

Curso de C

Comunicação e Arquivos

Tipos Avançados de Dados

Roteiro:

- Fluxos de Dados
- Arquivos
 - Abrir/Fechar
 - Ler/Escrever
 - Outras operações
- Entrada/Saída Padrão

Comunicação e Arquivos

Modelo de Fluxo de Dados

Fluxo de Dados: definição

É a comunicação entre o programa e outras entidades:

- Teclado
- Terminal/prompt DOS
- Arquivos
- Conexões de rede, Bluetooth
- Impressoras
- Portas seriais, USB, infra-vermelho
- Outros programas



Fluxo de Dados: vantagens

Modelo de comunicação genérico e unificado:

- Qualquer dispositivo parece funcionar da “mesma maneira”.
- Programas mais simples.
- Independência de plataforma e sistema operacional.

Fluxo de Dados: modelo

- Produtor: escreve no fluxo

Fluxo de Dados: modelo

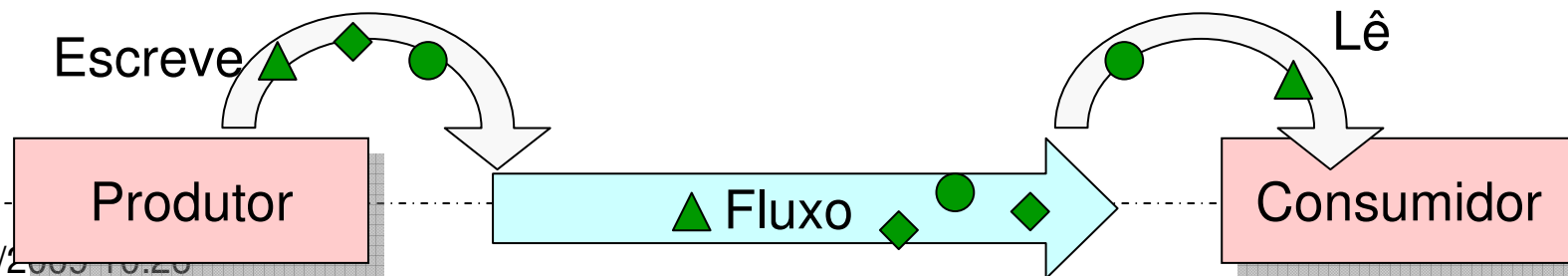
- Produtor: escreve no fluxo
- Consumidor: lê do fluxo

Fluxo de Dados: modelo

- **Produtor:** escreve no fluxo
- **Consumidor:** lê do fluxo
- **Fluxo:** fila de entrega de dados
 - O fluxo preserva a ordem
 - Produtor e consumidor operam em ritmos diferentes

Fluxo de Dados: modelo

- **Produtor:** escreve no fluxo
- **Consumidor:** lê do fluxo
- **Fluxo:** fila de entrega de dados
 - O fluxo preserva a ordem
 - Produtor e consumidor operam em ritmos diferentes



Fluxo de Dados: exemplo

- No fluxo, o programa pode ser consumidor!

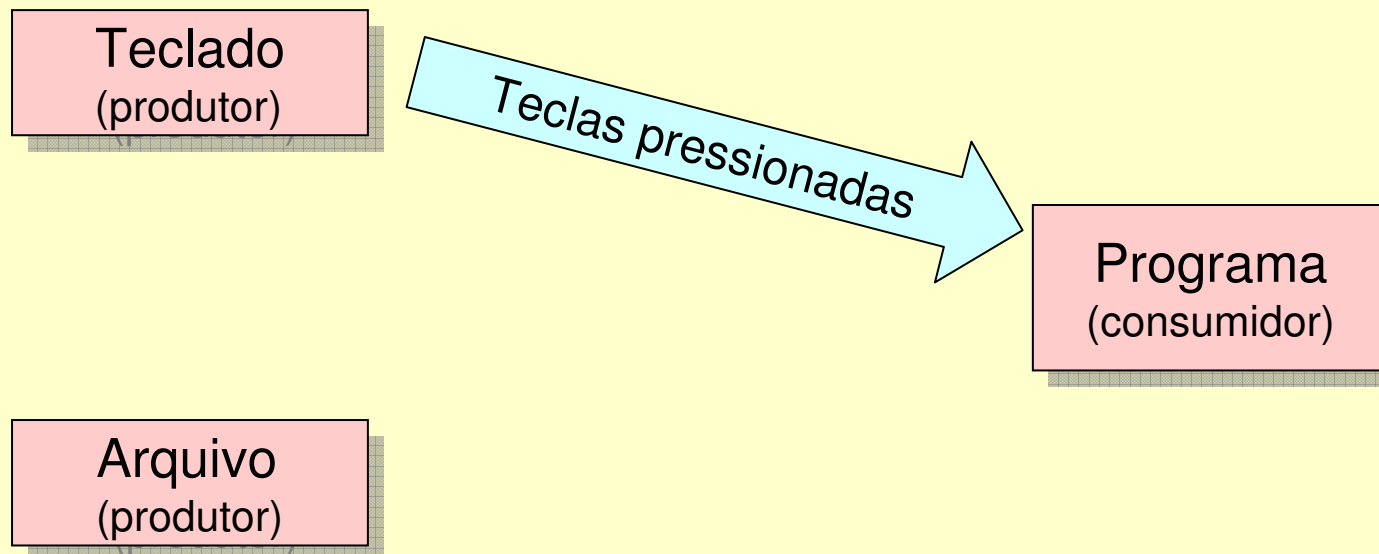
Teclado
(produtor)

