

TP_s6.1 : string (solution)

Lien avec le [MOOC Initiation à la Programmation \(en C++\)](#)
Lien avec ICC-Théorie en complément du MOOC

La solution des exercices du MOOC est disponible sur le MOOC.

ExC 3 : Opérations simples en langage naturel

Question1 :

avec opérandes entre zéro et neuf et résultats entre zéro et dix-huit

```
#include <iostream>
#include <array>
#include <string>

using namespace std;

constexpr size_t nb(19);

size_t indice_operande(string operande,
                       array<string,nb>& mot,
                       size_t max);

int main()
{
    constexpr size_t nb_chiffre(10);
    array<string, nb> mot = {"zéro", "un", "deux", "trois", "quatre",
                           "cinq", "six", "sept", "huit", "neuf",
                           "dix", "onze", "douze", "treize", "quatorze",
                           "quinze", "seize", "dix-sept", "dix-huit"};

    cout << "Entrez une opération : ";

    string val_gauche, operateur, val_droit;
    cin >> val_gauche >> operateur >> val_droit;

    size_t i_g(indice_operande(val_gauche, mot, nb_chiffre));
    size_t i_d(indice_operande(val_droit, mot, nb_chiffre));

    if(operateur != "plus" and operateur != "moins")
    {
        cout << "Erreur: operateur invalide" << endl;
        return 0;
    }

    cout << " égal " ;

    if(i_g < i_d and operateur == "moins")
        cout << "moins " << mot[i_d - i_g] << endl;
    else if(operateur == "moins")
        cout << mot[i_g - i_d] << endl;
    else
        cout << mot[i_g + i_d] << endl;

    return 0;
}
```

```

size_t indice_operande(string operande, array<string,nb>& mot, size_t max)
{
    size_t i(0);
    while(i < max and mot[i] != operande) ++i;

    if(i >= max)
    {
        cout << "Erreur: opérande invalide" << endl;
        exit(0);
    }
    return i;
}

```

Question2 :

avec opérandes signés entre moins neuf et neuf et résultats entre moins dix-huit et dix-huit

```

#include <iostream>
#include <array>
#include <string>

using namespace std;

constexpr size_t nb(19);
constexpr size_t nb_chiffre(10);

int indice_operande_signe(array<string,nb>& mot,
                          size_t max);

size_t indice_operande(string operande,
                      array<string,nb>& mot,
                      size_t max);

int main()
{
    array<string, nb> mot={"zéro","un","deux","trois","quatre",
                          "cinq","six","sept","huit","neuf",
                          "dix","onze","douze","treize","quatorze",
                          "quinze", "seize","dix-sept", "dix-huit"};

    cout << "Entrez une opération : ";

    int i_g(indice_operande_signe(mot,nb_chiffre));

    string operateur;
    cin >> operateur;
    if(operateur != "plus" and operateur != "moins")
    {
        cout << "Erreur: operateur invalide" << endl;
        return 0;
    }

    int i_d(indice_operande_signe(mot,nb_chiffre));

    cout << " égal " ;
    int val = ((operateur == "plus") ? (i_g + i_d) : (i_g - i_d) );

    if(val < 0)
        cout << "moins " << mot[-val] << endl;
    else
        cout << mot[val] << endl;

    return 0;
}

```

```

int indice_operande_signe(array<string,nb>& mot,
                          size_t max)
{
    int i(0);
    string test_signe, operande;
    cin >> test_signe;

    if(test_signe == "moins")
    {
        cin >> operande ;
        i = -indice_operande(operande,mot,nb_chiffre);
    }
    else
    {
        operande = test_signe;
        i = indice_operande(operande,mot,nb_chiffre);
    }
    return i;
}

size_t indice_operande(string operande,
                       array<string,nb>& mot,
                       size_t max)
{
    size_t i(0);
    while(i < max and mot[i] != operande) ++i;

    if(i >= max)
    {
        cout << "Erreur: opérande invalide" << endl;
        exit(0);
    }
    return i;
}

```