ACH2023 - Algoritmos e Estruturas de Dados I

Apresentação da disciplina

Prof. Flávio Luiz Coutinho
flcoutinho@usp.br

Objetivos

Capacitar o aluno a resolver problemas envolvendo as estruturas de dados básicas de **memória principal**, assim como discernir **qual a melhor estrutura para um determinado problema** no contexto em que ele se insere.

Programa

- Listas lineares
- Listas ordenadas e circulares
- Pilhas, filas e deques
- Listas duplamente encadeadas
- Árvores, árvores binárias, árvores de busca, árvores balanceadas (AVL)

Pré-requisitos

Introdução à Programação:

- Lógica de programação, resolução de problemas através de algoritmos.

- Linguagem C (estrutura de um programa, variáveis e tipos de variáveis, condicionais, laços, funções, vetores, matrizes, ponteiros, *structs*).

Pré-requisitos

Introdução à Análise de Algoritmos (cursada ao mesmo tempo que AED1):

- Recursão

- Ferramentas matemáticas para prever o consumo de recursos por um algoritmo (tempo, memória).

- Notações assintóticas (descrevem o comportamento de algoritmos)

Avaliação

- Provas (2 provas)
- Exercícios-Programa (quantidade a definir)
- Nota final = 0.7 MP + 0.3 MEP

Alguns pontos importantes:

- Linguagem de programação: C
- Ferramentas de desenvolvimento: um editor simples de texto e o gcc bastam
- Sistema operacional: qualquer um que execute as ferramentas acima, mas os EPs serão corrigidos no sistema Linux.

- eDisciplinas: avisos, material, entregas de tarefas, fórum de discussão.

Conceitos: estruturas de dados

Uma estrutura de dados:

Conceitos: estruturas de dados

Uma estrutura de dados:

- Armazena uma coleção informações

Conceitos: estruturas de dados

Uma estrutura de dados:

- Armazena uma coleção informações
- Organiza a informação armazenada

Conceitos: estruturas de dados

Uma estrutura de dados:

- Armazena uma coleção informações
- Organiza a informação armazenada
- Define operações que podem ser realizadas:

Conceitos: estruturas de dados

Uma estrutura de dados:

- Armazena uma coleção informações
- Organiza a informação armazenada
- Define operações que podem ser realizadas:
    - adição de nova informação

Conceitos: estruturas de dados

Uma estrutura de dados:

- Armazena uma coleção informações
- Organiza a informação armazenada
- Define operações que podem ser realizadas:
    - adição de nova informação
    - busca por uma informação específica

Conceitos: estruturas de dados

Uma estrutura de dados:

- Armazena uma coleção informações
- Organiza a informação armazenada
- Define operações que podem ser realizadas:
  - adição de nova informação
  - busca por uma informação específica
  - percurso pela informação armazenada

Conceitos: estruturas de dados

Uma estrutura de dados:

- Armazena uma coleção informações
- Organiza a informação armazenada
- Define operações que podem ser realizadas:
    - adição de nova informação
    - busca por uma informação específica
    - percurso pela informação armazenada
    - remoção de informação.

Conceitos: estruturas de dados

Cada estrutura diferente que vamos aprender implementa uma organização diferente da informação armazenada, visando vantagens específicas em algumas operações.

Conceitos: estruturas de dados

Cada estrutura diferente que vamos aprender implementa uma organização diferente da informação armazenada, visando vantagens específicas em algumas operações.

Difícil dizer que existe uma estrutura ideal, que é sempre a melhor em qualquer situação.

Sobre a linguagem C

Faremos uma revisão, nas primeiras aulas, sobre alguns pontos importantes da linguagem C:

- Vetores (*arrays*)
- Ponteiros
- Alocação dinâmica de memória
- Structs

Que serão fundamentais para o estudo/implementação das estruturas de dados a serem estudadas.

Exemplo (motivação)

Mas antes vamos discutir um exemplo simples, que serve de motivação para entender como diferentes formas de organização da informação podem impactar nas operações que podem ser realizadas sobre a informação armazenada.

