TD n°3 - Preuves de programmes

Exercice 1

La fonction suivante renvoit le minimum d'un tableau :

```
int minimum(int n, int* tab){
  int res = tab[0];
  for(int i=0, i<n, i+=1){
    if(tab[i]<res){res = tab[i];}
  }
  return res;
}
```

- Écire une spécification pour cette fonction.
- Trouver un invariant de la boucle pour tout. Indication : un invariant utile porte sur la valeur de `res`.
- Justifier la correction de cette fonction

Exercice 2

La fonction suivante calcule le terme k de la suite $(u_n)_{n\in\mathbb{N}}$ définie par $u_0 = c$ et $\forall n \in \mathbb{N}$, $u_{n+1} = a * u_n + b$ pour a, b, c des entiers :

```
int k_ieme(int a, int b, int c, int k){
  int res = c;
  for(int i=1; i<=k;i+=1){
    res = a*res+b;
  }
  return res;
}
```

1. Écrire une spécification pour cet algorithme.
2. Justifier que ce programme termine.
3. Trouver un invariant.
4. Justifier la correction de ce programme.

Exercice 3

La fonction suivante a pour but de calculer la partie entière de la racine carrée de $n \in \mathbb{N}$.

```
int racine_approchee(int n){
  assert(n>0);
  int r = 0;
  int s = 1;
  while (s<=n){
    r = r+1;
    s = s+2*r+1;
  }
  return r;
}
```

1. Écrire une spécification de l'algorithme (précondition, postcondition)
2. Montrer que "r est positif" est un invariant de la boucle.
3. Démontrer que la boucle se termine.
4. Montrer que $(r+1)^2 = s$ est un invariant de la boucle.
5. En déduire que la postcondition est vérifiée.

Exercice 4

Outre le fait de permettre de prouver la correction et la terminaison, réfléchir aux invariants en avance permet d'écrire du code bien ordonné.

Dans cet exercice on cherche à écrire un algorithme réalisant la division euclidienne de a par b avec $a, b \in \mathbb{N}$ différents de $(0, 0)$. (Précondition)

On cherche donc un couple $q, r \in \mathbb{N}$ tel que $a = bq + r$ avec $r \in [|0, b-1|]$. (Postcondition)

On vous donne le squelette de code suivant, pour ne pas avoir à renvoyer q et r (ce qui en C n'est pas immédiat), on les affiche :

```
void affiche_div(int a, int b){
 assert(a>=0);
 assert(b>=0);
 assert( !(a==0&&b==0) );

 int q =   ;
 int r =   ;

 while(      ){



 }
 printf("quotient = %d et reste = %d",q,r);
}
```

Le code précédent utilise des variables q et r qui vérifient toujours $a = qb + r$ et $0 \leq r$ tout au long de l'algorithme, mais pas forcément $r < b$. De plus, un variant de la boucle doit être r.

1. En considérant les deux invariants et le variants, déduire la condition de fin de la boucle.
2. Compléter l'initialisation de q et r pour que les invariants soient vérifiés.
3. Compléter le corps de la boucle pour que les invariants et le variant soient satisfaits.

Exercice 5

Pour chacune des fonctions récursives suivantes, montrer qu'elles sont correctes (et qu'elles terminent).

Cette fonction calcule x^n

```
int exp_semi_rapide(int x, int n){
  if (n==0){return 1;}
  else if (n%2==0){return exp_semi_rapide(x*x,n/2);}
  else {return x*exp_semi_rapide(x,n-1);}
}
```

Cette fonction détermine si n est pair.

```
bool est_pair(int n){
  if (n==0){return true;}
  if (n==1){return false;}
  return est_pair(n-2);
}
```

Le but de cette fonction est de déterminer la somme des éléments d'un tableau, mais il faut que vous commenciez par préciser exactement la somme de quels éléments du tableau elle calcule.

```
int somme_rec(int* tab, int i, int taille_tab){
  if (i==taille_tab){return 0;}
  else {return tab[i]+somme_rec(tab,i+1,taille_tab);}
}
```

Exercice 6 (*)

On considère l'algorithme suivant, dit "algorithme de recherche dichotomique".

Précondition : tab est un tableau trié dans l'ordre croissant et valeur est une valeur du bon type.

Postcondition : la fonction renvoie un indice i tel que $tab[i] = valeur$ si un tel i existe et -1 sinon.

```
Fonction recherche
Entrées tab un tableau et valeur un élément
Soit gauche = 0 et droite = longueur(tab)-1
Tant que gauche <= droite:
    m = (gauche+droite)/2
    Si valeur = tab[m] alors renvoyer m et arrêter l'algorithme
    Sinon, si valeur < tab[m] alors droite = m-1
          sinon gauche = m+1
Renvoyer -1
```

1. droite est il un variant de la boucle tant que? gauche est il un variant? Comment construire un variant avec droite et gauche? (faire des dessins de l'évolution de droite et gauche sur plusieurs exemples)

2. En déduire la terminaison.

3. Supposons que valeur est dans tab. Quel peut être un invariant? Prouver l'invariant.

4. En déduire que soit valeur est dans le tableau et la fonction renvoie son indice, soit valeur n'est pas dans le tableau et la fonction renvoie -1.