UNIVERSIDADE VEIGA DE ALMEIDA – UVA

CURSO: ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

DISCIPLINA: ESTRUTURA DE DADOS (IL10301)

TUTOR: LUCIANO DE PINNA VIEIRA

ALUNO: JOÃO PAULO DANTAS

MATRÍCULA: 1220303425

Trabalho da Disciplina

Estrutura de Dados (IL10301)

AVA 1

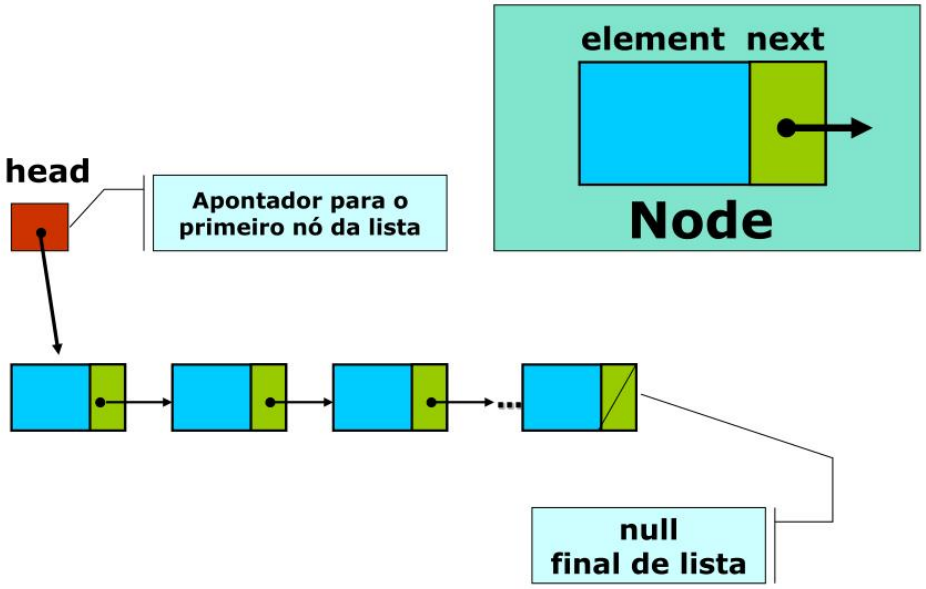
Estrutura de dados lista

A lista linear é uma estrutura de dados em que todos os elementos são armazenados de forma sequencial, e seu encadeamento pode ocorrer de forma simples ou dupla.

Você é programador de computador em linguagem C da empresa Renalf Mega Data e precisa realizar as operações elementares com os dados de uma lista simplesmente encadeada, visando à listagem de todos os dados armazenados nessa estrutura de dados.

Procedimentos para elaboração do TD

Desenvolva um programa de computador que permita ao usuário realizar as operações elementares – inclusão, consulta, alteração e remoção – com a estrutura de dados lista simplesmente encadeada.



lista simplesmente encadeada

Uma lista encadeada é uma representação de uma sequência de objetos, todos do mesmo tipo, na memória RAM (= random access memory) do computador. Cada elemento da sequência é armazenado em uma célula da lista: o primeiro elemento na primeira célula, o segundo na segunda, e assim por diante.

*na aplicação, os comentário // é porquê estão em uma linha só.

Foi usada a IDE Code::Blocks no Linux Fedora 39

// arquivo-cabeçalho

#include <stdio.h> // diz ao compilador que ele deve incluir esse arquivo-cabeçalho onde existem declarações de funções úteis para entrada e saída de dados padronizadas.
#include <stdlib.h> // diz ao compilador que ele deve incluir esse arquivo-cabeçalho que possui funções envolvendo alocação de memória, controle de processos, conversões e outras.

// usando a struct pessoa para armazenar o nome e a idade de uma pessoa

struct pessoa { // define uma estrutura chamada "pessoa". desta forma criamos uma variável composta chamada "pessoa" com quatro variáveis dentro e assim, formando um objeto.
  int id; // declara uma variável do tipo int com o nome "id".
  char nome[100]; // declara uma variável do tipo char com o nome "nome" e um tamanho e com até 100 caracteres.
  int idade; // declara uma variável do tipo int com o nome "idade".
  struct pessoa *prox; // essa linha declara a variável do tipo ponteiro "prox" para armazenar o ponteiro para a próxima "pessoa" da lista.
};

// ponteiro para a fila de "pessoas" e a inicializa com o valor NULL

struct pessoa *inicio = NULL; // a variável "inicio" é um ponteiro para a estrutura "pessoa" e a inicializa com o valor NULL. O valor NULL é um ponteiro especial que indica que o ponteiro não aponta para nenhum objeto.

