

Aula 02 - Estruturas lineares de dados

Vinícius Brito

Março de 2020



Tópicos da aula

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

1 Motivação

2 Estruturas lineares de dados

- Filas
- Pilhas
- Deques
- Pairs

3 Exercícios

Motivação

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Não é raro termos que lidar com uma grande quantidade de informações. Só que gerenciá-las pode ser uma dor de cabeça, e queremos diminuir ao máximo o nosso trabalho, especialmente quando se faz alguma modificação em porções menores de informação que estejam associadas ao restante (por exemplo, numa fila de atendimento, é extremamente importante saber quem vem antes ou depois de alguém, mesmo que alguma pessoa desista da fila ou entre alguém com prioridade).

Para lidar habilmente com tantos dados, que podem ser acessados ou modificados dinamicamente, fazemos uso das *Estruturas de Dados*, ou, mais formalmente, dos *Abstract Data Type* (TAD, em português).

Motivação

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Dequeues

Pairs

Exercícios

Infelizmente, a linguagem C não os tem prontos, e implementá-los consome um bom tempo. Por isso, faremos uso dos TADs prontos da linguagem C++, os quais estão associados ao "espaço padrão" (*standard namespace*) e à "biblioteca padrão" (*Standard Template Library*).

Primeiramente, vejamos como ficarão os códigos:

Motivação

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Infelizmente, a linguagem C não os tem prontos, e implementá-los consome um bom tempo. Por isso, faremos uso dos TADs prontos da linguagem C++, os quais estão associados ao "espaço padrão" (*standard namespace*) e à "biblioteca padrão" (*Standard Template Library*).

Primeiramente, vejamos como ficarão os códigos:

Exemplo de uso com a linguagem C++

```
#include <stdio.h>
//Incluirmos novas bibliotecas, mas agora são do C++

using namespace std;

int main( ){
    return 0;
}
```

Estruturas de dados - Filas

Estruturas de
dados (I)

Vinícius Brito

Motivação

Estruturas
lineares de
dados

Filas

Pilhas

Deque

Pairs

Exercícios

Nosso primeiro TAD são as filas, que, como o nome sugere, são estruturas dinâmicas de dados que são realmente análogas às filas por se basearem no funcionamento FIFO (*First In, First Out*). No entanto, só podemos acessar dela o elemento há mais tempo. Sua biblioteca associada é `<queue>`.

As funções mais usadas desse tipo são:

Estruturas de dados - Filas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Nosso primeiro TAD são as filas, que, como o nome sugere, são estruturas dinâmicas de dados que são realmente análogas às filas por se basearem no funcionamento FIFO (*First In, First Out*). No entanto, só podemos acessar dela o elemento há mais tempo. Sua biblioteca associada é `<queue>`.

As funções mais usadas desse tipo são:

- `push(x)` - Para inserir o elemento `x` na fila;

Estruturas de dados - Filas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Nosso primeiro TAD são as filas, que, como o nome sugere, são estruturas dinâmicas de dados que são realmente análogas às filas por se basearem no funcionamento FIFO (*First In, First Out*). No entanto, só podemos acessar dela o elemento há mais tempo. Sua biblioteca associada é `<queue>`.

As funções mais usadas desse tipo são:

- `push(x)` - Para inserir o elemento `x` na fila;
- `front()` - Para acessar o elemento há mais tempo na fila;

Estruturas de dados - Filas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Nosso primeiro TAD são as filas, que, como o nome sugere, são estruturas dinâmicas de dados que são realmente análogas às filas por se basearem no funcionamento FIFO (*First In, First Out*). No entanto, só podemos acessar dela o elemento há mais tempo. Sua biblioteca associada é `<queue>`.

As funções mais usadas desse tipo são:

- `push(x)` - Para inserir o elemento `x` na fila;
- `front()` - Para acessar o elemento há mais tempo na fila;
- `pop()` - Para remover o elemento há mais tempo na fila;

Estruturas de dados - Filas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Nosso primeiro TAD são as filas, que, como o nome sugere, são estruturas dinâmicas de dados que são realmente análogas às filas por se basearem no funcionamento FIFO (*First In, First Out*). No entanto, só podemos acessar dela o elemento há mais tempo. Sua biblioteca associada é `<queue>`.

As funções mais usadas desse tipo são:

- `push(x)` - Para inserir o elemento `x` na fila;
- `front()` - Para acessar o elemento há mais tempo na fila;
- `pop()` - Para remover o elemento há mais tempo na fila;
- `empty()` - Booleana indicando se a fila está vazia;

Estruturas de dados - Filas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Nosso primeiro TAD são as filas, que, como o nome sugere, são estruturas dinâmicas de dados que são realmente análogas às filas por se basearem no funcionamento FIFO (*First In, First Out*). No entanto, só podemos acessar dela o elemento há mais tempo. Sua biblioteca associada é `<queue>`.

