Pilhas e Filas

Filas

Prof. Edson Alves – UnB/FGA

Sumário

- 1. Filas
- 2. Filas na STL

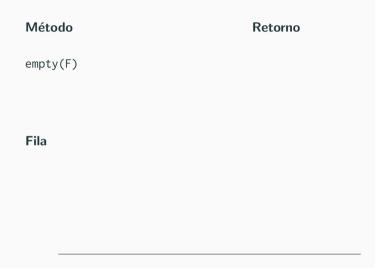
Filas

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)
- ullet 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



Método	Retorno
empty(F)	True
Fila	

Método	Retorno
push(F, 5)	
Fila	



Método Retorno

push(F, 11)

Fila























Implementação de uma fila

- Como uma fila é um tipo de dados abstrato, ela não impõe nenhuma restrição quanto à sua implementação
- É possível implementar uma fila 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 <hits/stdc++ h>
₃ using namespace std;
5 template<typename T, size_t N>
6 class Queue {
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;
          last = (last + 1) \% N:
18
          _size++;
20
```

Exemplo de implementação de fila em C++

```
void pop()
22
          if (_size == 0) throw "Lista vazia";
24
          first = (first + 1) \% N;
26
          _size--:
28
29
      const T& front() const
30
31
          if (_size == 0) throw "Lista vazia";
32
          return elems[first];
34
35
36
37 private:
      array<T, N> elems;
     int first, last;
      size_t _size;
41 };
```

Exemplo de implementação de fila em C++

```
43 int main() {
                                    Queue<int, 10> q;
 44
 45
                                    cout << "Empty? " << q.empty() << '\n';</pre>
 46
 47
                                    for (int i = 1; i \le 10; i \ne 10
 48
 49
                                    cout << "Size = " << q.size() << '\n':
50
                                      cout << "Front = " << a.front() << '\n':
51
52
                                    for (int i = 0; i < 5; ++i) q.pop();
53
54
                                    q.push(11);
55
                                    a.push(12):
56
57
                                    cout << "Size = " << q.size() << '\n':
58
                                    cout << "Front = " << q.front() << '\n';
59
60
                                      return 0;
61
 62 }
```

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, a 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++

```
#include <bits/stdc++.h>
3 int main()
4 {
      std::queue<std::string> q { "um", "dois", "tres" };
5
6
      std::cout << "Vazia? " << q.emptv() << '\n':
                                                                     // Falso
7
8
      std::cout << "Primeiro elemento: " << q.front() << '\n': // "um"</pre>
9
      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>
18
      return 0:
19
20 }
```

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.