arquivos

INF1005 -- Programação I -- 2016.1
Prof. Roberto Azevedo
razevedo@inf.puc-rio.br

arquivos

tópicos

- por que usar arquivos
- manipulação de arquivos
- funções para manipulação de arquivos em C

referência

- Capítulo 6 da apostila
- Capítulo 15 do livro

por que utilizar arquivos?

armazenamento permanentemente dados

e.g., que tenham sido digitados por alguém ou que estejam em arquivos num formato diferente do desejado

tratar uma grande quantidade de dados

- e.g., quando temos muitos dados não é conveniente digitá-los toda vez que o programa executar; também não é conveniente exibir uma grande quantidade de dados na tela.

processar dados de diferentes maneiras (transformando-os, combinando-os, gerando novos etc.)

e.g., a partir das notas dos alunos de uma turma calcular as suas médias (e, possivelmente, gerar um novo arquivo)

por que utilizar arquivos?

arquivo de entrada

se um programa precisa carregar uma grande quantidade de dados é mais adequado que o programa *carregue* (leia) esses dados de um arquivo

arquivo de saída

de forma análoga, se um programa precisa exibir uma grande quantidade de dados é mais adequado que o programa *salve* (escreva) esses dados em um arquivo

tipos de arquivos em C

arquivos de texto

- basicamente, uma sequência de caracteres
- normalmente, organizado em linhas terminadas por caractere de nova linha

arquivos binários

- uma sequência de bytes diretamente correspondida com o dispositivo externo, isto é, sem tradução dos caracteres
- **não será abordado neste curso!**

como manipular arquivos

leitura

- abrir o arquivo para leitura (*read*)
- ler os dados no formato em que foram escritos
- Fechar o arquivo

escrita

- abrir o arquivo para escrita (write)
- escrever os dados no formato desejado
- fechar o arquivo

como manipular arquivos?

Para que programas possam ler e salvar dados em um arquivo, é necessário ter acesso a um conjunto de funções que permitam realizar essas operações

<stdio.h>

- no nosso curso, usaremos um conjunto de funções presentes na biblioteca **<stdio.h>**
- **fopen, fscanf, fprintf, fclose, fgetc, fgets, fputs**

leitura de arquivo

fopen: abrindo arquivo para leitura

```
#include <stdio.h>                          /* biblioteca de entrada e saída */
...
FILE* fpIn;                                 /* ponteiro para estrutura de arquivo */
...
fpIn = fopen("arquivoEntrada.txt", "r");    /* abre arquivo para leirutra (r: read) */
...
/* lê e processa dados */
fclose (fpIn);                              /* fecha arquivo */
```

Se o arquivo não existir ou não puder ser aberto para leitura **fopen** retorna **NULL**.

fopen: abrindo arquivo para leitura

```c
#include <stdio.h> /* biblioteca de funções de entrada e saída */
#include <stdlib.h> /* biblioteca padrão */

int main(void)
{
   FILE* fpIn; /* ponteiro para estrutura de arquivo */

   fpIn = fopen ("arquivoEntrada.txt", "r");  /* abre arquivo (que deve existir) para leitura (r: read)*/
   if (fpIn == NULL) /* verifica se arquivo pôde ser aberto */
   {
      printf ("Erro na abertura do arquivo de entrada.\n");
      exit(1); /* aborta o programa (função da biblioteca padrão) */
   }
   ... /* lê e processa conteúdo do arquivo */
   fclose (fpIn); /* fecha arquivo */
   return 0;
}
```

Se o arquivo não existir ou não puder ser aberto para leitura **fopen** retorna **NULL**.

fscanf: lendo dados de arquivo

```c
#include <stdio.h>
int main(void) {
   FILE* fpIn;
   float p1, p2, p3;
   int mat;
   fpIn = fopen ("notas.txt", "r"); /* abre para leitura */
   if (fpIn == NULL) {  /* verifica se arquivo foi aberto */
      printf("Erro.\n");
      return 1;
   }

   /* lê número de matrícula e notas de cada aluno do arquivo de entrada */
   /* (no final do arquivo, fscanf retorna EOF (-1)) */
   while ( fscanf (fpIn, "%d %f %f %f ", &mat, &p1, &p2, &p3) == 4 )
      /* escreve matrícula e média de cada aluno na tela */
      printf("%d %.1f\n", mat, (p1+p2+p3)/3);

   /* fecha arquivo de entrada */
   fclose (fpIn);
   return 0;
}
```

exercício 01

modifique o programa anterior para ler 3 ou 4 notas, dependendo da média das 3 primeiras, e exibir a média. Lembrando: caso o aluno precise fazer P4, a média final é a média aritmética das suas 3 maiores.

exercício 01

modifique o programa anterior para ler 3 ou 4 notas, dependendo da média das 3 primeiras, e exibir a média. Lembrando: caso o aluno precise fazer P4, a média final é a média aritmética das suas 3 maiores.