Programa
(consumidor)

Arquivo
(produtor)

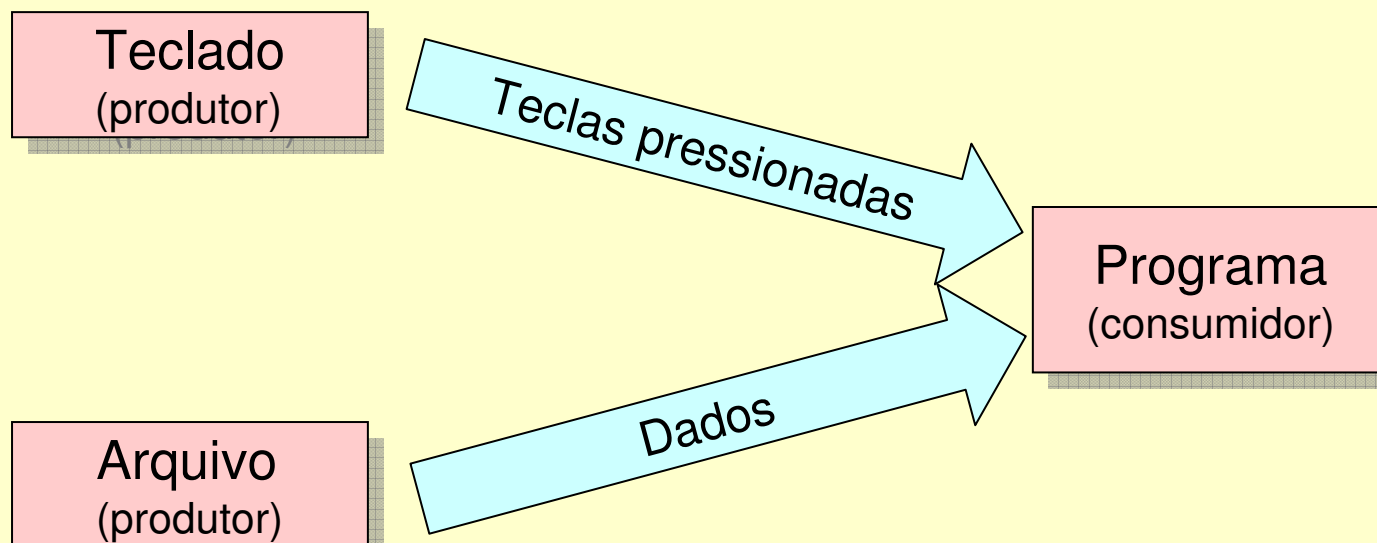
Fluxo de Dados: exemplo

- No fluxo, o programa pode ser consumidor!



