UET Introduction à la Programmation — Université Grenoble Alpes
2017-18.

Feuille de TP n° 1.

1 Exercices types, flottants

1. Afficher une table de multiplication demandée à l'utilisateur
2. Déterminer le plus petit entier n tel que `(1.0+2^(-n))-1.0==0` en utilisant le type `float` et `double`. Attention `^` n'est pas la puissance (c'est le ou exclusif en C). Attention, utiliser `1.0f` pour coder 1.0 en flottants. En déduire le nombre de bits de la mantisse dans les 2 cas.
3. Comparer le résultat des commandes

   `cout << 2*1/2 << endl;`, `cout << (2*1)/2 << endl;`,
   `cout << 2*(1/2) << endl;`.

   Expliquer. Faire un programme qui calcule $n!$ pour un entier donné n. Pour quelles valeurs de n la réponse est-elle satisfaisante ? On pourra utiliser le type `int`, puis le type `long int` (et comparer les rapports successifs de $n!/(n-1)!$).
4. Résoudre par dichotomie $\cos(x) = x$ sur l'intervalle $[0, 1]$. Rappel : si f continue change de signe sur $[a, b]$, on peut déterminer une valeur approchée de r tel que $f(r) = 0$ en prenant c le milieu de a et b, si $f(c)f(a) < 0$ on remplace b par c sinon on remplace a par c. Attention, il est inutile de faire plus de 53 itérations avec le type `double`. Utiliser `#include<cmath>`pour la fonction `cos`.
5. Calculer des sommes partielles de séries à terme positif décroissant en commençant par le terme de plus grand ou de plus petit indice, comparer les résultats en `float` ou `double`, par exemple pour

$$\sum_{k=1}^{N} \frac{1}{k^2}, \quad \sum_{k=1}^{N} \frac{1}{k}, \ldots$$

2 Exercices fonctions/vector

Pour lire un `vector<double>` depuis un fichier `data` du disque dur contenant le nombre d'éléments suivi par la liste des valeurs, (par exemple faire `emacs data &` puis entrer `3 1.01 2.3 4.7` et sauvegarder), vous pouvez vous inspirer du prorgamme suivant :

```
#include <iostream>
#include <fstream>
#include <vector>

using namespace std;

istream & operator >> (istream & is,vector<double> & v){
  int n;
  is >> n;
```

```
  v.resize(n);
  for (int i=0;i<n;i++)
    is >> v[i];
  return is;
}

ostream & operator << (ostream & os,vector<double> & v){
  int n=v.size();
  os << n << " ";
  for (int i=0;i<n;++i)
    os << v[i] << " ";
  return os;
}

int main(){
  vector<double> v;
  ifstream fichier("data");
  fichier >> v;
  cout << v << endl;
}
```

1. Écrire une fonction min qui calcule le minimum de 2 entiers, puis l'utiliser pour créer une fonction min qui calcule le minimum de 3 entiers.
2. Écrire une fonction qui calcule le min d'une liste de réels.
3. Écrire une fonction qui prend en argument deux nombres a, b et remplace a par le plus petit et b par le plus grand.
4. Écrire une fonction prenant en argument un `vector<double>` et calculant les éléments statistiques : moyenne, écart-type, médiane, quartiles. On pourra utiliser `sort` pour trier le vecteur.
5. Écrire une fonction prenant en argument un `vector<double>` représentant les coefficients d'un polynôme P et un `double` représentant x et calculant $P(x)$.
6. Écrire une fonction calculant une valeur approchée de $\cos(x)$ sur $[-1, 1]$ en utilisant le développement de Taylor en 0 à l'ordre 12.
7. Écrire une fonction récursive puis itérative pour calculer le PGCD de 2 entiers (`int`)
8. Écrire une fonction récursive calculant $a^n \pmod p$ (si $n = 0$ on renvoie 1, si $n = 1$ on renvoie a, si n est pair on renvoie le carré de $a^{n/2} \pmod p$ etc.)
9. Écrire une fonction déterminant la position d'un nombre dans une liste triée de nombres par dichotomie. On renverra -1 si le nombre n'est pas trouvé.
10. Écrire une fonction de dichotomie qui prend en argument une fonction `double` donne `double` et les extremités de l'intervalle a, b. Tester.