MC102 - Algoritmos e programação de computadores

Aula 25: Arquivos

Arquivos

- Podem armazenar grande quantidade de informação
- Dados são persistentes (gravados em disco)
- Acessso aos dados pode ser não seqüencial (acesso direto a registros em um banco de dados)
- Acesso à informação pode ser concorrente (mais de um programa ao mesmo tempo)

Nomes e extensões

arq.txt	arquivo texto simples
arq.c	código fonte em C
arq.pdf	*portable document format*
arq.html	arquivo para páginas WWW *(hypertext markup language)*
arq*	arquivo executável (UNIX)

Tipos de arquivos

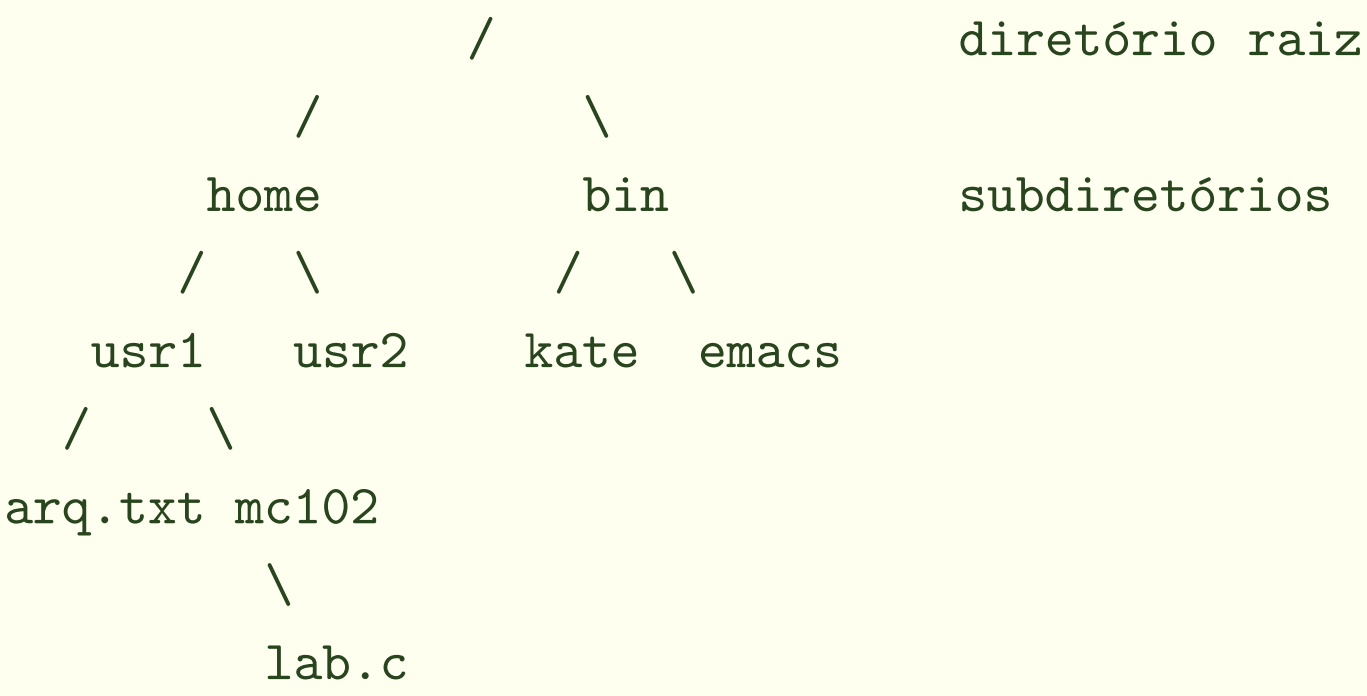
- Arquivo texto - armazena caracteres que podem ser mostrados diretamente na tela ou modificados por um editor de textos simples. Por exemplo:

  ```
  *** Arquivo texto. ***
  ```

- Arquivo binário - seqüência de bits sujeita às convenções dos programas que o gerou (exemplos arquivos executáveis, arquivos compactados, arquivos de registros).

Diretório

- também chamado de pasta
- contém arquivos e/ou outros diretórios



Caminhos absolutos ou relativos

- Caminho absoluto:
  - Descrição de um caminho desde o diretório raiz:

    `/bin/emacs`

    `/home/usr1/arq.txt`
- Caminho relativo:
  - Descrição de um caminho desde o diretório corrente:

    `arq.txt`

    `mc102/lab.c`
  - veja o diretório corrente com o comando `pwd`
  - mude de diretório com o comando `cd`

Atributos de arquivos

- nome do arquivo;
- proprietário do arquivo;
- datas de criação, alteração e acesso;
- tamanho em bytes;
- permissão de acesso.

