

## ENSEIRB

Première Année Informatique 2003-2004

Épreuve de techniques de programmation (session de janvier)

### Remarques :

- Notes de cours et de TD autorisées.
- Durée de l'épreuve 2h.

## Exercice

Soit `appliquer(f, p1, p2, ..., pn, NULL)` une fonction à nombre variable de paramètres où

- `f` est un pointeur de fonction ayant pour paramètre et pour résultat un pointeur générique (de type `void *`);
- les `pi` sont des pointeurs génériques;
- le résultat est un tableau dynamique de pointeurs génériques contenant le résultat de l'application de la fonction `f` à chacun des paramètres `pi`.

Par exemple, l'appel de la fonction `appliquer(f, o1, o2, o3, NULL)` retourne un pointeur vers un tableau dynamique contenant `{f(o1), f(o2), f(o3)}`.

1. Écrire le prototype de la fonction `appliquer`, on pourra définir des types intermédiaires.
2. Écrire le corps de la fonction `appliquer`.

## Exercice

La commande `diff` d'UNIX permet de trouver les différences entre deux fichiers. Dans cet exercice, on se propose de réaliser en C une version simple de cette commande, notre commande s'appelle `my_diff` et prend comme arguments les noms de deux fichiers mais se limite à afficher `egaux` si les deux fichiers sont égaux et `différents` sinon.

1. Écrire un programme pour réaliser cette commande.
2. L'invocation de cette commande peut échouer. En effet, il suffirait que l'un des deux fichiers que l'on désire comparer n'existe pas, ou qu'il ne soit pas accessible en lecture pour que la commande ne marche pas.
  - (a) proposer une solution en utilisant les sauts non locaux (`setjmp/longjmp`)
  - (b) proposer une solution mettant en oeuvre le mécanisme de gestion des exceptions.

## Exercice

Dans cet exercice, on se propose de réaliser un module traitant des polynômes. Ce module proposera les fonctionnalités suivantes :

- créer un polynôme nul;
- détruire un polynôme;
- évaluer un polynôme en un double donné.  
Exemple : si  $P$  désigne le polynôme

$$P(x) = 2x^2 + 3x + 1$$

et  $a = 1.5$ , alors évaluer  $P$  en  $a$  retournera la valeur  $2 * (1.5)^2 + 3 * (1.5) + 1 = 10$

- faire la somme de deux polynômes;
- faire le produit de deux polynômes;

Afin de réaliser cette implémentation, on considérera un polynôme comme un pointeur sur un monôme. Un monôme étant défini par son degré, son coefficient, et un pointeur sur le monôme suivant. Dans notre exemple, le polynôme  $P$  est formé de 3 monômes : le monôme de coefficient 2 et de degré 2, le monôme de coefficient 3 et de degré 1, et enfin, le monôme de coefficient 1 et de degré 0.

Le type de base utilisé sera donc

```
struct monome{
    double coeff;
    long degre;
    struct monome *suiv;
};
```

1. proposer un fichier en-tête pour le module polynôme;
2. proposer un fichier source pour ce même module. N'écrivez pas toutes les fonctions décrites ci-dessus, mais seulement les fonctions pour créer et détruire un polynôme.

## Exercice

Soit programme suivant :

```
#include <stdlib.h>
#define F(x) x++

int main(){
    int x,y = 3,z = 2;
    x = F(y + z);
    printf("%d\n",x);
    return EXIT_SUCCESS;
}
```

1. Qu'affichera ce programme;
2. ce résultat est-il correct ? si la réponse est non le corriger.