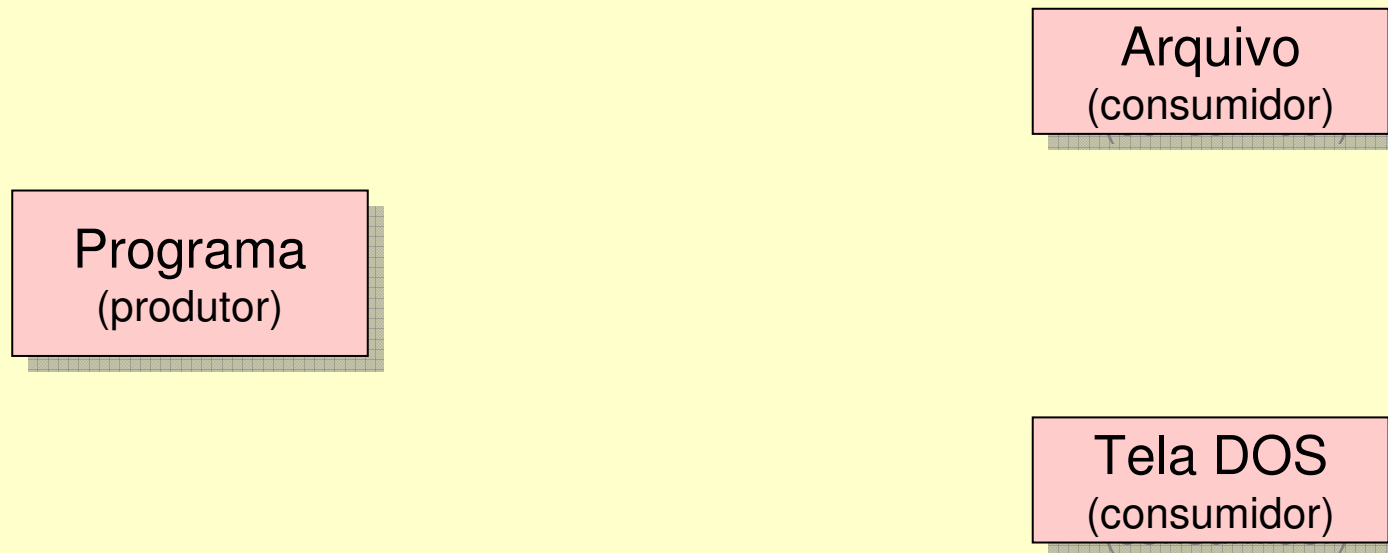
Fluxo de Dados: exemplo

- No fluxo, o programa pode ser consumidor!



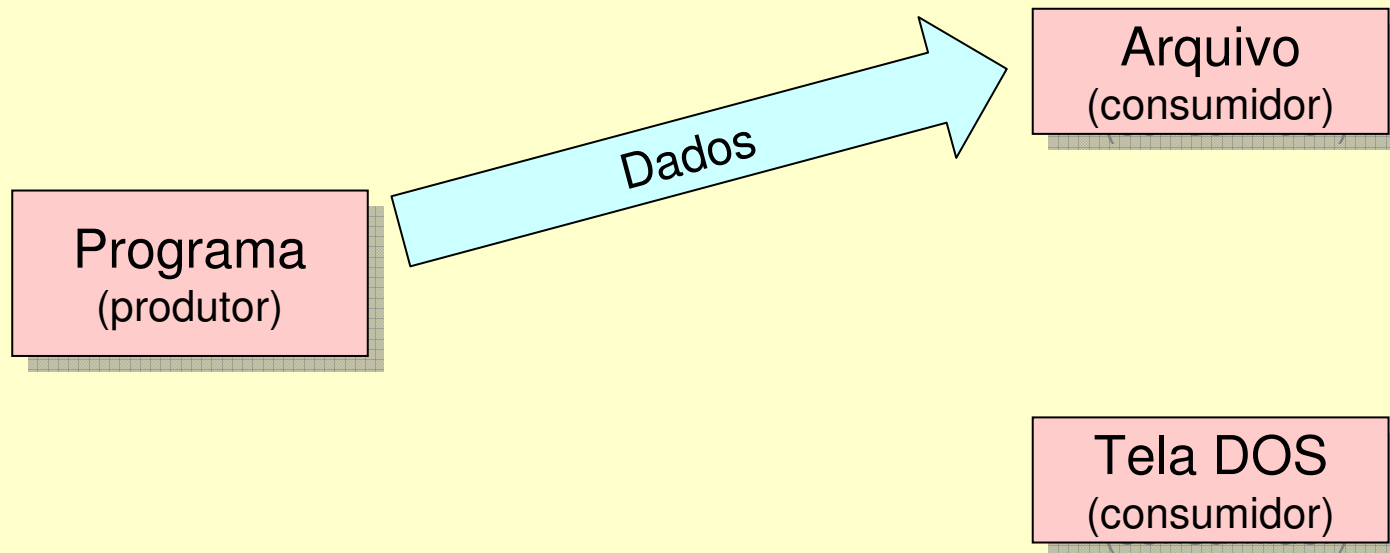
Fluxo de Dados: exemplo

- No fluxo, o programa pode ser produtor!



