Exercice 1. Utilisation des entrées-sorties clavier

Écrire un programme qui demande à l'utilisateur de saisir un nombre et puis affiche le nombre saisi à l'aide des flots d'entrée-sortie *std::cin* et *std::cout* définis dans *iostream*. Le faire pour les types **int**, **short**, **long**, **char**, **float**, **double**, et **char ***. Que pouvez-vous conclure sur les opérateurs << et >> ? Testez les manipulateurs définis dans *iomanip* (au minimum setw, setfill et setprecision).

Exercice 2. La surcharge de fonction en C++

Soit les fonctions suivantes :

```
void affiche(int x, double y)
{
    cout << "affiche 1" << endl;
    cout << x << " " << y << endl;
}
void affiche(double x, int y)
{
    cout << "affiche 2" << endl;
    cout << x << " " << y << endl;
}
```

Testez/appelez ces 2 fonctions dans la fonction main avec 2 variables en paramètres effectifs de type : **double** et **int**, puis **int** et **double**, puis **double** et **double**, et finalement **int** et **int**. Conclure.

Exercice 3. Fonctions C++ et passage par référence des arguments/paramètres

Écrire une fonction *echange* qui permute le contenu de deux variables à l'aide du *passage par référence* (différent du passage par valeur ou par adresse/pointeur). Plus précisément, écrire 2 exemplaires de cette fonction, 1 ayant pour arguments 2 **int&** et l'autre 2 **double&** (`void echange(int& i1, int& i2);` et `void echange(double& d1, double& d2);`). Testez ces 2 fonctions dans le programme principal, en passant 2 variables du type adéquate, et ensuite une variable de type **int** l'autre de type **double**. Conclure sur le passage par référence.

Exercice 4. Compréhension de code C++ et utilisation d'un argument/paramètre par défaut

Soit la fonction suivante (une version style C):

```
#include <cstring> // pour utiliser les fonctions de manipulation des chaînes style C
char* concat(char *dest, const char *ajout, int nb_carac_max=0){
   int l_dest = strlen(dest), l_ajout=strlen(ajout) ;

   if( nb_carac_max <= 0 || nb_carac_max > l_dest + l_ajout )
      nb_carac_max = l_dest + l_ajout ;

   if( nb_carac_max > l_dest )
      memmove (dest+l_dest, ajout, nb_carac_max - l_dest);

   dest[nb_carac_max] = '\0';

   return dest ;
}
```

Que fait cette fonction *concat* (memmove copie un certain nombre d'octets) ? Quel est l'intérêt de l'argument par défaut `int nb_carac_max=0` ? Expliquer les différents cas d'utilisation de la fonction *concat*.

Exercice 5. Comprendre les intérêts de la Programmation Orientée Objet

Soit la fonction suivante, qui est la version *Programmation Orientée Objet* de la fonction précédente :

```
#include <string> // pour pouvoir utiliser la classe chaîne std::string du C++
using namespace std; // permet de ne plus préfixer les types et fonctions de ce namespace
string& concat_string(string& dest, const string& ajout, int nb_carac_max=0){
  int l_dest = dest.size(), l_ajout = ajout.size() ;

  if( nb_carac_max <= 0 || nb_carac_max >= l_dest + l_ajout )
     dest = dest + ajout;
  else
  {
     if( nb_carac_max > l_dest )
        dest = dest + ajout.substr(0, nb_carac_max - l_dest);
     else if( nb_carac_max < l_dest ) dest = dest.substr(0, nb_carac_max);
  }

  return dest ;
}
```

string est la **classe** *chaîne de caractères* du C++ qui est définie dans l'espace de nom std. **Une classe est un type pourvu de méthodes** (fonctions définies dans la classe) **et qui cache les détails d'implantation à l'utilisateur**. Une variable, dont le type est une classe, est un objet. On peut appliquer une méthode d'instance sur un objet de son type avec la syntaxe *monObjet.maMethode()*.

Listez les différences entre la version procédurale et la version POO. Quelle fonction est la plus facile à utiliser, et la plus maintenable ? Est-ce que l'utilisation de l'argument `nb_carac_max` est toujours justifiée dans le cas de la version POO ? Justifiez vos réponses.

Si *prenom* et *nom* sont deux string, est-ce que le code suivant est correct :

```
concat_string(prenom, nom) = concat_string(prenom, nom) + prenom;
```

Si oui, expliquez pourquoi et donnez le résultat ?