❑ **Exercice 1** : *Spécifications*

Proposer un nom, une spécification, des préconditions et un jeu de tests pour les fonctions suivantes :

1.
```
bool fonction1(int a, int b, int c)
  {return (a==b) || (b==c) || (a==c);}
```

2.
```
float fonction2(float x, float y)
  {return 1/(x*x+y*y);}
```

3.
```
int fonction3(float a, float b) {
  if (a<b)
  {return a;}
  else
  {return b;}
  }
```

❑ **Exercice 2** : *Fonction mystère*

On considère la fonction `mystere` suivante :

```
bool mystere(int n) {
  int d=2;
  while (d*d<=n)
  {  if (n%d==0)
      {return false;}
    d=d+1;}
  return true;
}
```

1. Nommer cette fonction et en donner une spécification.
2. Tracer son graphe de flot de contrôle.
3. Proposer un jeu de tests permettant de couvrir tous les arcs.

❑ **Exercice 3** : *nombre de jours dans un mois*

1. Ecrire une fonction `nb_jours` qui prend en argument un entier `mois` et un entier `annee` et qui renvoie le nombre de jours de ce mois. Par exemple, `nb_jours(5,1970)` doit renvoyer le nombres de jours du mois de mai 1970. On pourra utiliser sans la réécrire la fonction `bissextile` vue en cours.
2. Proposer des préconditions pour cette fonction.
3. Proposer un jeu de tests pour cette fonction.

❑ **Exercice 4** : *Triangles*

1. Ecrire une fonction `triangle` qui prend en argument trois entiers et renvoie :
   — 0 si les trois entiers ne sont pas les côtés d'un triangle
   — 1 si les trois entiers sont les côtés d'un triangle scalène
   — 2 si les trois entiers sont les côtés d'un triangle isocèle non rectangle
   — 3 si les trois entiers sont les côtés d'un triangle équilatéral
   — 4 si les trois entiers sont les côtés d'un triangle rectangle
2. Tracer le graphe de flot de contrôle de cette fonction.
3. Proposer un jeu de tests pour cette fonction.

❑ **Exercice 5** : *Compte à rebours* On considère la fonction C suivante :

```c
// Compte à rebours de n à 0 et afficher "Partez !"
void compte_rebours(int n)
{
    while (n != 0)
    {
        printf("%d \n",n);
        n = n-1;
    }
    printf("Partez !\n");
}
```

1. Cette fonction est-elle conforme à sa spécification ?
2. n est-il un variant de boucle ? Sinon quelle propriété est manquante ?

❑ Exercice 6 : *Somme des éléments pairs*

1. Ecrire en C une fonction qui renvoie la somme des éléments pairs d'un tableau. Par exemple pour le tableau `{2, 5, 4, 9, 7, 6}`, la fonction renvoie `2 + 4 + 6 = 12`.
2. Prouver la correction de cette fonction.

❑ Exercice 7 : *Etude d'un algorithme*

On considère l'algorithme suivant :

Algorithme : Quotient dans la division euclidienne

Entrées : $a \in \mathbb{N}, b \in \mathbb{N}^*$
Sorties : $q \in \mathbb{N}$, quotient de a par b

```
q ← 0
tant que a − bq > 0 faire
    q ← q + 1
fin
return q
```

1. Proposer une implémentation de cet algorithme en langage C, sous la forme d'une fonction dont on précisera soigneusement la spécification.
2. Etudier la terminaison de cet algorithme.
3. Etudier la correction de cet algorithme.

❑ Exercice 8 : *Multiplication à la russe*

On considère l'algorithme suivant :

Algorithme : Multiplication à la russe

Entrées : $a \in \mathbb{N}, b \in \mathbb{N}$
Sorties : ab

```
m ← 0
tant que a > 0 faire
    si a mod 2 = 1 alors
        m ← m + b
        a ← a − 1
    fin
    sinon
        a ← a/2
        b ← b + b
    fin
fin
return m
```

1. Faire fonctionner cet algorithme à la main pour $a = 7$ et $b = 5$.
2. Donner une implémentation en C de cette algorithme sous la forme d'une fonction dont on précisera soigneusement la spécification
3. Prouver la terminaison de cet algorithme.
   ⊗ a est un variant de boucle.

4. Prouver la correction de cet algorithme.
   🛟 $m + ab$ vaut toujours le produit des valeurs initales de a et de b.

❑ **Exercice 9** : *Tri par sélection*

L'algorithme du tri par sélection d'un tableau t de longueur n, consiste, pour chaque entier i de 0 à $n-1$ à :

— rechercher le plus petit élément du sous tableau $\{t[i], \ldots, t[n-1]\}$.

— l'échanger avec celui situé à l'indice i.

1. Ecrire cet algorithme en pseudo langage
2. En donner une implémentation en langage C en spécifiant soigneusement les fonctions utilisées.
3. Proposer un jeu de tests
4. Prouver la correction totale de cet algorithme.