// função para inserir uma "pessoa" na fila

void inserir(int id, char *nome, int idade) { // é um bloco de código chamado "inserir", que recebe três parâmetros e não retorna nenhum valor, ele será usado para inserir um "objeto pessoa" na lista.
  struct pessoa *novo = malloc(sizeof(struct pessoa)); // aloca memória para uma nova estrutura "pessoa". O tamanho da estrutura "pessoa" é definido pela constante: sizeof(struct pessoa).
  novo->id = id; // essa linha inicializa a variável "id" da estrutura "pessoa" com o valor fornecido pelo parâmetro "id".
  strcpy(novo->nome, nome); // strcpy() é uma função que copia uma string de um local para outro. ela copia a string fornecida pelo parâmetro "nome" para a variável "nome" da estrutura "pessoa".
  novo->idade = idade; // inicializa a variável "idade" da estrutura "pessoa" com o valor fornecido pelo parâmetro "idade".
  novo->prox = inicio; // essa linha atribui o endereço da estrutura "pessoa" anterior a variável "prox" da nova estrutura "pessoa". Isso significa que a nova estrutura "pessoa" será a próxima estrutura "pessoa" da lista.

inicio = novo; // essa linha atribui o endereço da nova estrutura “pessoa” à variável “inicio”. Isso significa que a nova estrutura “pessoa” será a primeira estrutura “pessoa” na lista.
}

// função para consultar uma “pessoa” da fila

void consultar(int id) { // é um bloco de código chamado ""consultar", que recebe um parâmetro e não retorna nenhum valor, ele será usado para consultar uma ”pessoa" da lista.
struct pessoa *aux = inicio; // essa linha inicializa a variável “aux” com o endereço da primeira estrutura “pessoa” da lista.
while (aux != NULL) { // essa linha inicia um loop while que continuará executando enquanto a variável ”aux" não for NULL.
if (aux->id == id) { // verifica se o "id" da estrutura “pessoa" atual é igual ao ”id" fornecido pelo parâmetro “id”.
printf("ID: %d\nNome: %s\nIdade: %d\n----------\n", aux->id, aux->nome, aux->idade); // essa linha imprime os dados da estrutura “pessoa” atual.
return; // encerra a execução de uma função e retorna o controle para a função de chamada.
}
aux = aux->prox; // essa linha avança para a próxima estrutura “pessoa” na lista..
}
printf("Não foi encontrado nenhum registro com o ID %d.\n", id); // essa linha imprime uma mensagem de erro se a “pessoa” com o "id" fornecido não for encontrado.
}

// função para alterar os dados de uma “pessoa” na fila

void alterar(int id, char *nome, int idade) { // é um bloco de código chamado "alterar", que recebe três parâmetro e não retorna nenhum valor, ele será usado para alterar os valores de uma “pessoa” da lista.
struct pessoa *aux = inicio;
while (aux != NULL) {
if (aux->id == id) {
strcpy(aux->nome, nome); // essa linha copia a string fornecida pelo parâmetro ”nome" para a variável ”nome" do objeto “pessoa” atual.
aux->idade = idade; // essa linha altera o valor da variável “idade” do objeto “pessoa” atual para o valor fornecido pelo parâmetro “idade".
return;
}
aux = aux->prox;
}
printf("Não foi encontrado nenhum registro com o ID %d.\n", id);
}

//função para tirar uma “pessoa” da fila

void remover(int id) { // é um bloco de código chamado "remover”, que recebe um parâmetro e não retorna nenhum valor, ele será usado para excluir um objeto “pessoa” da lista.
  struct pessoa *aux = inicio;
  struct pessoa *ant = NULL; // essa linha inicializa a variável “ant” com o valor NULL. A variável “ant” é usada para armazenar o endereço da “pessoa” anterior à “pessoa” atual.
  while (aux != NULL) {
    if (aux->id == id) {
      if (ant == NULL) { // essa linha verifica se a variável “ant” é igual a NULL.
        inicio = aux->prox; // essa linha altera o endereço da primeira “pessoa” da lista para o endereço da próxima “pessoa” da lista.
      } else {
        ant->prox = aux->prox; // essa linha altera o endereço da pessoa “anterior” para o endereço da próxima “pessoa” da lista.
      }
      free(aux); // essa linha libera a memória alocada para a pessoa atual.
      return;
    }
    ant = aux;
    aux = aux->prox;
  }
  printf("Não foi encontrado nenhum registro com o ID %d.\n", id);
}

// função para listar a fila

void listar() { // é um bloco de código chamado "estrutura pessoa”listar"estrutura pessoa”, que recebe nenhum parâmetro e não retorna nenhum valor, ele será usado para listar as "estrutura pessoa”pessoas"estrutura pessoa” da lista encadeada pessoa.
  struct pessoa *aux = inicio;
  while (aux != NULL) {
    printf("ID: %d\nNome: %s\nIdade: %d\n----------\n", aux->id, aux->nome, aux->idade);

