

Cahier de TD/TP

Génie Logiciel

Qualité et maintenance du code

1°/ Analyse du code

Soit le code source suivant (Langage C):

```
#include<stdio.h>
#define TM 100
int main(void){
float x,v,c[TM];
int i,n;
float x0,s;
scanf("%d",&n);
for(i=0;i<=n;i++){scanf("%f",&c[i]);}
scanf("%f",&x);
v= c[0];x0=x;s=v;
for(i=1;i<=n;i++){s=s+c[i]*x0;x0=x0*x;}
v=s;printf("%f",v);
return(0);}
```

Analysez le code, que fait-il ?

Quelles sont les améliorations que vous pouvez apporter à ce code ?

2°/ Recherche d'erreurs

Le code source suivant correspond à un programme qui utilise la fonction *compare* (a, b) permettant la comparaison de deux chaînes de caractères a et b. La fonction retourne une valeur négative, nulle ou positive selon que « a » est lexico-graphiquement inférieure, égale ou supérieure à b.

```
#include<stdio.h>

compare(char a[TM],char b[TM]){
    int i;
    while((a[i]==b[i]) && ((a[i]!='\0') && (b[i]!='\0'))){
        i++;
    }

    if (a[i]==b[i]) return(0);
    if (a[i]=='\0' || b[i]=='\0') return(-2);
    if (a[i]>b[i]) return(1);

    return(-1);
}

int main(void){
    String a[TM],b[TM];
    int comp;
    printf("Saisissez un mot a et un mot b:");
    scanf("%s %s",&a,&b);
    comp=compare(a,b);

    switch (comp)
    {
        case 0: printf("%s et %s sont égaux! \n", a, b);
```

```
        case 1: printf("%s est lexico-graphiquement inferieure a %s\n",a,b);  
        case (-1): printf("%s est lexico-graphiquement superieure a %s\n",b,a);  
        default: printf("Il y a inclusion!\n");  
    }  
}
```

Analysez et corrigez les erreurs qui se trouvent dans le code.

Quelles sont les améliorations que vous pouvez apporter à ce code ?

3°/ Optimisation

Un client aimera disposer d'un programme qui lui permet de calculer le nombre de combinaisons possible de p objets parmi n . La fonction est donnée par la formule suivante :

$$C_n^p = \frac{n!}{p! (n-p)!}$$

Le client propose le test suivant (méthode main) pour vérifier le programme :

```
int main(void){  
    int n,p;  
    double c;  
    printf("Calculons C(n,p)\n");  
    printf("Entrez la valeur de n ");  
    scanf("%d",&n);  
    printf("Entrez la valeur de p ");  
    scanf("%d",&p);  
    c=factoriel(n)/(factoriel(p)*factoriel(n-p));  
    printf(" C(%d,%d) vaut %lf", n,p,c);  
    printf("\n");  
    return(0);  
}
```

Proposez une solution pour ce client.

Peut-on proposer une solution optimisée ?

Est-ce qu'on peut avoir une solution sans utiliser l'opération de multiplication* ?

$$* : C_n^p = C_{n-1}^{p-1} + C_{n-1}^p$$