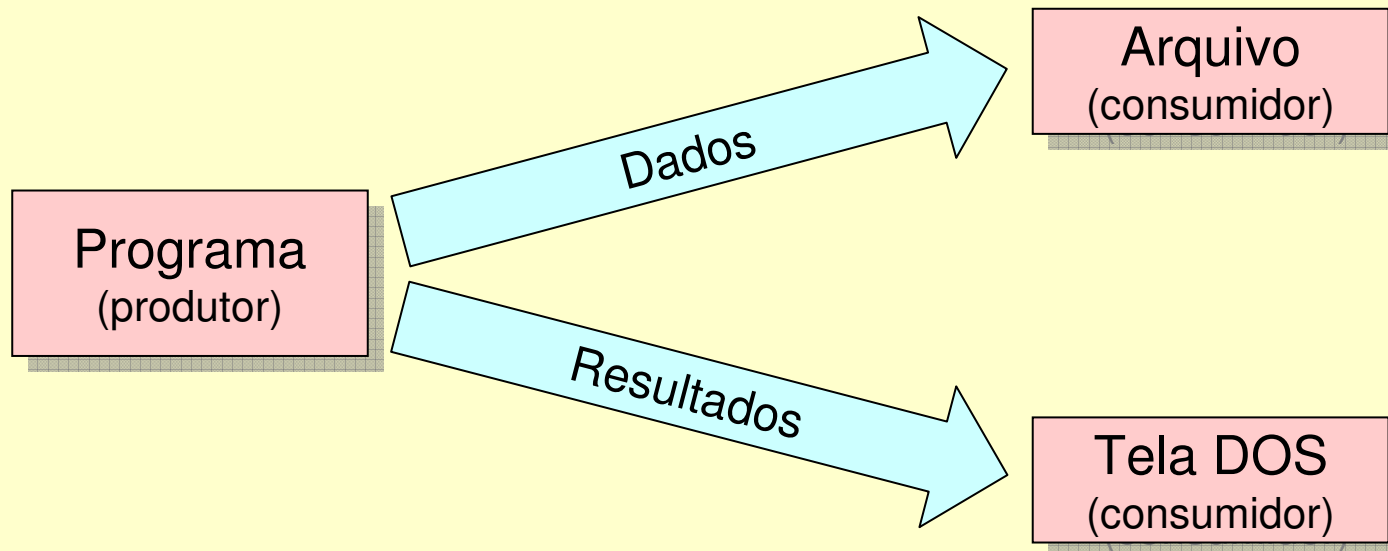
Fluxo de Dados: exemplo

- No fluxo, o programa pode ser produtor!



Fluxo de Dados: exemplo

- No fluxo, o programa pode ser produtor!



Fluxo de Dados: tipos

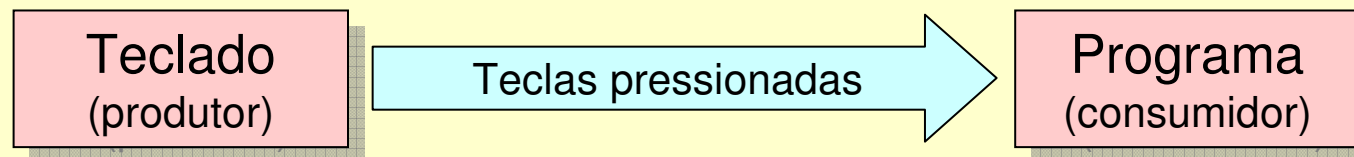
- Somente leitura

Teclado
(produtor)

Programa
(consumidor)

