.txt')
```

**Escrita no arquivo**

# Leitura de arquivos

- Novamente, é muito parecido com leitura do teclado, só que precisa conectar com o arquivo antes (abrir o arquivo)
- Vamos ver um exemplo...

# Exemplo: lendo números **do teclado** e escrevendo a média

```
def escrever_media(qtd_numeros):
    soma = 0
    for i in range(qtd_numeros):
        num = float(input("Digite o número:"))
        soma += num
    return soma/qtd_numeros

escrever_media(100)
```

# Exemplo: lendo números **de um arquivo** e escrevendo a média

O arquivo será procurado na raiz do projeto do PyCharm

```
def escrever_media(qtd_numeros, nome_arquivo):
    arquivo_numeros = open(nome_arquivo)
    soma = 0

    for i in range(qtd_numeros):
        num = float(arquivo_numeros.readline().strip())
        soma += num
    arquivo_numeros.close()
    return soma/qtd_numeros

escrever_media(100, 'media.txt')
```

**Abertura do arquivo para leitura e posterior fechamento**

# Exemplo: lendo números **de um arquivo** e escrevendo a média

```
def escrever_media(qtd_numeros, nome_arquivo):
    arquivo_numeros = open(nome_arquivo)
    soma = 0

    for i in range(qtd_numeros):
        num = float(arquivo_numeros.readline().strip())
        soma += num
    arquivo_numeros.close()
    return soma/qtd_numeros

escrever_media(100, 'media.txt')
```

**Leitura do arquivo**

# Exemplo: lendo números **de um arquivo** e escrevendo a média

```
def escrever_media(qtd_numeros, nome_arquivo):
    arquivo_numeros = open(nome_arquivo)
    soma = 0

    for i in range(qtd_numeros):
        num = float(arquivo_numeros.readline().strip())
        soma += num
    arquivo_numeros.close()
    return soma/qtd_numeros

escrever_media(100, 'media.txt')
```

**Interação no arquivo quando a quantidade de valores no arquivo é conhecida**

# Fazendo de outra forma...

```
def escrever_media(nome_arquivo):
    arquivo_numeros = open(nome_arquivo)
    soma = 0
    qtd_numeros = 0
    for num in arquivo_numeros:
        num = float(num.strip())
        soma += num
        qtd_numeros += 1
    arquivo_numeros.close()
    return soma/qtd_numeros

escrever_media('media.txt')
```

Lendo todos os valores do arquivo com o comando **for**

# Agora usando while...

Lendo todos os valores do arquivo com o comando **while**

```
def escrever_media(nome_arquivo):
    arquivo_numeros = open(nome_arquivo)
    soma = 0
    qtd_numeros = 0
    num = arquivo_numeros.readline()

    while num != "":
        num = float(num.strip())
        soma += num
        qtd_numeros += 1
        num = arquivo_numeros.readline()