Conhecimento prévio

Com o conhecimento adquirido em IP, os recursos que temos para armazenar e organizar informação são basicamente:

- uso de variáveis “simples”: `int a, b, c;`
- uso de vetores (*arrays*): `int v[100];`

Conhecimento prévio

Com o conhecimento adquirido em IP, os recursos que temos para armazenar e organizar informação são basicamente:

- ~~uso de variáveis “simples”:~~ ~~`int a, b, c;`~~
- uso de vetores (*arrays*): `int v[100];`

Conhecimento prévio

Com o conhecimento adquirido em IP, os recursos que temos para armazenar e organizar informação são basicamente:

- ~~uso de variáveis “simples”:~~ ~~`int a, b, c;`~~
- uso de vetores (*arrays*): `int v[100];`

Apesar de rudimentar, podemos dizer que vetores correspondem a um tipo de estrutura de dados.

Exemplo (motivação)

Partindo daquilo que já conhecemos, vamos pensar em como podemos usar um vetor para representar uma coleção de valores inteiros positivos, e duas operações básicas: adição de um novo valor, e busca por um valor específico.

Exemplo (motivação): abordagem 1

Organização/armazenamento da informação:

```
int v[100];

int livre = 0;
```

Exemplo (motivação): abordagem 1

Adição do valor x:

```
if(livre < 100) {
    v[livre] = x;
    livre++;
}
```

Exemplo (motivação): abordagem 1

Busca pelo valor x:

```
int i;

for(i = 0; i < livre; i++) {

    if(v[i] == x) return 1;

}

return 0;
```

Exemplo (motivação)

Funciona? Sim, funciona…

Exemplo (motivação)

Funciona? Sim, funciona…

É a única forma de se representar uma coleção de inteiros? Não…

Exemplo (motivação)

Funciona? Sim, funciona…

É a única forma de se representar uma coleção de inteiros? Não…

Vamos dar uma olhada em outra alternativa:

Exemplo (motivação): abordagem 2

Organização/armazenamento da informação:

```
int v[VALOR_MAXIMO + 1];

// inicializa o vetor, tal que v[i] = 0 para todo i.
```

Exemplo (motivação): abordagem 2

Adição do valor x:

```
v[x]++;
```

Exemplo (motivação): abordagem 2

Busca pelo valor x:

```
return v[x] > 0;
```

Exemplo (motivação)

Funciona? Também funciona…

Exemplo (motivação)

Funciona? Também funciona…

Qual das duas abordagens é melhor?

Exemplo (motivação)

Funciona? Também funciona…

Qual das duas abordagens é melhor? Depende…

Exemplo (motivação)

Funciona? Também funciona…

Qual das duas abordagens é melhor? Depende…

- Busca da abordagem 1 é visivelmente mais custosa (depende da quantidade de elementos armazenados).

Exemplo (motivação)

Funciona? Também funciona…

Qual das duas abordagens é melhor? Depende…

- Busca da abordagem 1 é visivelmente mais custosa (depende da quantidade de elementos armazenados).

- Enquanto a busca da abordagem 2 leva sempre o mesmo tempo (tem custo constante, independente da quantidade de elementos armazenados).

Exemplo (motivação)

Funciona? Também funciona…

Qual das duas abordagens é melhor? Depende…

- Por outro lado, o valor máximo que pode ser armazenado na abordagem 2, está atrelado ao tamanho do vetor alocado (e isso não é legal).

Exemplo (motivação)

Funciona? Também funciona…

Qual das duas abordagens é melhor? Depende…

- Por outro lado, o valor máximo que pode ser armazenado na abordagem 2, está atrelado ao tamanho do vetor alocado (e isso não é legal). Para ser possível armazenar qualquer valor inteiro, o vetor tem que ser muito grande, e muito provavelmente diversas posições deste vetor nunca serão usadas. Logo, a abordagem 2 não é boa no que diz respeito ao consumo de memória.

Exemplo (motivação)

Funciona? Também funciona…

Qual das duas abordagens é melhor? Depende…

- Se pensarmos na operação (não ilustrada nos esboços de código) que envolve percorrer todos os valores armazenados, mais uma vez a abordagem 2 fica em desvantagem.

Exemplo (motivação)

Funciona? Também funciona…

Qual das duas abordagens é melhor? Depende…

- Se pensarmos na operação (não ilustrada nos esboços de código) que envolve percorrer todos os valores armazenados, mais uma vez a abordagem 2 fica em desvantagem. Mas continua sendo imbatível na busca!

Exemplo (motivação)

Funciona? Também funciona…

Qual das duas abordagens é melhor? Depende…

- A escolha por uma estrutura irá depender do contexto em que ela será usada (qual problema será resolvido com o auxílio da estrutura, quais são operações “críticas” para o problema em questão, e até onde toleramos as potenciais desvantagens).

Revisão linguagem C

Vamos para a revisão…