- veja os comandos `ls -l` e `stat`

Permissão de acesso

```
islene@emilia% ls -l
-rw-r----- aula.txt
-rwxr-xr-x
drwxr-xr-x mc102/
```

- Três níveis de controle: proprietário - grupo - todos
- r: leitura
- w: escrita
- x:
  - execução para arquivos
  - permissão de entrada para diretórios

Abrindo um arquivo para leitura

- A chamada abaixo tenta abrir o arquivo `teste.txt`

```
if (fopen("teste.txt", "r") == NULL)
  perror("Erro ao abrir o arquivo.\n");
else
  printf("Arquivo aberto para leitura.\n");
```

- Em caso de erro:
  - a função retorna NULL
  - a função `perror` exibe uma mensagem explicativa
- Veja o código: `fopen-r.c`

Lendo dados de um arquivo

```
FILE *f = fopen ("teste.txt", "r");
while (fscanf(f, "%c", &c) != EOF)
  printf("%c", c);
fclose(f);
```

- fopen retorna um *stream pointer*
- fscanf é semelhante à função scanf
- fclose fecha o arquivo
- Veja o código: `fscanf.c`

Escrevendo dados em um arquivo

```
FILE *fr = fopen ("teste.txt", "r");
FILE *fw = fopen ("saida.txt", "w");
while (fscanf(fr, "%c", &c) != EOF)
  fprintf(fw,"%c", c);
fclose(fr);
fclose(fw);
```

- fprintf é semelhante à função printf
- Veja o código: `fprintf.c`

fopen

```
FILE* fopen(const char *caminho, char *modo);
```

modo	operações	ponto no arquivo
r	leitura	início
r+	leitura e escrita	início
w	escrita	início
w+	leitura e escrita	início
a	escrita	final
a+	leitura	início
	escrita	final

Lendo um vetor de um arquivo

```
FILE  *fr;
int i, n, *v;

fr = fopen ("v-in.txt", "r");
fscanf(fr, "%d", &n);  /* Dimensão do vetor */
v = (int *) malloc (n * sizeof(int));
for (i = 0; i < n; i++)
  fscanf(fr, "%d", &v[i]);
fclose(fr);
```

- Veja o código: `le_vetor.c`

Escrevendo um vetor em um arquivo

```
FILE  *fw = fopen ("v-out.txt", "w");
fprintf(fw, "%d\n", n); /* Dimensão do vetor */
for (i = 0; i < n; i++)
  fprintf(fw, "%d\n", v[i]);
fclose(fw);
```

- Veja o código: `le_vetor.c`

Lendo uma matriz de um arquivo

- Alocação dinâmica em um vetor linear de dimensão `nlin * ncol`

```
int *v = (int *)
         malloc (nlin * ncol * sizeof(int));
for (i = 0; i < nlin * ncol; i++)
  fscanf(fr, "%d", &v[i]);
```

- Veja o código: `le_matriz.c`

Escrevendo uma matriz em um arquivo

```
for (i = 0; i < nlin; i++)
  for (j = 0; j < ncol; j++)
   fprintf(fw, "mat[%d][%d] = %d\n",
           i, j, v[i*ncol + j]);
```

- `mat[i][j] = v[i*ncol + j]`
- Veja o código `le_matriz.c`

Lendo uma matriz de um arquivo

- Alocação dinâmica em nlin vetores de ncol inteiros

```
int **v = (int **) malloc(nlin * sizeof(int*));
for (i = 0; i < nlin; i++)
  v[i] = (int *) malloc(ncol * sizeof(int));
for (i = 0; i < nlin; i++)
  for (j = 0; j < ncol; j++)
    fscanf(fr, "%d", &v[i][j]);
```

- Veja o código `le_matriz2.c`

Argumentos para o main

```
int main (int argc, char* argv[]) {
  FILE *fr, *fw
  if (argc < 3) {
    printf("Uso: %s <origem> <destino>\n",
            argv[0]);
    return 1;
  }
  fr = fopen (argv[1], "r");
  fw = fopen (argv[2], "w");
```

- Veja o código: `cp.c`