Pilhas e Filas

Filas

Prof. Edson Alves - UnB/FGA 2020

Sumário

- 1. Filas
- 2. Filas na STL

Filas

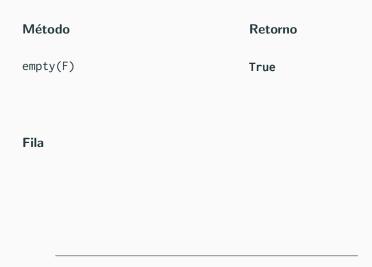
Definição de fila

- Uma fila é um tipo de dados abstrato cuja interface define que o primeiro elemento inserido na pilha é o primeiro a ser removido
- Esta estratégia de inserção e remoção é denominada FIFO First In,
 First Out
- De acordo com sua interface, uma fila n\u00e3o permite acesso aleat\u00f3rio ao seus elementos (apenas o elemento do topo da fila pode ser acessado)
- \bullet As operações de inserção e remoção devem ter complexidade O(1)

Interface de uma fila

Método	Complexidade	Descrição
clear(F)	O(N)	Esvazia a fila F, removendo todos os seus elementos
empty(F)	O(1)	Verifica se a fila F está vazia ou não
push(F, x)	O(1)	Insere o elemento x no final da fila F
pop(F)	O(1)	Remove o elemento que está no início da fila F
front(F)	O(1)	Retorna o elemento que está no início da fila F
size(F)	O(1)	Retorna o número de elementos armazenados na fila F

Métod	lo	Retorno
empty(F)	
Fila		
_		



Método Retorno push(F, 5) Fila

Método Retorno

push(F, 5)

Fila

5

7

Método Retorno
push(F, 11)

Fila

5

Método Retorno
push(F, 11)

Fila



Método Retorno
push(F, 7)

Fila



Método Retorno

push(F, 7)

Fila

11

5

11









Fila



Método Retorno front(F) Fila 11



Fila



Implementação de uma fila

- Como uma pilha é um tipo de dados abstrato, ela não impõe nenhuma restrição quanto à sua implementação
- É possível implementar uma pilha por composição, usando listas encadeadas ou uma deque (double-ended queue)
- A estratégia FIFO precisa de operações de inserção e remoção eficientes nos dois extremos do contêiner, o que inviabiliza o uso do vector e da forward_list
- Se há uma estimativa do tamanho máximo de elementos a serem inseridos na fila, é possível usar um array estático e o mesmo princípio de uma lista circular para implementar uma fila

Exemplo de implementação de fila em C++

```
1 #include <bits/stdc++ h>
3 using namespace std:
5 template<tvpename T. size t N>
6 class Oueue {
7 public:
     Queue() : first(0), last(0), _size(0) {}
9
     void clear() { first = last = _size = 0; }
10
     bool empty() const { return _size == 0; }
     size_t size() const { return _size; }
     void push(const T& x)
14
          if ( size == N) throw "Fila cheia":
16
         elems[last] = x:
18
          last = (last + 1) \% N;
          size++:
20
```

Exemplo de implementação de fila em C++

```
void pop()
24
          if (_size == 0) throw "Lista vazia";
26
          first = (first + 1) \% N;
          size--:
28
29
30
      const T& front() const
31
          if (_size == 0) throw "Lista vazia";
34
          return elems[first];
36
38 private:
      array<T, N> elems;
39
     int first, last;
40
     size_t _size;
41
42 };
```

Exemplo de implementação de fila em C++

```
44 int main()
45 {
      Oueue<int. 10> a:
46
47
      cout << "Empty?" << g.empty() << '\n':
48
49
      for (int i = 1: i \le 10: ++i) a.push(i):
50
51
      cout << "Size = " << q.size() << '\n';
      cout << "Front = " << q.front() << '\n';
54
      for (int i = 0: i < 5: ++i) g.pop():
55
56
      q.push(11);
      q.push(12);
58
59
      cout << "Size = " << q.size() << '\n';
60
      cout << "Front = " << q.front() << '\n';
61
62
      return 0;
63
64 }
```

Filas na STL

Filas em C++

- A STL do C++ oferece uma implementação de fila: a classe queue
- Assim como no caso das pilhas, o contêiner usado na composição é, por padrão, deque
- Este contêiner pode ser substituído por qualquer contêiner que contenha os métodos pop_front(), push_back() e size(), dentre outros
- O método swap() também está disponível

Exemplo de uso de fila em C++

```
1 #include <iostream>
3 int main()
4 {
      std::queue<std::string> q { "um", "dois", "tres" };
      std::cout << "Vazia? " << g.emptv() << '\n':
                                                                      // Falso
8
      std::cout << "Primeiro elemento: " << q.front() << '\n'; // "um"</pre>
      std::cout << "Último elemento: " << q.back() << '\n'; // "tres"</pre>
10
      q.push("quatro");
      std::cout << "Tamanho da fila: " << q.size() << '\n';</pre>
                                                                      // 4
14
      q.pop();
      std::cout << "Primeiro elemento: " << q.front() << '\n';  // "dois"</pre>
      return 0;
20
21 }
```

Referências

- 1. **DROZDEK**, Adam. *Algoritmos e Estruturas de Dados em C++*, 2002.
- 2. **KERNIGHAN**, Bryan; **RITCHIE**, Dennis. *The C Programming Language*, 1978.
- 3. **STROUSTROUP**, Bjarne. *The C++ Programming Language*, 2013.
- 4. C++ Reference. Queue, acesso em 22/04/2019.