As funções mais usadas desse tipo são:

- `push(x)` - Para inserir o elemento `x` na fila;
- `front()` - Para acessar o elemento há mais tempo na fila;
- `pop()` - Para remover o elemento há mais tempo na fila;
- `empty()` - Booleana indicando se a fila está vazia;
- `size()` - Para acessar o tamanho da fila

Estruturas de dados - Filas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

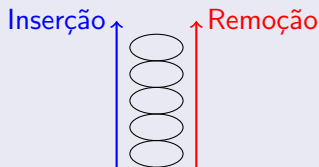
Exemplo de uso da fila

```
#include <stdio.h>
#include <queue>

using namespace std;

queue<int> fila;

int main( ) {
    fila.push(3);
    fila.push(5);
    printf("%d", fila.front( ));
    fila.pop( );
    printf("%d", fila.front( ));
    return 0;
}
```



Estruturas de dados - Pilhas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Em seguida, temos as pilhas, que, como o nome sugere, são estruturas dinâmicas de dados que são realmente análogas às pilhas por se basearem no funcionamento LIFO (*Last In, First Out*). No entanto, só podemos acessar dela o elemento há menos tempo. Sua biblioteca associada é `<stack>`.

As funções mais usadas desse tipo são:

Estruturas de dados - Pilhas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Em seguida, temos as pilhas, que, como o nome sugere, são estruturas dinâmicas de dados que são realmente análogas às pilhas por se basearem no funcionamento LIFO (*Last In, First Out*). No entanto, só podemos acessar dela o elemento há menos tempo. Sua biblioteca associada é `<stack>`.

As funções mais usadas desse tipo são:

- `push(x)` - Para inserir o elemento `x` na pilha;

Estruturas de dados - Pilhas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Em seguida, temos as pilhas, que, como o nome sugere, são estruturas dinâmicas de dados que são realmente análogas às pilhas por se basearem no funcionamento LIFO (*Last In, First Out*). No entanto, só podemos acessar dela o elemento há menos tempo. Sua biblioteca associada é `<stack>`.

As funções mais usadas desse tipo são:

- `push(x)` - Para inserir o elemento `x` na pilha;
- `top()` - Para acessar o elemento há menos tempo na pilha;

Estruturas de dados - Pilhas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Em seguida, temos as pilhas, que, como o nome sugere, são estruturas dinâmicas de dados que são realmente análogas às pilhas por se basearem no funcionamento LIFO (*Last In, First Out*). No entanto, só podemos acessar dela o elemento há menos tempo. Sua biblioteca associada é `<stack>`.

As funções mais usadas desse tipo são:

- `push(x)` - Para inserir o elemento `x` na pilha;
- `top()` - Para acessar o elemento há menos tempo na pilha;
- `pop()` - Para remover o elemento há menos tempo na pilha;

Estruturas de dados - Pilhas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Em seguida, temos as pilhas, que, como o nome sugere, são estruturas dinâmicas de dados que são realmente análogas às pilhas por se basearem no funcionamento LIFO (*Last In, First Out*). No entanto, só podemos acessar dela o elemento há menos tempo. Sua biblioteca associada é `<stack>`.

As funções mais usadas desse tipo são:

- `push(x)` - Para inserir o elemento `x` na pilha;
- `top()` - Para acessar o elemento há menos tempo na pilha;
- `pop()` - Para remover o elemento há menos tempo na pilha;
- `empty()` - Booleana indicando se a pilha está vazia;

Estruturas de dados - Pilhas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Em seguida, temos as pilhas, que, como o nome sugere, são estruturas dinâmicas de dados que são realmente análogas às pilhas por se basearem no funcionamento LIFO (*Last In, First Out*). No entanto, só podemos acessar dela o elemento há menos tempo. Sua biblioteca associada é `<stack>`.

As funções mais usadas desse tipo são:

- `push(x)` - Para inserir o elemento `x` na pilha;
- `top()` - Para acessar o elemento há menos tempo na pilha;
- `pop()` - Para remover o elemento há menos tempo na pilha;
- `empty()` - Booleana indicando se a pilha está vazia;
- `size()` - Para acessar o tamanho da pilha

Estruturas de dados - Pilhas

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

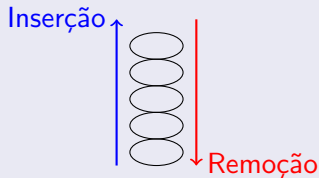
Exemplo de uso da pilha

```
#include <stdio.h>
#include <stack>

using namespace std;

stack<int> pil;

int main( ){
    pil.push(3);
    pil.push(5);
    printf("%d", pil.top( ));
    pil.pop( );
    printf("%d", pil.top( ));
    return 0;
}
```



Estruturas de dados - Deques

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deques

Pairs

Exercícios

Também temos as *deques* (Double-ended queues), que são um *mix* dos dois TADs anteriores. Nelas, podemos acessar, que nem um vetor, qualquer elemento armazenado. Sua biblioteca associada é `<deque>`.

As funções mais usadas desse tipo são:

Estruturas de dados - Deques

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Dequeues

Pairs

Exercícios

Também temos as *dequeues* (Double-ended queues), que são um *mix* dos dois TADs anteriores. Nelas, podemos acessar, que nem um vetor, qualquer elemento armazenado. Sua biblioteca associada é `<deque>`.

