Engenharia de Computação

Fundamentos de Computação

Aula 15 – Arquivos (fgets,fputs,fwrite,fread)

Prof. Fernando Barreto
informatica-ap@utfpr.edu.br

Arquivo Texto: Operações

- Escrever vetor de char em formato string no arquivo

  int fputs(*char *str, FILE *ponteiro_arquivo***)**

  - OBS: O caracter '\0' não é inserido no arquivo
  - Retorna:
    - EOF, se houver erro de escrita.
    - Valor != 0, indicando sucesso na escrita
  - Parâmetros:
    - *char *str* é uma string a ser escrita no arquivo
    - *FILE *ponteiro_arquivo* é o ponteiro para o arquivo aberto

Arquivo Texto: Operações

- Ler 1 vetor de char do arquivo aberto, para gerar uma string

char *fgets(*char *str, int tam, FILE *ponteiro_arquivo***)**

  - Lê-se tam-1 caracteres em *char *str*, ou até achar um '\n' primeiro. Se houver '\n', ele é inserido na string e interrompe o fgets(). A função finaliza inserindo '\0' automaticamente no vetor de char.
  - Retorna:
    - NULL, se houver erro de leitura ou final do arquivo atingido
    - O ponteiro str, se houver sucesso.
  - Parâmetros:
    - *char *str* é uma string a ser gerada
    - *int tam* é o limite máximo de chars a serem lidos
    - *FILE *ponteiro_arquivo* é o ponteiro para o arquivo aberto

Exercícios

- Usando as funções aprendidas até agora:
  - 2_1) Faça uma função que receba uma matriz de chars contendo 2 strings e crie um arquivo a partir dessas strings. Cada string deve ficar em uma linha no arquivo.
  - 2_2) Faça outra função que receba por parâmetro outra matriz de 2 strings, leia o arquivo gerado na função anterior para preencher as 2 strings. Mostre as duas strings na tela.

Arquivo Binário: Operações

- Abrir arquivo com modo binário: "rb" ou "wb"
- Escrever blocos de bytes genéricos

int fwrite(*void *buf, int tam_item, int nro_itens, FILE *ptr_arquivo***)**

  - Retorna:
    - Um valor < *nro_itens*, se houver erro de escrita
    - O valor de *nro_itens*, se houver sucesso.
  - Parâmetros:
    - *void *buf* ponteiro para a região de memória onde estão os items/blocos
    - *int tam_item* é o tamanho em bytes de cada item/bloco apontado por *buf a ser escrito no arquivo
      - » Geralmente usa-se *sizeof(tipo_usado_pelo_item)*
    - int nro_itens é a quantidade de items/blocos a ser escrito
      - » O total de bytes escritos será *tam_item*nro_itens*
    - *FILE *ptr_arquivo* é o ponteiro para o arquivo aberto

Arquivo Binário: Operações

- Ler blocos de bytes genéricos

int fread(*void *buf, int tam_item, int nro_itens, FILE *ptr_arquivo***)**

  - Retorna:
    - Um valor < *nro_itens*, se houver erro de leitura
    - O valor de *nro_itens*, se houver sucesso.
  - Parâmetros:
    - *void *buf* ponteiro para a região de memória onde serão armazenados os items/blocos
    - *int tam_item* é o tamanho em bytes de cada item/bloco de *buf a ser lido do arquivo
      - » Geralmente usa-se *sizeof(tipo_usado_pelo_item)*
    - int nro_itens é a quantidade de items/blocos a ser lido
      - » O total de bytes lidos será *tam_item*nro_itens*
    - *FILE *ptr_arquivo* é o ponteiro para o arquivo aberto

Exercícios

- 3_1) Faça um função que receba uma matriz 4x4 de float e a escreva em arquivo usando fwrite.
- 3_2) Faça outra função que receba outra matriz 4x4 de float e a preencha a partir do arquivo gerado acima.
  - Dica: para escrever/ler matriz no arquivo, escreva/leia considerando uma linha como sendo um bloco da matriz a ser escrito no arquivo por vez.
- 4_1) Faça uma função que receba um vetor de 3 structs, já preenchido com as informações abaixo. Grave esse vetor em arquivo.

Nome: Zezao Idade: 58 Salario: 1202.10	Nome: Deputado Idade: 51 Salario: 35000.50	Nome: Dick Vigarista Idade: 79 Salario: 600.00

- 4_2) Faça outra função que receba outro vetor de struct Y para ser preenchido com as informações do arquivo gerado no exercício anterior