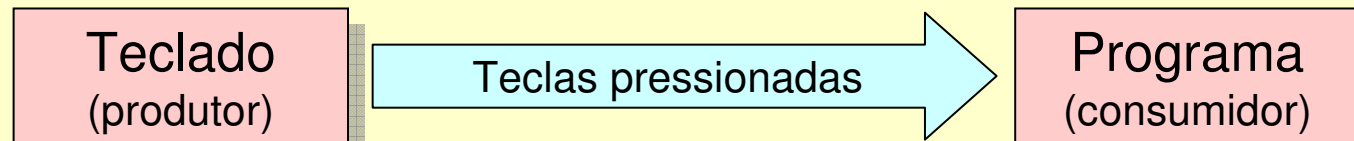
Fluxo de Dados: tipos

- Somente leitura

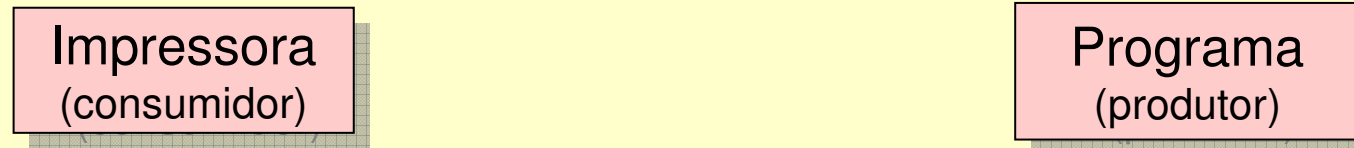


Fluxo de Dados: tipos

- Somente leitura

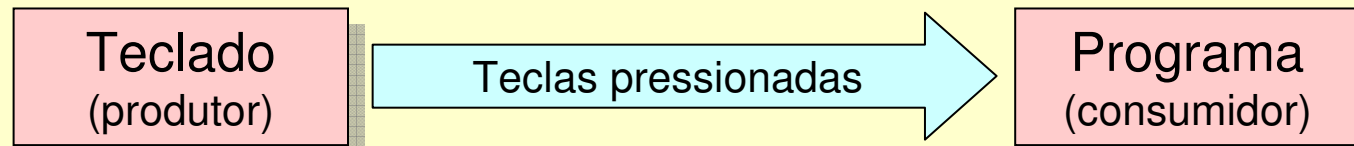


- Somente escrita

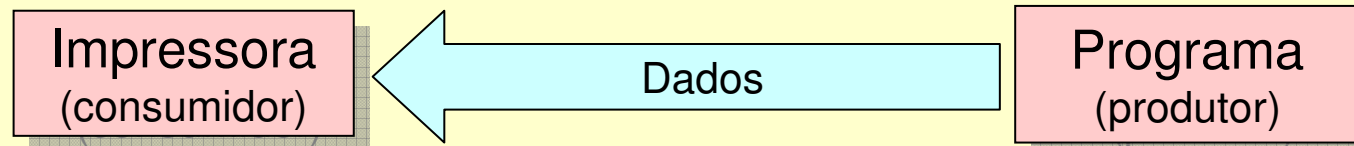


Fluxo de Dados: tipos

- Somente leitura

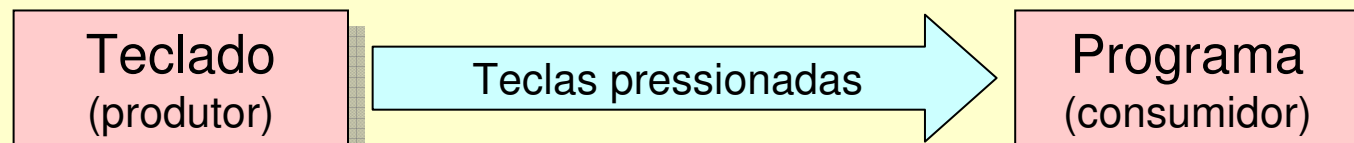


- Somente escrita

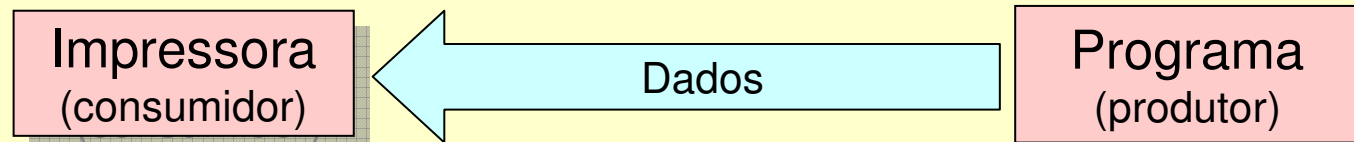


Fluxo de Dados: tipos

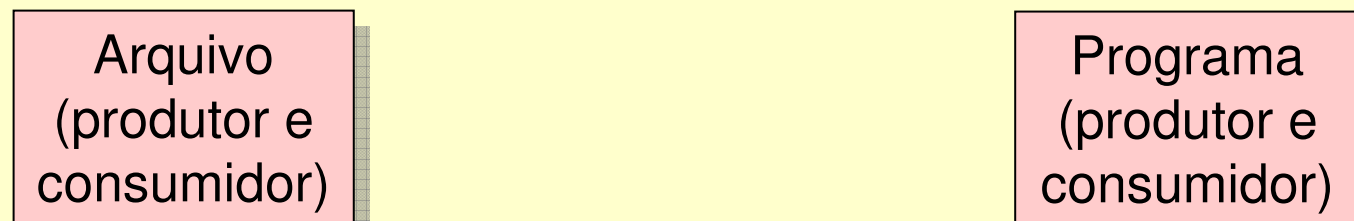
- Somente leitura



- Somente escrita

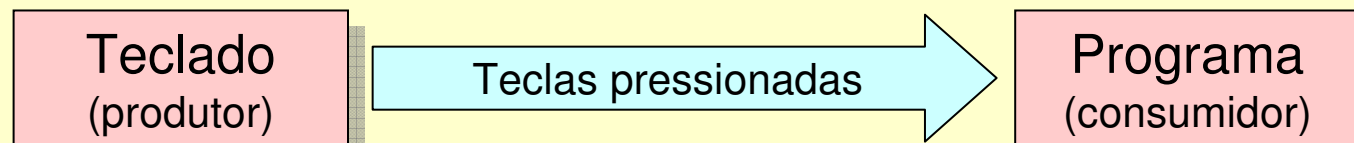


- Leitura e escrita

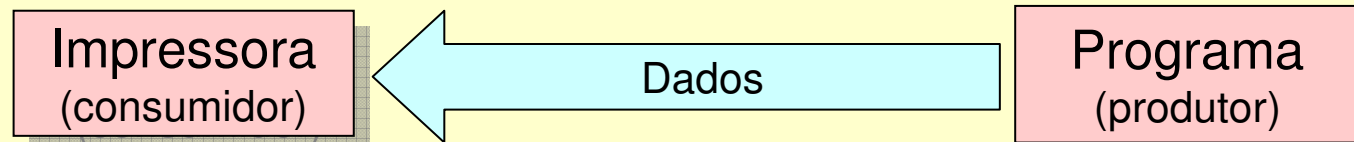


Fluxo de Dados: tipos

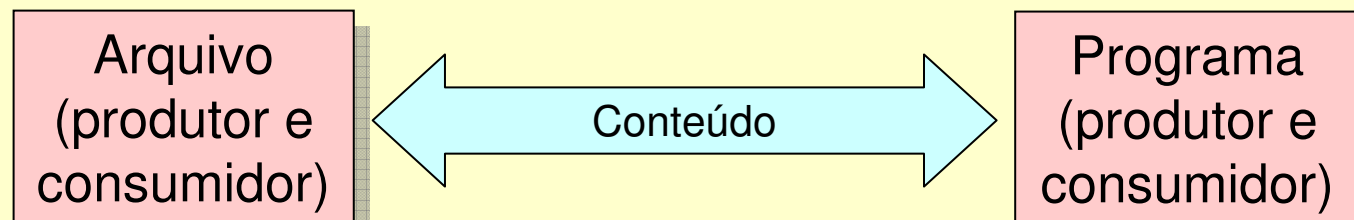
- Somente leitura



- Somente escrita



- Leitura e escrita



Fluxo de Dados: fluxos de leitura

- Posição de leitura:
 - Seqüencial ou aleatório

Fluxo de Dados: fluxos de leitura

- Posição de leitura:
 - Seqüencial ou aleatório
- Comprimento:
 - Limitado ou ilimitado
 - Marcas BOF & EOF

Fluxo de Dados: fluxos de leitura

- Posição de leitura:
 - Seqüencial ou aleatório
- Comprimento:
 - Limitado ou ilimitado
 - Marcas BOF & EOF
- Recebimento de dados:
 - Bloqueante ou não bloqueante

Fluxo de Dados: fluxos de escrita

- Posição de escrita:
 - Seqüencial ou aleatório

Fluxo de Dados: fluxos de escrita

- Posição de escrita:
 - Seqüencial ou aleatório
- Comprimento:
 - Praticamente ilimitado

Fluxo de Dados: fluxos de escrita

- Posição de escrita:
 - Seqüencial ou aleatório
- Comprimento:
 - Praticamente ilimitado
- Envio de dados:
 - Bloqueante ou não bloqueante

Fluxo de Dados: leitura e escrita

- Posição de leitura ou escrita:
 - Sempre aleatório

Fluxo de Dados: leitura e escrita

- Posição de leitura ou escrita:
 - Sempre aleatório
- Comprimento:
 - Praticamente ilimitado

Fluxo de Dados: leitura e escrita

- Posição de leitura ou escrita:
 - Sempre aleatório
- Comprimento:
 - Praticamente ilimitado
- Envio de dados:
 - Bloqueante ou não bloqueante

Fluxo de Dados: peculiaridades

- Fluxo binário

Fluxo de Dados: peculiaridades

- Fluxo binário
- Fluxo texto
 - Reconhecimento de '\n'
 - Traduções automáticas de símbolos