 // essa linha imprime os objetos “pessoa” da lista.
    aux = aux->prox;
  }
}

// função principal

int main() { // é a função principal é responsável por iniciar a execução do programa e fornecer um ponto de entrada para o código do programa.
  int opcao, id, idade; // declaração das variáveis do tipo int;
  char nome[100]; // declaração das variáveis do tipo char.

```
do { // é o início de uma estrutura de repetição do-while

// lista dos valores para o usuário escolher uma opção digitando o número correspondente
   printf("1. Inserir\n2. Consultar\n3. Alterar\n4. Remover\n5. Listar\n0. Sair\n"); // printf é uma função da biblioteca padrão que é usada para imprimir dados na tela.
   printf("Digite uma opção: ");
   scanf("%d", &opcao); // scanf é uma função da biblioteca padrão que é usada para ler dados do teclado.

   switch (opcao) { // switch é o início de uma estrutura de controle switch. A estrutura switch é um tipo de estrutura de controle que é usada para escolher entre várias opções.

//valor 1 é para inserir uma "pessoa" na fila
     case 1: // o rótulo case 1 é usado para indicar que o bloco de código associado será executado se a expressão de seleção da estrutura switch for igual a 1.
       printf("Numero identificador da pessoa: %d\n", id++); // atribuir automaticamente um valor inteiro na variável id.
       printf("Nome: ");
       scanf("%s%*[^\n]",&nome); // %s%*[^\n] -  significa ler uma string de caracteres e descartar qualquer quebra de linha existente e armazenar na variável apontada com o nome de "nome".
       printf("Idade: ");
       scanf("%d", &idade); // significa ler um valor inteiro digitado e armazenar na variável apontada com o nome de "idade".
       inserir(id, nome, idade); // chamar a função "inserir" passando os três valores que estão guardados nas variáveis id, nome e idade.
       break;

//valor 2 é para consultar uma "pessoa" da fila
     case 2:
       printf("Digite o numero identificador da pessoa: ");
       scanf("%d", &id); /* ler um valor digitado e armazenar na variável apontada com o nome de "id". */
       consultar(id); // chamar a função "consultar" passando o valor que está guardado na variável "id".
       break;

//valor 3 é para alterar os dados de uma "pessoa" na fila
     case 3:
       printf("Digite o numero identificador da pessoa que voce quer alterar: ");
       scanf("%d", &id);
       printf("Digite o novo nome: ");
```

```
        scanf("%s%*[^\n]",&nome);
        printf("Digite a nova idade: ");
        scanf("%d", &idade);
        alterar(id, nome, idade); // chamar a função “alterar” passando os valores que estão guardados nas variáveis “id”, “nome” e “idade”.
        break;

//valor 4 é para tirar uma “pessoa” da fila
      case 4:
        printf("digite o numero identificador da pessoa que deseja excluir: ");
        scanf("%d", &id);
        remover(id);
        break;

//valor 5 é para listar a fila
      case 5:
        listar(); // chamar a função listar sem fornecer nenhum parâmetro.
        break;

//valor 0 é para sair do aplicativo
      case 0:
        break;

//qualquer valor diferente de: 0, 1, 2, 3, 4 e 5 é considerado como inválido
      default:
        printf("Opção inválida.\n");
        break;
    }
  } while (opcao != 0); // enquanto for diferente de 0;

  return 0;
}

//fim do programa
```

Referência:

- https://www.inf.ufpr.br/silvia/oficinac/aulas/ES.pdf
- http://www.univasf.edu.br/~fabio.nelson/arq/alg/Linguagem_C_UFMG.pdf
- https://pt.wikipedia.org/wiki/Stdlib.h
- http://mtm.ufsc.br/~azeredo/cursoC/aulas/c220.html
- https://blog.pantuza.com/artigos/tipos-abstratos-de-dados-lista-encadeada-linked-list
- https://cursos.alura.com.br/forum/topico-lista-encadeada-inserir-no-meio-149294
- https://www.ime.usp.br/~pf/algoritmos/aulas/fila.html
- https://www.ufsm.br/pet/sistemas-de-informacao/2020/04/01/entendendo-listas-pilhas-e-filas
- https://www.ime.usp.br/~pf/algoritmos/aulas/lista.html
- https://www.inf.pucrs.br/~pinho/LaproI/ComandosDeRepeticao/Repeticao.html
- https://www.cprogressivo.net/2013/03/Lendro-e-Escrevendo-Strings-em-C.html#:~:text=Por%20exemplo%2C%20se%20quisermos%20ler,dizer%20isso%20dentro%20da%20fun%C3%A7%C3%A3o.&text=Logo%2C%20nosso%20c%C3%B3digo%20da%20scanf,__fpurge(stdin)%20
- https://acervolima.com/recebendo-entrada-de-string-com-espaco-em-c-4-metodos-diferentes/
- https://pt.stackoverflow.com/questions/9427/limpeza-do-buffer-do-teclado-ap%C3%B3s-scanf
- PPT - Listas Simplesmente Encadeadas PowerPoint Presentation, free download - ID:5922983 (slideserve.com)