

Fonctions

Fonctions

Les fonctions

- ☐ Réduit la complexité des programmes.
- ☐ Ensemble d'instructions servant à exécuter une tâche particulière dans la résolution d'un problème.
- ☐ Chaque fonction peut être testée et mise au point indépendamment des autres.
- ☐ Un programme peut se résumer à une séquence d'appels à des fonctions.

Fonction

```
Type_resultat Nom(liste des types et paramètres)
{
    partie déclarations de la fonction < au besoin >

    instructions de la fonction

    return(expression_de_retour); < au besoin >
}
```

Exemples :

```
int f(int a, int b)
{
    int c=a+b;
    return c;
}
```

```
void g(int a, int b)
{
    int c=a+b;
    cout <<c<<endl;
}
```

Fonction: les paramètres

- ☐ Les paramètres servent à augmenter l'utilisation de la fonction en obtenant de l'information de l'extérieur lors de l'appel.
- ☐ Il doit y avoir le même nombre d'arguments lors de l'appel que de paramètres définis dans la fonction.
- ☐ L'ordre des arguments dans l'appel de la fonction est important puisque c'est la position de l'argument à l'appel qui détermine le paramètre associé.

Transmission de paramètres

Par valeur

- La valeur de l'argument est transmis au paramètre correspondant;
- Le paramètre est initialisé à la valeur reçue.
- Lorsque le paramètre est modifié, l'argument n'est pas modifié.

Par référence (C++) et par adresse (C et C++)

- L'adresse de l'argument est transmise au paramètre correspondant.
- Le paramètre est initialisé à l'adresse (de l'argument) reçue et par le fait même à la valeur de l'argument.
- Lorsque le paramètre est modifié, l'argument est modifié

Transmission par valeur et par référence (C++)

Fonctions

```
#include<iostream>
using namespace std;
void f(int a)
{
    a=a+10;
}
void g(int &a)
{
    a=a+10;
}
```

Paramètre passé par valeur

Paramètre passé par référence

```
int main()
{
    int a=5,b=5;
    cout <<"Avant : a=" <<a<<" , b="<<b<<endl;
    f(a);
    g(b);
    cout <<"Après : a=" <<a<<" , b="<<b<<endl;

    return 0;
}
```

Avant : a=5, b=5
Après : a=5, b=15

Transmission par valeur, par référence et par adresse

Fonctions

```
void f(int a)
{
    a=a+10;
}
void g(int &a)
{
    a=a+10;
}
void h(int* a)
{
    (*a)=(*a)+10;
}
int main()
{
    int a=5,b=5,c=5;
    cout <<"Avant : a=" <<a<<" , b="<<b<<" , c="<<c<<endl;
    f(a);
    g(b);
    h(&c);
    cout <<"Après : a=" <<a<<" , b="<<b<<" , c="<<c<<endl;
    return 0;
}
```

Paramètre passé par valeur

Paramètre passé par référence

Paramètre passé par adresse

Avant : a=5, b=5, c=5
Après : a=5, b=15, c=15

Transmission de paramètres

- ❑ La transmission par valeur sert à fournir à la fonction une valeur dont elle a besoin pour exécuter correctement ses instructions. On qualifie le paramètre (IN).
- ❑ La transmission par adresse sert à retourner une valeur au point d'appel via l'argument. Dans ce cas on qualifie le paramètre (OUT). La fonction peut, au besoin, utiliser une valeur initiale. Dans ce cas on qualifie le paramètre (IN\OUT).

Transmission de paramètres

- ☐ Il doit y avoir correspondance entre les arguments de l'appel et les paramètres de la fonction pour ce qui est de leur nombre et leur type.
- ☐ L'argument dans un appel doit absolument être une variable si le paramètre correspondant est transmis par adresse. Il ne peut être une constante ni une expression.
- ☐ Si le paramètre est transmis par valeur, l'argument dans l'appel peut être une variable, une constante ou une expression.

12345678910

Un tableau comme paramètre

Toujours transmis par adresse (sans &).

```
#include<iostream>
using namespace std;

void f(int t[])
{
    t[0]+=10;
}

int main()
{
    //initialisation d'un tableau de 5 éléments
    int t[]={6,3,7,8,2};

    cout <<"Avant : t[0]="<<t[0]<<endl;
    f(t);
    cout <<"Après : t[0]="<<t[0]<<endl;

    return 0;
}
```

