Exercice : Entropie

```cpp
#include <iostream>
#include <cmath> // pour log()
#include <cctype> // pour isalpha() et toupper()
#include <vector>
#include <array>
using namespace std;

// ========================================================================
typedef vector<double> Distribution;

// ========================================================================
Distribution calcule_probas(string const& s, bool compter_espace = false)
{
  array<double, 27> frequences; // 28 = 26 caractères alphabétiques + 1 espace
  for (auto& f : frequences) f = 0.0;
  double somme(0.0); /* Nombre de caractères pris en compte.
                      * N'est pas forcément égal à s.size() si il y a des caractères para
                      * Mis en double pour la division ultérieure. */

  for (const char c : s) {
    if (isalpha(c)) {
      ++frequences[toupper(c) - 'A'];
      ++somme;
    } else if ((c == ' ') and (compter_espace)) {
      ++frequences.back();
      ++somme;
    }
  }

  // Crée la distribution
  Distribution probas;

  for (auto& f : frequences) {
    if (f > 0.0) { // on ne retient que les non nuls ici
      probas.push_back(f / somme);
    }
  }

   return probas;
}

// ========================================================================
double log2(double x)
{
  if (x == 0.0) return 0.0; // pour éviter le NaN
  return log(x) / log(2.0);
}

// ========================================================================
double entropie(Distribution const& probas)
{
  double entropy(0.0);
  for (auto p : probas) {
    entropy -= p * log2(p);
  }
  return entropy;
}

// ========================================================================
double entropie(string const& s)
{
  return entropie(calcule_probas(s));
}

// ========================================================================
```

```cpp
// Tests
// -----------------------------------------------
void test2(Distribution const& probas) {
  cout << "p=(";
  for (auto p : probas) cout << p << ", ";
  cout << ") --> H = " << entropie(probas) << " bit" << endl;
}
// -----------------------------------------------
void test1(double p0) {
  test2({ p0, 1.0 - p0 });
}
// -----------------------------------------------
void test3(string const& s) {
  cout << "Chaîne « " << s << " » :" << endl;
  test2(calcule_probas(s));
  cout << "et directement : H = " << entropie(s) << " bit" << endl;
}

// ====================================================================
int main()
{
  // ---- 1) tests entropie binaire
  test1(0.0);
  test1(0.3);
  test1(0.5);
  test1(0.7);
  test1(1.0);

  test2({ 0.1, 0.2, 0.3, 0.4 });
  test2(Distribution(8, 1.0/8.0));  // équirépartie

  test3("IL FAIT BEAU A IBIZA");
  test3("AAAAA");
  test3("A");

  cout << "Entrez une chaîne : ";
  string s;
  cin >> s;
  test3(s);

  return 0;
}
```