 - Tratamento automático de símbolo EOF

Fluxo de Dados: operações

1. Abrir (estabelecer) o fluxo

- Define operações permitidas
- Especifica tipo de fluxo (binário/texto)
- Tipo de acesso (seqüencial ou aleatório)

Fluxo de Dados: operações

1. Abrir (estabelecer) o fluxo
 - Define operações permitidas
 - Especifica tipo de fluxo (binário/texto)
 - Tipo de acesso (seqüencial ou aleatório)
2. Ler e/ou escrever dados

Fluxo de Dados: operações

1. Abrir (estabelecer) o fluxo
 - Define operações permitidas
 - Especifica tipo de fluxo (binário/texto)
 - Tipo de acesso (seqüencial ou aleatório)
2. Ler e/ou escrever dados
3. Fechar (terminar) o fluxo
 - Libera recursos
 - Permite uso do fluxo por outro programa

Comunicação e Arquivos

A background image showing a close-up of a traditional abacus with black beads on wooden rods. Two hands are visible, with fingers positioned to move the beads. The image is slightly faded and serves as a backdrop for the text.

Acesso a arquivos



Arquivos: tipos de dados

Declaração:

```
FILE * arquivo;
```

- Define um fluxo para leitura e escrita em arquivo.
- Cada variável declarada é fluxo **independente**
- Não é relevante como funciona o tipo `FILE *`

Arquivos: abrir e fechar

```
FILE *arquivo;  
arquivo = fopen(nome, modo);
```

Arquivos: abrir e fechar

```
FILE *arquivo;  
arquivo = fopen(nome, modo) ;
```

- Unidade de disco e diretório
- Caminho Relativo
- Nome do arquivo

Arquivos: abrir e fechar

```
FILE *arquivo;  
arquivo = fopen(nome, modo) ;
```

- Unidade de disco e diretório
- Caminho Relativo
- Nome do arquivo

“r” “r+”
“w” “w+”
“a” “a+”
“b”: binário
“t”: texto

Arquivos: abrir e fechar

```
FILE *arquivo;  
arquivo = fopen(nome, modo) ;
```

- Unidade de disco e diretório
- Caminho Relativo
- Nome do arquivo

```
fclose(arquivo) ;
```

“r” “r+”
“w” “w+”
“a” “a+”
“b”: binário
“t”: texto

Arquivos: abrir e fechar

```
FILE *arquivo;  
arquivo = fopen("alunos.txt", "r");  
...  
// Lê o nome de todos os alunos  
...  
fclose(arquivo);
```

Arquivos: leitura

```
FILE *arquivo;  
arquivo = fopen(nome, modo);  
  
fscanf(arquivo, "fomato", &variavel);
```