Avant : t[0]=6
Après : t[0]=16

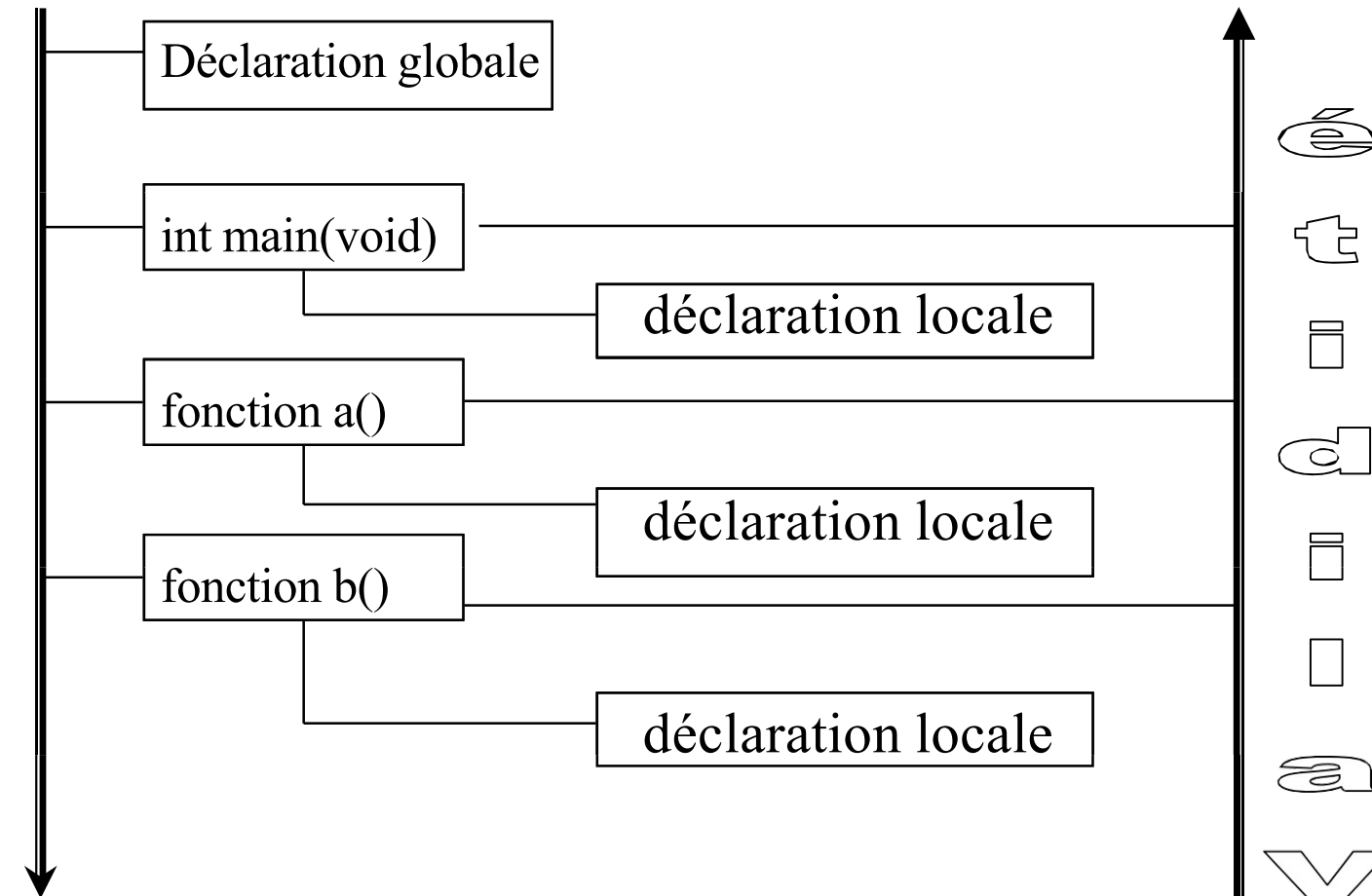
Fonction récursive

Dans le code la fonction, on **fait appel à la fonction elle-même**

```
int fact(int n)
{
    if(n==0)
        return 1;

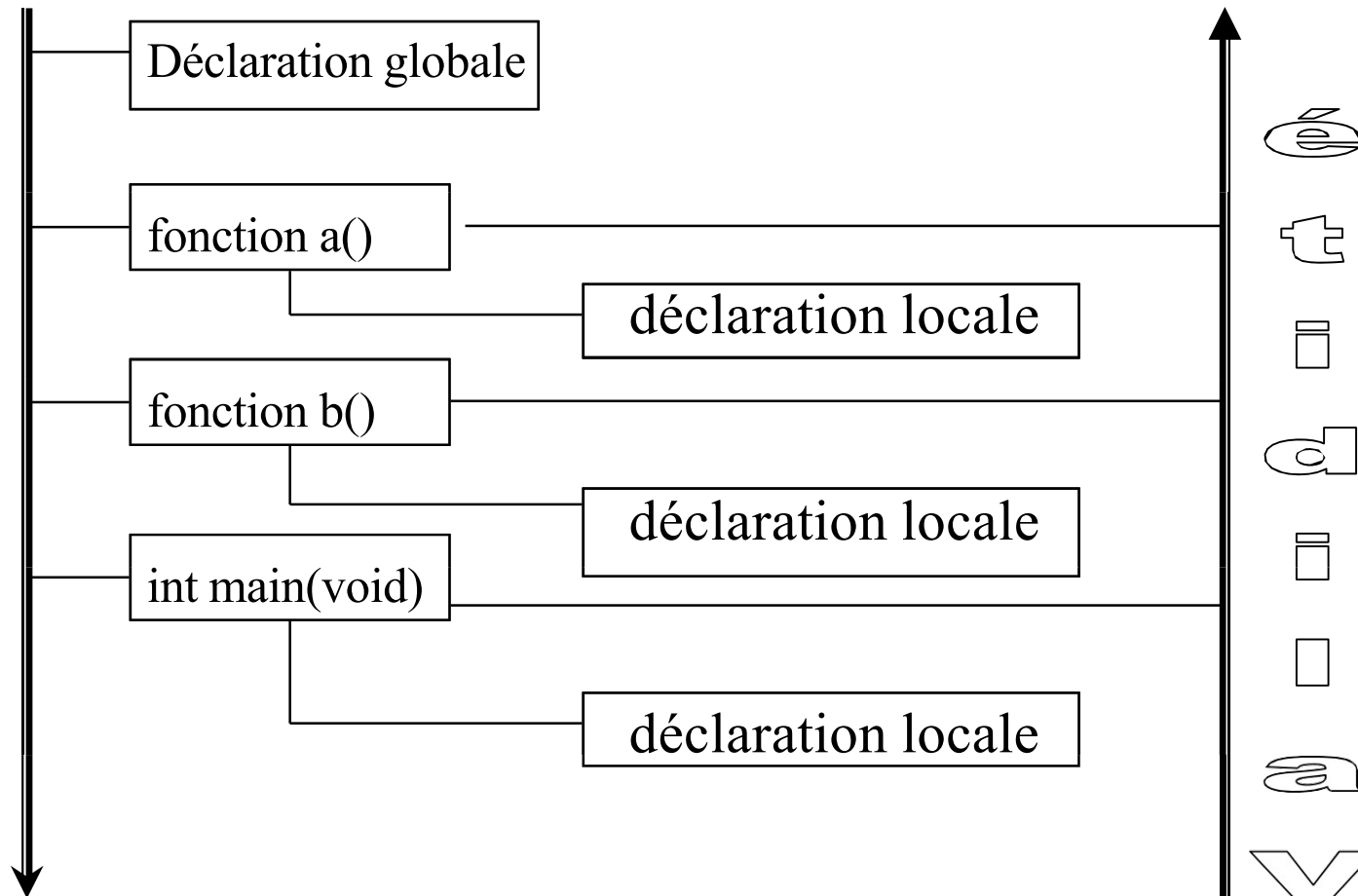
    return n*fact(n-1);
}
```

Domaine de validité



La fonction main() n'a accès à aucune
fonction ==> **prototypage**

Domaine de validité



La fonction main() a accès à toutes les fonctions déclarées auparavant.

```
#include <iostream>
using namespace std;
```

```
double totalCost(int numberParameter, double priceParameter)
{
    const double TAXRATE = 0.05; //5% sales tax
    double subtotal;

    subtotal = priceParameter * numberParameter;
    return (subtotal + subtotal*TAXRATE);
}
```

Entête de
fonction

Définition de la
fonction

Corps de la
fonction

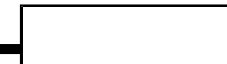
```
void main( )
{
    double price, bill;
    int number;

    cout << "Enter the number of items purchased: ";
    cin >> number;
    cout << "Enter the price per item $";
    cin >> price;

    bill = totalCost(number, price);

    cout.setf(ios::fixed);
    cout.setf(ios::showpoint);
    cout.precision(2);
    cout << number << " items at "
         << "$" << price << " each.\n"
         << "Final bill, including tax, is $" << bill
         << endl;
}
```

Appel de
fonction



Fonctions

Emplacement des fonctions

```
#include <iostream>
using namespace std;
```

```
double totalCost(int numberParameter, double priceParameter);
//Computes the total cost, including 5% sales tax,
//on numberPar items at a cost of pricePar each.
```

Déclaration de
fonction ou
prototype

```
void main( )
{
```

```
    double price, bill;
    int number;
```

```
    cout << "Enter the number of items purchased: ";
    cin >> number;
    cout << "Enter the price per item $";
    cin >> price;
```

```
    bill = totalCost(number, price);
```

Appel de
fonction

```
    cout.setf(ios::fixed);
    cout.setf(ios::showpoint);
    cout.precision(2);
    cout << number << " items at "
         << "$" << price << " each.\n"
         << "Final bill, including tax, is $" << bill
         << endl;
```

```
}
```

```
double totalCost(int numberParameter, double priceParameter)
{
```

```
    const double TAXRATE = 0.05; //5% sales tax
```

```
    subtotal = priceParameter * numberParameter;
    return (subtotal + subtotal*TAXRATE);
```

Entête de
fonction

Corps de la
fonction

Définition de la
fonction