```cpp
/*
  ---------------------------------------------------------------------------
  Fichier     : calculMatrice.cpp
  Auteur(s)   : Mario Amos & Théo Coutaudier
  Date        : 08.12.2021
  But         : Librairie mettant à disposition des utilitaires pour le traitement
                de vecteur ou matrice

  Compilateur : Mingw-w64 g++ 11.2.0
  ---------------------------------------------------------------------------
*/

#include <algorithm>
#include <numeric>

#include "matrice.h"

using namespace std;

/**
 * @param v1 Premier vecteur à comparer
 * @param v2 Second vecteur à comparer
 * @return Retourne true si v1 est plus petit ou égal que v2, sinon false
 */
bool comparerTailleVecteur(const Vecteur& v1, const Vecteur& v2);

/**
 * Trouve la taille maximum des colonnes d'une matrice
 * @param m
 * @return
 */
unsigned maxCol(const Matrice& m);

/**
 * Effectue la somme de toutes les valeurs d'un vecteur
 * @param v
 * @return
 */
int sommeVecteur(const Vecteur& v);

/**
 * Effectue l'addtion la valeur de deux cellules d'un vecteur
 * @param v1 Valeur 1
 * @param v2 Valeur 2
 * @return Valeurs additionnées
 */
int add(int v1, int v2);

/**
 * Est équivalent à l'addition de deux vecteur:
 * Valeur de retour = {v1[0]+v2[0],v1[1]+v2[1],v1[2]+v2[2],...}
 * @param v1
 * @param v2
 * @return
 */
Vecteur additionColonne(Vecteur v1, Vecteur v2);

unsigned minCol(const Matrice& m) {
   if (m.empty())
      return 0;
   return (*min_element(m.begin(), m.end(), comparerTailleVecteur)).size();
}

Vecteur sommeLigne(const Matrice& m) {
   Vecteur resultat(m.size());
   transform(m.begin(), m.end(), resultat.begin(), sommeVecteur);
   return resultat;
}

Vecteur sommeColonne(const Matrice& m) {
   return accumulate(m.begin(), m.end(), Vecteur(maxCol(m), 0), additionColonne);
}
```

```
73
74  Vecteur vectSommeMin(const Matrice& m) {
75     if (m.empty())
76        return {};
77
78     Vecteur somme = sommeLigne(m);
79     //On récupère un itérateur de min_element, on en soustrait le début du vecteur
80     //pour en ressortir l'index
81     unsigned index = min_element(somme.begin(), somme.end()) - somme.begin();
82     return m[index];
83  }
84
85  bool comparerTailleVecteur(const Vecteur& v1, const Vecteur& v2) {
86     return v1.size() <= v2.size();
87  }
88
89  unsigned maxCol(const Matrice& m) {
90     if (m.empty())
91        return 0;
92     return (*max_element(m.begin(), m.end(), comparerTailleVecteur)).size();
93  }
94
95  int sommeVecteur(const Vecteur& v) {
96     return accumulate(v.begin(), v.end(), 0);
97  }
98
99  int add(int v1, int v2) {
100    return (v1 + v2);
101 }
102
103 Vecteur additionColonne(Vecteur v1, Vecteur v2) {
104    //On doit resize le vecteur le plus petit, sinon cela va aller chercher une
105    //valeur "aléatoire" en mémoire pour l'addition
106    const size_t TAILLE_MAX = max(v1.size(), v2.size());
107    if (v1.size() < TAILLE_MAX)
108       v1.resize(TAILLE_MAX, 0);
109    if (v2.size() < TAILLE_MAX)
110       v2.resize(TAILLE_MAX, 0);
111
112    Vecteur res(TAILLE_MAX, 0);
113    //Additionne des deux vecteurs
114    transform(v1.begin(), v1.end(), v2.begin(), res.begin(), add);
115    return res;
116 }
```