    arquivo_numeros.close()
    return soma/qtd_numeros

escrever_media('media.txt')
```

# Detalhes do comando de abertura do arquivo

- var_arquivo = open(nome_arquivo, **modo**, buffering)

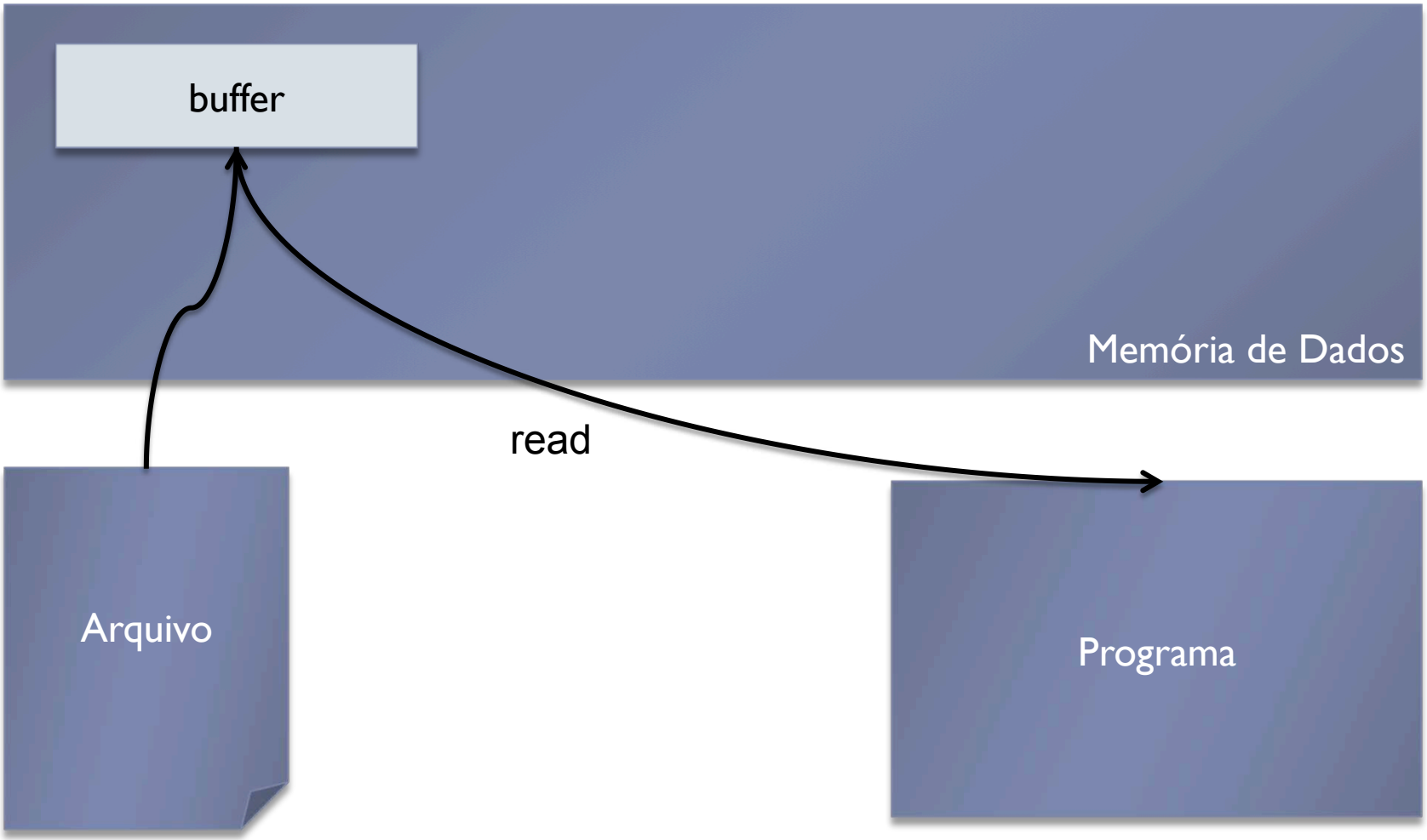
- modo
  - r: leitura (default) – o arquivo deve existir
  - w: escrita – conteúdo existente no arquivo será apagado
  - b: binário
  - a: escrita a partir do final do arquivo (*append*)
  - +: (usado com r) indica leitura e escrita

# Detalhes do comando de abertura do arquivo

- var_arquivo = open(nome_arquivo, modo, **buffering**)

- buffering (opcional)
  - Indica se memória (buffer) é usada para acelerar operações de entrada e saída
    - 0: buffers não são usados
    - 1 (ou qq número negativo): um buffer de tamanho padrão é usado (default)
    - 2 ou maior: tamanho do buffer em bytes

# Como funciona um buffer?



# Detalhes do comando de escrita

- Necessário que o arquivo **não** tenha sido aberto em modo **r**
- var_arquivo.write(string)
  - Escreve a string no arquivo
  - Devido ao uso de buffers, a escrita pode não ser feita imediatamente
  - Use o comando flush() ou close() para assegurar a escrita física no arquivo

# Detalhes do comando de escrita

- Necessário que o arquivo **não** tenha sido aberto em modo **r**
- var_arquivo.writelines(sequencia)
  - Escreve a lista de strings no arquivo, uma por uma
  - Caracteres de final de linha não são acrescentados no arquivo
    - Fica tudo numa única linha

# Detalhes do comando de leitura

- Necessário que o arquivo tenha sido aberto em modo leitura ou leitura/escrita
- var_string = var_arquivo.readline()
  - Lê uma linha do arquivo e a retorna como string
- var_lista_string = var_arquivo.readlines()
  - Lê o arquivo do ponto atual até o final, e retorna o conteúdo em uma lista de strings
  - Cada linha do arquivo é guardada em uma posição da lista

# Exemplo: Copiando dois arquivos

```
def copia_arquivo(velho_arquivo, novo_arquivo):
    f1 = open(velho_arquivo, "r")
    f2 = open(novo_arquivo, "w")
    for texto in f1:
        f2.write(texto)
    f1.close()
    f2.close()


copia_arquivo("velho.txt", "novo.txt")
```

# Interação com o OS

- Operações de entrada e saída são na verdade realizadas pelo sistema operacional
- O módulo **os** possui diversas variáveis e funções que ajudam um programa Python a se adequar ao sistema operacional

# Funções do módulo **os**

- os.getcwd()
  - Retorna o diretório corrente
- os.chdir(dir)
  - Muda o diretório corrente para **dir**
- os.sep
  - É uma string que contém o caracter que separa os componentes de um caminho (‘/" para Unix, "\\" para Windows)
- os.path.exists(path)
  - Retorna True se o arquivo **path** existe

# Exercícios

1. Faça um programa que leia um número N e gere um arquivo com N nomes e idades aleatórios
   - Faça uso de duas listas criadas na mão: uma que contenha 20 nomes e outra que contenha 20 sobrenomes
   - Cada linha do arquivo resultante deve conter um nome completo e a sua idade
2. Estenda o exemplo do cadastro para considerar também a altura da pessoa
   - Armazene a altura como float

# Exercícios

3. Escreva uma função que recebe dois nomes de arquivos e copia o conteúdo do primeiro arquivo para o segundo arquivo. Considere que o conteúdo do arquivo de origem é um texto. Sua função não deve copiar linhas comentadas (que começam com //)

4. Faça um programa contendo uma função que recebe como argumentos os nomes de dois arquivos. O primeiro arquivo contém nomes de alunos e o segundo arquivo contém as notas dos alunos. No primeiro arquivo, cada linha corresponde ao nome de um aluno e no segundo arquivo, cada linha corresponde às notas dos alunos (uma ou mais). Assuma que as notas foram armazenadas como strings, e estão separadas umas das outras por espaços em branco. Leia os dois arquivos e gere um terceiro arquivo que contém o nome do aluno seguido da média de suas notas.

# Exercícios

5. Faça um programa para alterar uma das notas de um aluno (usando os arquivos do exercício anterior). O programa deve ter uma função que recebe o nome do aluno, a nota velha e a nova nota. A função deve fazer a alteração no arquivo.

6. Faça uma função que leia um arquivo texto contendo uma lista de endereços IP e gere dois outros arquivos, um contendo os endereços IP válidos e outro contendo os endereços inválidos. O formato de um endereço IP é **num1.num.num.num**, onde **num1** vai de 1 a 255 e **num** vai de 0 a 255.

# Referências

- Slides feitos em conjunto com Aline Paes e Vanessa Braganholo

# Arquivos

Vanessa Braganholo
vanessa@ic.uff.br