```cpp
#include <iostream>
#include <sstream>
#include <initializer_list>

template <int N, typename T>
class tableau {

private:
  // le tableau
  T elem[N];

public:
  // constructeur pour initialiser tous les élements
  // à partir d'une liste de valeur
  tableau(const std::initializer_list<T> &t) {
    int size = t.size();
    for (int i=0; i<size; i++)
      elem[i] = *(t.begin()+i);
  }
  // constructeur pour initialiser tous les élements
  // du tableau à une valeur v
  tableau(const T& v=T()) {
    for(int i=0 ; i<N ; i++) this->elem[i] = v;
  }

  // constructeur de copie
  tableau (const tableau<N,T>& t) {
    for(int i=0 ; i<N ; i++)
      this->elem[i] = t.elem[i];
  }

  // surcharge de l'opérateur d'affectation
  tableau<N,T> &operator=(const tableau<N,T>& t) {
    for(int i=0 ; i<N ; i++)
      this->elem[i] = t.elem[i];
    return *this;
  }

  // rôle : renvoie le nombre d'éléments du tableau courant
  int longueur() const {
    return N;
  }

  // surcharge de l'opérateur [], renvoie une référence sur t[i]
  T &operator[] (int i) {
    return this->elem[i];
  }

  // rôle : renvoie la représentation sous forme de string
  //        de l'objet courant
  const std::string toString() const {
    std::ostringstream s;
    s << "[ ";
    for (auto x : this->elem) s <<  x << " ";
    s << "]";
    return s.str();
  }

  // surcharge de l'opérateur de sortie <<
  friend std::ostream& operator<<(std::ostream& f, const tableau<N,T> &t)  {
    return f << t.toString();
  }
};
```