As funções mais usadas desse tipo são:

- `push_back(x)` - Para inserir o elemento `x` no fim da deque;

Estruturas de dados - Deques

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deques

Pairs

Exercícios

Também temos as *deques* (Double-ended queues), que são um *mix* dos dois TADs anteriores. Nelas, podemos acessar, que nem um vetor, qualquer elemento armazenado. Sua biblioteca associada é `<deque>`.

As funções mais usadas desse tipo são:

- `push_back(x)` - Para inserir o elemento `x` no fim da deque;
- `push_front(x)` - Para inserir o elemento `x` no começo da deque;

Estruturas de dados - Deques

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deques

Pairs

Exercícios

Também temos as *deques* (Double-ended queues), que são um *mix* dos dois TADs anteriores. Nelas, podemos acessar, que nem um vetor, qualquer elemento armazenado. Sua biblioteca associada é `<deque>`.

As funções mais usadas desse tipo são:

- `push_back(x)` - Para inserir o elemento `x` no fim da deque;
- `push_front(x)` - Para inserir o elemento `x` no começo da deque;
- `pop_back()` - Para remover o elemento no final da deque;

Estruturas de dados - Deques

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deques

Pairs

Exercícios

Também temos as *deques* (Double-ended queues), que são um *mix* dos dois TADs anteriores. Nelas, podemos acessar, que nem um vetor, qualquer elemento armazenado. Sua biblioteca associada é `<deque>`.

As funções mais usadas desse tipo são:

- `push_back(x)` - Para inserir o elemento `x` no fim da deque;
- `push_front(x)` - Para inserir o elemento `x` no começo da deque;
- `pop_back()` - Para remover o elemento no final da deque;
- `pop_front()` - Para remover o elemento no começo da deque;

Estruturas de dados - Deques

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deques

Pairs

Exercícios

Também temos as *deques* (Double-ended queues), que são um *mix* dos dois TADs anteriores. Nelas, podemos acessar, que nem um vetor, qualquer elemento armazenado. Sua biblioteca associada é `<deque>`.

As funções mais usadas desse tipo são:

- `push_back(x)` - Para inserir o elemento `x` no fim da deque;
- `push_front(x)` - Para inserir o elemento `x` no começo da deque;
- `pop_back()` - Para remover o elemento no final da deque;
- `pop_front()` - Para remover o elemento no começo da deque;
- `size()` - Para acessar o tamanho da deque;

Estruturas de dados - Deques

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deques

Pairs

Exercícios

Também temos as *deques* (Double-ended queues), que são um *mix* dos dois TADs anteriores. Nelas, podemos acessar, que nem um vetor, qualquer elemento armazenado. Sua biblioteca associada é `<deque>`.

As funções mais usadas desse tipo são:

- `push_back(x)` - Para inserir o elemento `x` no fim da deque;
- `push_front(x)` - Para inserir o elemento `x` no começo da deque;
- `pop_back()` - Para remover o elemento no final da deque;
- `pop_front()` - Para remover o elemento no começo da deque;
- `size()` - Para acessar o tamanho da deque;
- `empty()` - Booleana indicando se a deque está vazia

Estruturas de dados - Deques

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deques

Pairs

Exercícios

Exemplo de uso da deque

```
#include <stdio.h>
#include <deque>

using namespace std;

deque<int> dq;

int main( ){
    dq.push_back(3);
    dq.push_front(5);
    printf("%d", dq[0]);
    pil.pop_front( );
    printf("%d", dq[0]);
    return 0;
}
```

Estruturas de dados - Pairs

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Por fim, apresentamos os *pairs*, que, como o nome sugere, são pares de elementos. O mais interessante (e útil) é que os elementos pareados podem não ser do mesmo tipo! Eles podem ser vistos como *structs* pré-configurados. A biblioteca associada a eles é a `<utility>`. (Spoiler) Na próxima aula, será vista uma estrutura parecida com o pair, mas ainda mais poderosa. (Outro spoiler) O impacto desta aula é tão grande que envolve até grafos!

Exemplo de uso do pair

```
#include <stdio.h>
```

```
#include <utility>
```

```
using namespace std;
```

```
pair<int, int> ponto;
```

```
int main( ){
```

```
    ponto=make_pair(3, 5);
```

```
    printf("%d", ponto.first);
```

```
    ponto.second=ponto.first+2;
```

```
    printf("%d", ponto.first);
```

```
    return 0;
```

```
}
```

Exercícios Indicados

Estruturas de dados (I)

Vinícius Brito

Motivação

Estruturas lineares de dados

Filas

Pilhas

Deque

Pairs

Exercícios

Todos os problemas seguintes estão no URI Online Judge:

- ⇒ Brinde Face
- ⇒ Jogando Cartas Fora
- ⇒ $\text{Fibonacci}^{\{k\}}_i$
- ⇒ A Lenda de Flavious Josephus
- ⇒ TDA Racional
- ⇒ A Fila de Desempregados
- ⇒ Balanço de Parênteses
- ⇒ Abelha Maja