Arquivos: leitura

```
FILE *arquivo;  
arquivo = fopen(nome, modo);  
  
fscanf(arquivo, "fomato", &variavel);
```

Semelhante a
scanf

Arquivos: leitura

```
FILE *arquivo;  
arquivo = fopen(nome, modo);
```

```
fscanf(arquivo, "fomato", &variavel);
```

Semelhante a
scanf

%d, %f, %c, %s, etc

Arquivos: leitura

```
FILE *arquivo;  
arquivo = fopen(nome, modo);
```

```
fscanf(arquivo, "fomato", &variavel);
```

Semelhante a
scanf

%d, %f, %c, %s, etc

Lista de
variáveis

Arquivos: leitura

Leitura:

Arquivo:

José	9.5	8.5
Ana	7.0	8.0
Paulo	3.5	5.5

Arquivos: leitura

Leitura:

```
FILE *arquivo;  
char nome[30];  
float nota1, nota2;
```

Arquivo:

José	9.5	8.5
Ana	7.0	8.0
Paulo	3.5	5.5

```
arquivo = fopen("alunos.txt", "r");  
...  
// Lê o nome e nota do primeiro aluno  
fscanf(arquivo, "%s %f %f", nome, &nota1,  
        &nota2);  
...  
fclose(arquivo);
```


Arquivos: leitura

```
FILE *arquivo;  
char c, linha[102];  
arquivo = fopen(nome, modo);
```

Ler um caractere:

Ler uma linha:

Arquivos: leitura

```
FILE *arquivo;  
char c, linha[102];  
arquivo = fopen(nome, modo);
```

Ler um caractere: c = **fgetc(arquivo)** ;

Ler uma linha:

Arquivos: leitura

```
FILE *arquivo;  
char c, linha[102];  
arquivo = fopen(nome, modo);
```

Ler um caractere: `c = fgetc(arquivo);`

Ler uma linha: `fgets(linha, 100, arquivo);`



Arquivos: leitura

```
FILE *arquivo;  
char c, linha[102];  
arquivo = fopen(nome, modo);
```

Ler um caractere: `c = fgetc(arquivo);`

Ler uma linha: `fgets(linha, 100, arquivo);`

↓
variável para armazenar conteúdo

Arquivos: leitura

```
FILE *arquivo;  
char c, linha[102];  
arquivo = fopen(nome, modo);
```

Ler um caractere: `c = fgetc(arquivo);`

Ler uma linha: `fgets(linha, 100, arquivo);`

variável para armazenar conteúdo

Tamanho
máximo

Arquivos: leitura

Arquivo:
Prezado cliente,
Gostaríamos de...



Arquivos: leitura

Arquivo:
Prezado cliente,
Gostaríamos de...

```
FILE *arquivo;  
char linha1[102], linha2[102];  
  
arquivo = fopen("mensagem.txt", "r");  
...  
fgets(linha1, 100, arquivo);  
fgets(linha2, 100, arquivo);  
...  
fclose(arquivo);
```



Arquivos: leitura

Arquivo:
Prezado cliente,
Gostaríamos de...

```
FILE *arquivo;  
char linha1[102], linha2[102];  
  
arquivo = fopen("mensagem.txt", "r");  
...  
fgets(linha1, 100, arquivo);  
fgets(linha2, 100, arquivo);  
...  
fclose(arquivo);
```

linha1 ← "Prezado cliente,"
linha2 ← "Gostaríamos de..."

Arquivos: fim de arquivo

```
FILE *arquivo;  
arquivo = fopen("mensagem.txt", "r");  
...  
while (! eof(arquivo) ) {  
    ...  
    Operação de leitura  
    ...  
}  
...  
fclose(arquivo);
```

Arquivos: fim de arquivo

```
FILE *arquivo;
char nome[30];
float nota1, nota2;

arquivo = fopen("notas.txt", "r");
...
while (! eof(arquivo)) {
    q = fscanf(arquivo, "%s %f %f", nome, &nota1, &nota2);
    if (q == 0) break;
    ...
}
...
fclose(arquivo);
```

Arquivos: escrita

```
FILE *arquivo;  
arquivo = fopen(nome, modo);  
  
fprintf(arquivo, "texto", &variavel);
```

Arquivos: escrita

```
FILE *arquivo;  
arquivo = fopen(nome, modo);  
  
fprintf(arquivo, "texto", &variavel);
```

Semelhante a
printf

Arquivos: escrita

```
FILE *arquivo;  
arquivo = fopen(nome, modo);
```

```
fprintf(arquivo, "texto", &variavel);
```

Semelhante a
printf

%d, %f, %c, %s, etc

Arquivos: escrita

```
FILE *arquivo;  
arquivo = fopen(nome, modo);
```

```
fprintf(arquivo, "texto", &variavel);
```

Semelhante a
printf

Lista de
variáveis

%d, %f, %c, %s, etc

Arquivos: escrita

```
FILE *arquivo;  
char nome[30];  
float nota1, nota2;  
  
arquivo = fopen("alunos.txt", "w");  
...  
// Escreve nome e nota do primeiro aluno  
fprintf(arquivo, "%s %f %f", nome, nota1, nota2);  
...  
fclose(arquivo);
```