Como ler três valores e depois mais um, se necessário?

1. lê três valores
2. calcula a média
3. se a média < 5.0, lê quarto valor

```
while (fscanf(fpIn, "%d %f %f %f ", &mat, &p1, &p2, &p3) == 4) {
   media = (p1+p2+p3)/3;
   if (media < 5.0) {
      fscanf(fpIn, "%f ", &p4);
      /* calcula a nova média */
      ...
   }
   printf("%d %.1f\n", mat, media);
}
```

exercício 01

modifique o programa anterior para ler 3 ou 4 notas, dependendo da média das 3 primeiras, e exibir a média. Lembrando: **caso o aluno precise fazer P4, a média final é a média aritmética das suas 3 maiores**.

Como calcular a nova média (com a P4)?

1. Que tal: somar todas as quatro notas (P1 a P4), e depois subtrair a menor das outras três?
2. Como descobrir a menor de três?
   1. Problema mais simples: descobrir a menor de duas notas

```
float menor (float a, float b)
{
   if (a < b)
      return a;
   return b;
}
```

   2. Usando a função acima, come descobrir a menor de três notas?

```
nota_menor = menor(P1, menor(P2, P3));
```

exercício 01

```c
#include <stdio.h>

int main(void) {
   FILE* fpIn;
   float p1, p2, p3, p4, media;
   int mat;
   fpIn = fopen("notas.txt", "r");
   if (fpIn == NULL) { printf("Erro.\n"); return 1; }
   while (fscanf (fpIn, "%d %f %f %f ", &mat, &p1, &p2, &p3) == 4) {
      media = (p1+p2+p3)/3;
      if (media < 5.0) {
         fscanf(fpIn, "%f ", &p4);
         media = (p1 + p2 + p3 + p4 – menor(p1, menor(p2, p3)))/3;
      }
      printf("%d %.1f\n", mat, media);
   }
   fclose(fpIn);
   return 0;
}
```

escrita em arquivo

fopen: abrindo arquivo para escrita

```c
#include <stdio.h> /* biblioteca de funções de entrada e saída */
#include <stdlib.h> /* biblioteca padrão */

int main(void)
{
   FILE* fpIn; /* ponteiro para estrutura de arquivo */

   fpIn = fopen ("arquivoSaida.txt", "w");  /* abre arquivo (que deve existir) para leitura (r: read)*/
   if (fpIn == NULL) /* verifica se arquivo pôde ser aberto */
   {
      printf ("Erro na abertura do arquivo de saida.\n");
      exit(1); /* aborta o programa (função da biblioteca padrão) */
   }
   ... /* lê e processa conteúdo do arquivo */
   fclose (fpIn); /* fecha arquivo */
   return 0;
}
```

Se o arquivo não existir ou não puder ser aberto para leitura **fopen** retorna **NULL**. Se arquivo já existir, arquivo será sobrescrito.

fprintf: escrevendo dados em arquivo

```c
#include <stdio.h>

int main(void) {
   FILE *fpOut;
   float p1, p2, p3;
   int mat;
   fpOut = fopen("notas.txt", "w"); /* abre para escrita*/
   if (fpOut == NULL) { /* verifica se arquivos puderam ser abertos */
      printf("Erro.\n");
      return 1;
   }
   while (1) {
      printf("Digite o numero de matricula (0 para terminar): ");
      scanf("%d", &mat);
      if (mat <= 0) break;
      printf("Digite as notas da P1, P2 e P3: ");
      scanf("%f %f %f", &p1, &p2, &p3);
      /* escreve os dados de cada aluno no arquivo de saída */
      fprintf (fpOut, "%d %.1f %.1f %.1f\n", mat, p1, p2, p3);
   }
   fclose (fpOut); /* fecha arquivo de saída */
   return 0;
}
```

modos de abertura de arquivos

Modo	Significado	Descrição	Ação caso o arquivo exista	Ação caso o arquivo não exista
"r"	leitura (read)	Abre um arquivo de texto para leitura	lê do início	falha ao abrir (retorna NULL)
"w"	escrita (write)	Cria um arquivo de texto para escrita	destrói o conteúdo	cria novo
"a"	anexar (append)	Adiciona texto ao fim de um arquivo texto	escreve no final	cria novo
"r+"	leitura extendida (read extended)	Abre um arquivo texto para leitura/escrita	lê do início	erro
"w+"	escrita extendida (write extended)	Cria um arquivo texto para leitura/escrita	destrói o conteúdo	cria novo
"a+"	anexar extendido (append extended)	Anexa ao final ou cria um arquivo para leitura/escrita	escreve no final	cria novo

Se um arquivo existente for aberto para **escrita**, **será apagado e um novo será criado**. Se for aberto para **leitura/escrita**, ele **não será apagado**. E, caso ele **não exista**, então **será criado**.