Arquivos: escrita

```
FILE *arquivo;  
arquivo = fopen(nome, modo) ;  
char c, texto[100];
```

Escrever um caractere:

Escrever um texto:

Garantir escrita no disco:

Arquivos: escrita

```
FILE *arquivo;  
arquivo = fopen(nome, modo) ;  
char c, texto[100];
```

Escrever um caractere: `fputc(c, arquivo);`

Escrever um texto:

Garantir escrita no disco:

Arquivos: escrita

```
FILE *arquivo;  
arquivo = fopen(nome, modo) ;  
char c, texto[100];
```

Escrever um caractere: `fputc(c, arquivo);`

Escrever um texto: `fputs(linha, arquivo);`

Garantir escrita no disco:

Arquivos: escrita

```
FILE *arquivo;  
arquivo = fopen(nome, modo);  
char c, texto[100];
```

Escrever um caractere: `fputc(c, arquivo);`

Escrever um texto: `fputs(linha, arquivo);`

Garantir escrita no disco: `fflush(arquivo);`



Arquivos: escrita

```
FILE *arquivo;  
  
arquivo = fopen("mensagem.txt", "w");  
  
...  
fprintf(arquivo, "Resultado da operacao:\n");  
// ou:  
fputs("Resultado da operacao:\n", arquivo);  
  
...  
fclose(arquivo);
```



Arquivos: deslocamentos

```
FILE *arquivo;  
arquivo = fopen(nome, modo) ;  
    obrigatório: "w+" ou "r+" ←
```

Voltar ao inicio do arquivo:

Consultar a posição atual:

Avançar/Retroceder:

Arquivos: deslocamentos

```
FILE *arquivo;  
arquivo = fopen(nome, modo);  
    obrigatório: "w+" ou "r+" ←
```

Voltar ao inicio do arquivo: `rewind(arquivo);`

Consultar a posição atual:

Avançar/Retroceder:



Arquivos: deslocamentos

```
FILE *arquivo;  
arquivo = fopen(nome, modo);  
    obrigatório: "w+" ou "r+" ←
```

Voltar ao início do arquivo: `rewind(arquivo);`

Consultar a posição atual: `p = ftell(arquivo);`

Avançar/Retroceder:

Arquivos: deslocamentos

```
FILE *arquivo;  
arquivo = fopen(nome, modo);  
    obrigatório: "w+" ou "r+" ←
```

Voltar ao início do arquivo: `rewind(arquivo);`

Consultar a posição atual: `p = ftell(arquivo);`

Avançar/Retroceder:

```
fseek(arquivo, deslocamento, referência);
```


Arquivos: deslocamentos

```
FILE *arquivo;  
arquivo = fopen(nome, modo);
```

obrigatório: "w+" ou "r+" ←

Voltar ao início do arquivo: `rewind(arquivo);`

Consultar a posição atual: `p = ftell(arquivo);`

Avançar/Retroceder:

```
fseek(arquivo, deslocamento, referência);
```

positivo: avança

negativo: retrocede ←

Arquivos: deslocamentos

```
FILE *arquivo;  
arquivo = fopen(nome, modo);
```

obrigatório: "w+" ou "r+" ←

Voltar ao inicio do arquivo: `rewind(arquivo);`

Consultar a posição atual: `p = ftell(arquivo);`

Avançar/Retroceder:

```
fseek(arquivo, deslocamento, referência);
```

positivo: avança

negativo: retrocede

SEEK_CUR, SEEK_END,
SEEK_SET

Comunicação e Arquivos

Entrada/Saída Padrão

Entrada/Saída Padrão

- Três arquivos abertos automaticamente:

Entrada/Saída Padrão

- Três arquivos abertos automaticamente:
 - **stdin**: entrada padrão (teclado)

Entrada/Saída Padrão

- Três arquivos abertos automaticamente:
 - **stdin**: entrada padrão (teclado)
 - **stdout**: saída padrão (terminal/tela DOS)

Entrada/Saída Padrão

- Três arquivos abertos automaticamente:
 - **stdin**: entrada padrão (teclado)
 - **stdout**: saída padrão (terminal/tela DOS)
 - **stderr**: saída de erro (terminal/tela DOS)

A hand is shown using an abacus, a traditional calculating tool with beads on rods. The abacus is dark-colored with light-colored beads. The hand is positioned on the left side of the frame, with fingers touching the beads. The background is a light blue gradient.

Entrada/Saída Padrão

- Três arquivos abertos automaticamente:
 - **stdin**: entrada padrão (teclado)
 - **stdout**: saída padrão (terminal/tela DOS)
 - **stderr**: saída de erro (terminal/tela DOS)
- Equivalentes:

```
printf("texto");  
fprintf(stdout, "texto");
```


Comunicação e Arquivos

FormatArq